ixia
A Keysight Business

BreakingPoint

User Guide

Version 8.50

Notices

Copyright Notice

Warranty

The material contained in this document is provided “as is,” and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Keysight disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Keysight shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Keysight and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is "commercial computer software," as defined by Federal Acquisition Regulation ("FAR") 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement ("DFARS") 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at http://www.keysight.com/find/sweula or https://support.ixiacom.com/support-services/warranty-license-agreements. The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data. 52.227-14 (June 1987) or DFAR 252.227-7015 (b) (2) (November 1995), as applicable in any technical data.

This page intentionally left blank.

Contact Us

Ixia headquarters

26601 West Agoura Road
Calabasas, California 91302
+1 877 367 4942 – Toll-free North America
+1 818 871 1800 – Outside North America
+1.818.871.1805 – Fax
www.ixiacom.com/contact/info

Support

Global Support	+1 818 595 2599	support@ixiacom.com
APAC Support	+91 80 4939 6410	support-asiapac@ixiacom.com
EMEA Support	+40 21 301 5699	support-emea@ixiacom.com
Greater China Region	+400 898 0598	support-china@ixiacom.com
India Office	+91 80 4939 6410	support-india@ixiacom.com
Japan Head Office	+81 3 5326 1980	support-japan@ixiacom.com
Korea Office	+82 2 3461 0095	support-korea@ixiacom.com
Singapore Office	+656 494 8910	support-asiapac@ixiacom.com

CHAPTER 1 About This Guide

This section provides safety information and a list of related documentation.

Related Documentation

The table below lists the documentation related to BreakingPoint. The latest documentation for each release can be found on the Ixia Support website.

Related Documentation

Documentation	Description
BreakingPoint Installation Guides	Provides installation instructions and information for the BreakingPoint system.
BreakingPoint User Guide	Provides information on how to use the Control Center to set up, customize, and run traffic through devices under test.
BreakingPoint Release Notes	Provides information about new features, resolved customer issues, known defects and workarounds (if available).
BreakingPoint Online Help	Online documentation for all BreakingPoint products. Proper viewing will require a supported HTML browser, see Software Specifications on page 1467 for details.

Safety and damage messages

This is the safety alert symbol: 

It is used to indicate a hazard. Specific keywords and additional graphical representations are used to explain the specific nature of each hazard.

The following table describes the safety warning and equipment damage icons and textual conventions used in this document.

Type of Hazard	Icon	Description
CAUTION		Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
WARNING		Indicates a hazardous situation that, if not avoided, could result in death or serious injury. Specific icons may be used to indicate specific types of warnings.
		Laser radiation hazard warning.
		Electrostatic discharge hazard warning.
		High voltage hazard warning.
		Hot surface hazard warning.
		Moving fan blades hazard warning.
DANGER		Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
NOTICE		Indicates a condition or occurrence that could result in property damage.

CHAPTER 2 Frequently Asked Questions

This section provides answers to some of the most frequently asked questions. If you have any questions you would like added to this section, please send them to support@ixiacom.com.

Account Questions

I've had 4 invalid login attempts to the Control Center, and my account is now locked. How do I unlock my account?

Close the Control Center window and open a new browser window.

How do I reset my Control Center account password?

You can have another user log into the Control Center to reset your password; you can log into the BPS Management port to reset the password; or you can telnet to the system's management IP address to reset the password.

Addressing Questions

How do I configure the system to use one MAC address per host?

If you edit the Network Neighborhood selected for your test, you can select "Host" as the type for the domain. This will allot one MAC address per host; selecting "Virtual Router" will use one MAC address total for all traffic from that subnet.

Why would I want to use one MAC address for all hosts?

A device has limited memory dedicated to its ARP table. If it takes too long for the ARP table to populate, the device may run out of buffer packets for that host and drop packets. So, you will want to use the "Virtual Router" option when using more addresses than the device's ARP table is capable of handling. Otherwise, entries will be dropped before they need to be used.

Can NAT be used across multiple test components?

No. Only one test component can use a domain that has NAT enabled. Any domain that has NAT enabled cannot be shared between test components.

How many subnets can I add to a domain?

The number of subnets that can be added depends on the type of subnet you are defining. Each domain can contain one non-VLAN subnet; each additional subnet must have a VLAN ID assigned to it. So, theoretically, the limit is 4,095 because you can assign VLAN IDs from 1-4,095.

How do I assign one IP address per subnet?

If you edit the Network Neighborhood selected for your test, you can enter in the single IP address you want to use in the **Minimum Range** field.

What type of Network Address Translation (NAT) is supported?

Source NAT, also known as Traditional NAT, Outbound NAT, or Unidirectional NAT and Destination NAT (DNAT).

Can I send and receive traffic on the same interface?

Yes. You can send and receive traffic on the same interface if you assign the interface a domain that has VLAN-tagging enabled.

How do I set up a one-arm server?

You will need to enable the server interface, but not the client interface, in a test configuration.

When would I want to set up a one-arm server?

You will want to set up a one-arm server if you have a client you want to communicate with the BreakingPoint system. This is typically a client that is operating independently, or a test set up in which you have a test component communicating with the proxy, and the proxy communicates with the server component on the client's behalf.

Application Error

I am receiving application errors during my test. Is there any additional information that can help me resolve these errors?

Error messages such as "Application failure" can appear in or outside of a NAT scenario when a DUT is not properly configured or willingly drops packets. Make sure to check the DUT logs and stats when these errors occur.

Application Traffic

How is application response time defined?

Application response time is time between the start and end of a transaction. For example, in the context of a simple HTTP request, the client request is the start of transaction, the server response is the end of transaction. Often these fit into single packets. In the context of something that is just a bulk file transfer, the transaction ends when the file is transferred.

How do transaction flags work?

If you do not manually set the transaction flags for your application flows, they are automatically set to the first and last packet. If there is just an end of transaction flag and no start flag, a start transaction flag is set on the first packet. If there is no end transaction flag, it is set on the last packet. If there is more than one transaction in a flow, and the last one has no end of transaction flag, then the last packet is set as the end of transaction. If a second transaction is nested within an outer transaction, its transaction flag is ignored.

How are application successes defined?

Application Success means that a transaction started, and then ended. If a flow is interrupted between the start and end, then it is a failure. For example, if it is a TCP flow, scenarios where the flow closes prematurely (e.g., retries, external RST, etc.) are counted as a failure. If it is a UDP flow, and the flow does not send a packet through, then this is counted as a failure. Generally, if a flow sends the last packet with an end of transaction flag, then it is considered a successful flow. However, if no transaction flags are explicitly set, then Application Success is defined as all the packets in the flow getting sent.

Note: Conflicting totals for the number of transactions that are attempted and successful are sometimes displayed in some sections of the Summary tab of the Real-Time Statistics page and in the Application Summary section of the report. Additionally, this issue causes the transactions attempted and successful to total zero (0). When this occurs, manually set the `transaction_start` flag on the second action in the Super Flow.

When should I set the transaction flags?

Transaction flags enable you to put multiple transactions on a single flow (e.g., HTTP 1.1, a database connection, FTP, etc.). This feature enables you to define what a transaction is; if you do not define where your flows start and end, then the system will not count the whole flow as a transaction.

Note that in some instances, the `transaction_end` flag is set on the Quit action that is in the "no match" case in the last Conditional Request. When this happens, successful transactions are omitted from the Real-Time Statistics count totals.To correct this error, add a Quit action in the matching pattern on the last Conditional Request and set the `transaction_end` flag on that action.

If a flow does not have any actions with the `transaction_end` flag set, the Network Processor will set this flag on the last action in the flow. You can choose to manually set the `transaction_end` flag on an action. If you do this, be aware the Conditional Request action can cause different actions to be taken based on match results. Because of this, there is more than one possible last action in the flow. Make sure that each last action has the desired transaction flag setting.

Bandwidth Questions

How do I define the maximum throughput for each test interface?

The maximum throughput is defined using the Data Rate parameters. This parameter is defined per test component, and it is the upper-bound rate for each interface, which means that the interface will never send more traffic than the value specified. For the session-based components, you can define the scope of the data rate, which enables you to set the maximum data rate per interface, or set the aggregate data rate for the entire test component.

What is the maximum throughput for each interface?

The maximum throughput is determined by the link speed of the device connected to the chassis.

How do I determine how much bandwidth each test component is using?

The system has a test status verification feature that tells you whether or not the test components have exceeded the maximum allowed bandwidth for each interface.

Why is the estimated bandwidth for my Recreate component nearly twice the actual bandwidth estimate for my capture?

The Recreate test component does not determine whether traffic was captured from either the server or client, so it estimates the bandwidth usage based on traffic coming from both interfaces.

For example, if you capture 500 Mbps of traffic on Interface 1, then the corresponding Recreate test will estimate that the data rate is 500 Mbps for both the transmitting and receiving interfaces. To set the data rate to be an aggregate sum for the test component, set the Data Rate Scope parameter to Limit Aggregate Throughput.

What is the maximum bandwidth usage for a test interface?

For test components that send bidirectional traffic – such as Session Sender, Application Simulator, and Recreate – the value defined for Frame Rate Distribution sets the upper bound limits for bandwidth usage per interface. However, the aggregate sum of the traffic sent by each interface will fluctuate between the data rate shared between both testing interfaces.

For example, if you have if a Session Sender test that uses 500 Mbps, then the test will never send more than 500 Mbps from an interface; however, the sum of traffic sent by both interfaces will fluctuate between 500 Mbps and 1000 Mbps.

BPS VE Performance Acceleration Mode Questions

BPS VE supports a performance acceleration mode based on DPDK support.

Before enabling Performance Acceleration, you should be aware of the following notes and prerequisites:

- A maximum of 4 components per vBlade can be run in performance acceleration mode. To run a maximum of 8 components per vBlade, the "Enable Performance Acceleration" option needs to be unchecked.
- Performance Acceleration is not supported for SRIoV and KVM Hypervisor.

Prerequisites for Performance Acceleration:

- vBlade processor should have SIMD extensions SSSE3 or above enabled.
- At least 8GB of RAM per vBlade.
- Ixia recommends using VMware ESXi 6.0 with build number 3029758 or above.
- Ixia recommends using the default settings of **Hypervisor** > **Configuration** > **Software**> **Advance Settings**> **Net**.

To enable Performance Acceleration:

1. Each vBlade on the Device Status page of the GUI displays a slot configuration button at the top-right corner. Click the **Slot Configuration** button.
2. Select the **Enable Performance Acceleration** option.
3. Click **Apply**.

Capture and Recreate Questions

Is there a size limit for the PCAP files captured by the BreakingPoint Storm?

Yes. There is a 2 GB limit per port on a 10 Gb blade and a 1 GB limit per port on a 1 Gb blade on the BreakingPoint Storm. The BreakingPoint FireStorm has a 4 GB limit on each port.

What happens if the capture exceeds the size limit?

The BreakingPoint system has a circular packet buffer, which means that older content will be overwritten with new data if the size limit is exceeded.

What type of capture files can I import?

Any libcap-compatible PCAP file. To successfully import PCAP files with Recreate in normal mode or in raw mode, the PCAP file must contain at least a TCP or UDP payload.

How do you know when a packet is sent from our device if we don't use a signature tag?

Recreate does not detect packet modifications. It tracks packets by the 3 or 5 tuple for the network flow and by the TCP sequence number. It compares the length and flow ID to determine which flow the packet belongs to.

Is there a size limit on the imported PCAP file?

Yes, there is a 4GB limit on imported PCAP files.

Is there a limit on the total amount of space that my imported PCAP files take up?

No. Currently, there is no imposed limit.

Can I use an external drive to capture traffic?

No. You should use the import tool to import PCAP files, or you should use the BreakingPoint system's packet capture buffer.

Do you modify the internal IP addresses with the Capture and Recreate feature?

No. We do not monitor internal addressing. There is no introspection into the protocols to track state per flow, so the traffic is stored as flows of UDP or TCP over IP.

How can I modify traffic that is captured on the BreakingPoint system?

Once you have created a Recreate test, you can modify any of the Recreate parameters if the **General Behavior** parameter to **Use User-specified settings**. Note that if **General Behavior** is set to **Use User-specified setting**, then the system will overwrite the settings in the PCAP file with the values defined for all of the Recreate parameters.

When I import a capture file, I get an error that says, "Invalid File Format on Capture Import". What does this mean?

If the import process detects that the capture packets have been truncated, you will get this message. The file format must be a standard PCAP file. You should make sure that the full contents of the packets are captured by setting the -s 0" option (if you are using Tcpdump).

How do I modify the payload of captured traffic?

You can use a third party tool to edit your PCAP data. After you've edited the payload, you can import the PCAP into the system.

Why is my imported traffic not replaying exactly as it was captured?

The Recreate component does not replicate the IP/TCP/Ethernet headers from the PCAP file, and it does not play back the exact number of flows from the file. Instead, the component will extract the application payloads from the PCAP file and play them back to the device under test. However, the application flows will not be played back in the same order; they will be played back randomly.

What is the difference between Use capture file settings and Use user-specified setting?

Use capture file settings essentially lets you replay the PCAP as it is, whereas **Use User-specified settings** enables you to control how fast or slow the traffic is replayed. For example, **Use capture file settings** will use the data rate, maximum simultaneous sessions, sessions per second, test duration, inter-packet delays, application payloads, and destination ports from the PCAP file. Whereas **Use User-specified settings** will only use the application payload and destination ports from the PCAP file.

Are there any limitations on the total number of PCAP files that I can import?

The number of PCAP files you can import with Recreate is limited only by the amount of available disk space.

How does Recreate estimate its bandwidth usage?

Traffic is captured using a single port; therefore, it is not possible to determine with certainty the direction of individual flows within the capture. When Recreate estimates the bandwidth usage required to generate traffic based on a capture, it does not know in advance if the traffic seen was all from the source interface to the destination interface or vice versa; therefore, it makes a worst-case estimate that the maximum bandwidth seen during the capture could result in transmitted traffic out either interface. For example, if you capture 900 Mbps/second of traffic on port 1, the corresponding Recreate test will estimate a worst-case of 900 Mbps/second on both the source and destination interfaces.

Why does the Replay Capture File In Single Mode only play UDP flows and not play any TCP flows?

This mode accepts the first 255 flows. Since the first 255 flows of the default pcap have only UDP flows, the Replay Capture File In Single Mode will play only UDP flows and will not play any TCP flows.

Endpoint Testing Questions

How do I configure my system to do endpoint testing?

First, you will need to set up external addressing for the device. To do this, go to the Network Neighborhood and configure addressing for the External tab. Once you have set up the external addressing, you will need to select the External interface for the test component.

Which test components can I use for endpoint testing?

You can use the Session Sender, Stack Scrambler, Client Simulator, and Security test components for endpoint testing.

Exporting Tests

If I export a test that uses imported data, such as certificates and Super Flow files, will that data be bundled with the exported test?

You can elect to have the imported PCAP file bundled with the test when the test is exported. If you do not make this selection, the system will not export the imported data.

IPv6 Questions

Which component can I use to generate IPv6 traffic?

You can use Routing Robot to generate IPv6 traffic.

How do I set up a test that generates IPv6 traffic?

First, create a test that uses the Routing Robot test component. Next, go to the Parameters area of the Routing Robot test component. Find the parameter **IP Header Version**, choose **IPv6**, and click the **Apply Changes** button; this will enable IPv6 traffic generation. Next, find the parameters that are tagged with the IPv6 label, and customize the fields to meet your specifications. Once you are done, click the **Apply Changes** button.

Do I have to make any changes in the Network Neighborhood in order to generate IPv6 traffic?

No. The Routing Robot test component will translate the addresses from the Network Neighborhood to IPv6 addresses on the wire.

Can I configure the IP address for the system to be an IPv6 address?

Yes. You can configure the IP address for the system to be an IPv6 address through the CLI. When you perform the initial configuration for the system, you will need to assign IPv4 addresses for the system's IP address and gateway. However, after you have performed an initial configuration for the system, you can use the `updateNetwork` command to configure an IPv6 address for the system.

See the example below.

```
bps> networkInfo
dhcp="no"
hostname="bps.ixiacom.com"
ip="10.10.10.10"
netmask="24"
gw="10.10.10.1"
currip="10.10.10.10"
currmask="24"
dns1="10.10.10.11"
dns2="10.10.10.12"
bps> updateNetwork-ip 2000:0:0:5::b5 -netmask 64 -gw 2000:0:0:5::b3 -dns1
2000:0:0:5::b9 -dns2 2000:0:0:5::b2
```

Load Profiles

Which test components support Load Profiles?

You can use Load Profiles with Application Simulator, Client Simulator, Recreate, and Session Sender.

What are Load Profiles?

Load Profiles allow you to customize the behavior of TCP sessions during the different phases of an Application Simulator, Client Simulator, Recreate, or Session Sender test.

Multi-box Testing Questions

How many boxes can I use at one time?

You can add up to five boxes to a multi-box test.

What is multi-box testing?

Multi-box testing allows you to use a single BPS management interface to control up to four additional boxes.This allows you to simultaneously run tests from different boxes.

Are test series supported in multi-box testing?

No. Test series are currently not supported.

What are some best practices to use for multi-box testing?

Because all data for a multi-box test is copied from the primary system to the secondary systems, it is recommended that the names for capture files, Strike Lists, App Profiles, DUT Profiles, Network Neighborhoods, and tests on the secondary systems are not the same as the ones on the primary system. Any data on a secondary system that are shares a name with data on the primary system will automatically be overwritten by the data from the primary system.Additionally, all ports on the secondary systems must have the same Active Group assignment as the primary system. For example, Slot 1's ports on the primary system are assigned to Active Group 1, then all ports that will be used on the secondary systems must also be assigned to Active Group 1.

Can I administer multiple boxes with a single system?

No. For administrative tasks – such as managing user accounts and updating the system – you still must log into each individual box to administer the system.

Do all systems in a multi-box test have to run the same ATI updates and firmware versions?

Yes. All systems must have the same ATI updates and firmware versions installed.

How many tests can I run with a multi-box test?

You can have up to five tests in a mulit-box test: one test for each system in a multi-box set up. This number is limited by the number of systems supported by the multi-box feature.

Packet Buffer Export

Can I perform more than one export at a time?

No, only one export can be done at a time.

Is there a size limit on the buffer?

Yes, the BreakingPoint Storm has a 2 GB size limit on the buffer. The BreakingPoint FireStorm has a 4 GB size limit on the buffer.

What happens if the packet buffer reaches its capacity?

Once the packet buffer reaches its capacity, the older content will be overwritten with newer data.

Port Mapping

Can I change the port mappings?

Yes, you can change the port mappings from the Device Status screen. First, select the Active Group whose ports you want to modify, and then click on the **Open port mapping options** button. From this screen, you use the drop-down buttons located under each interface to change the port/slot mapping.

What are port mappings?

Port mappings map ports on the BreakingPoint system to an interface in the Network Neighborhood.

What is the purpose behind port mappings?

Port mappings allow you to virtually "rewire" your port connections without having to physically enter the lab to do it yourself.

Port Reservations

Do I have to reserve ports in order to run a test?

Yes. You must have locked port reservations if you want to run a test. If you are running a test that uses a non-VLAN Network Neighborhood, then you must lock at least 2 port reservations. However, if

you are running a test that uses a VLAN-enabled Network Neighborhood, then you only need one locked port reservation.

What is the difference between a locked port reservation and a regular port reservation?

A locked port reservation provides you with the ability to run tests and export packet buffers from the ports. A regular port reservation simply reserves the port under your account; no other users can use these ports, however, there's not much you can do with these ports until you have locked the reservation on them. To lock a port's reservation, simply click on the port. All ports that have locked reservations under your account will have a key icon displayed over them.

Another user has a slot reserved. How can I reserve those ports for myself?

If you click on a reserved port, the system will ask you if you would like to force reserve the port. If you click **Yes**, the system will reserve all ports on that slot under your account, while lock reserving the port you clicked on.

What is the difference between a port that has a lock icon and a port that has a key icon?

A port that has a lock icon has been reserved by another user. A port that has key icon is reserved by you.

Port Settings

For the BPS-10K and BPS-1K, I was able to manually set the port speed. Can I manually set the port speed for the BreakingPoint system?

Yes. From the Device Status screen, you can right-click on a port and select Configure Port. From here, you can select a port speed that is available from the Speed Settings drop-down box.

Reporting Questions

How is a flow defined?

A flow includes both UDP and TCP flows.

What is the difference between a flow and a connection?

In the report, a flow is counted when a packet is sent on a particular 5-tuple, regardless of whether an actual TCP connection is established or not. A connection, on the other hand, is counted only when a finishing handshake has created a new connection.

Do you track UDP connections?

No. Since UDP flows are stateless, only statistics for UDP flows are posted.

What is the difference between connections per second (cps) and sessions per second (sps)?

Connections per second refers to only the rate at which sessions are opened. Sessions per second refer to the rate at which sessions are opened, data is sent, and closed.

Why does the Traffic Overview section of the report for my RFC 2544 test show that it has received slow start packets at every data rate?

The BreakingPoint system will send slow start packets in the reverse direction to the DUT for each iteration, enabling the DUT to identify the MAC addresses used by the BreakingPoint system.

I am trying to view several multi-box reports at once; however, after I open five reports, my browser will not load any additional reports. Is there a limitation on the number of reports I can have open?

We do not impose a limitation on the number of reports you can have open; however, the number of reports you can view at a time may be restricted by the Web browser you are using. Therefore, we recommend that you do not open more than five multi-box test reports at a time. If you experience any problems after you have attempted to open multiple reports, you should log out of the Control Center and log back in again.

Why is my report is missing the Ethernet Data Rates section?

Either the test's duration was not long enough or there were not enough frames transmitted for the Ethernet Data Rate to be calculated. To get results for the Ethernet Data Rate, try increasing the duration of the test (either in frames or in seconds).

What e-mail server is used to send our reports?

The BreakingPoint system will act as a mail server. It retrieves the IP address of the SMTP server via DNS. It will use the DNS server and hostname you specified during the initial configuration of the system.

To see what your DNS server and hostname settings are, telnet to the chassis. After you log into the box, use the networkInfo command to display the network configuration for the BreakingPoint system.

To edit the network information, use the updateNetwork command and any of the following options - hostname <dhcp hostname>, -ip <IPaddress>, -netmask <netmask>, -gateway <gateway IP address>, -dns1 <DNS server>, -dns2 <DNS server>, and -dns3 <DNS server>.

Why is the count of received multicast packets in my report larger than expected?

When multiple ranges are defined on the same physical interface, the total number of received multicast packets reported by the test will be multiplied by the number of ranges defined.

RFC 2544 Questions

Why can't I save my own copy of the RFC 2544 test?

Currently, the BreakingPoint system only allows you to have one working copy of the Quick Test - RFC 2544. Every time a user modifies and saves the test, it will overwrite the existing test settings.

Can I export a copy of the RFC 2544 test?

No, you cannot export a copy of the RFC 2544 test.

Which interfaces does the RFC 2544 test use?

The RFC 2544 test uses logical interface 1 as the transmitting interface and logical interface 2 as the receiving interface. These interfaces are normally defined on the Test Editor page; however, for the RFC 2544 test, these interfaces are automatically defined for you.

Can I change the logical interfaces assigned for the transmitting and receiving interfaces?

No, you cannot; however, you can remap the slot/port assignments for each interface.

Security Questions

Can I resend the same attacks every test?

Yes. If you set the random seed to a non-zero value, the system will generate static content for each Strike.

How long does it take to run the BreakingPoint All Strikes Strike List?

The amount of time it takes to run this Strike List will vary depending on the evasion options that you have set. A Security test running this Strike List can take up to 33 hours to run.

Is there a Strike List that contains non-fuzzing Strikes?

Yes. The BreakingPoint Strike Level 5 Strike List contains only non-fuzzing Strikes.

What is the random seed?

The random seed generator allows you to either generate dynamic or static content for each Strike. Setting the random seed to '0' will generate random content for each Strike. Any other value defined for the random seed will keep the contents of the Strike static. This is useful in cases where you want to continually resend the same exact Strikes; however, if any settings in the Strike List is changed (e.g., evasion options, adding/removing Strikes, etc.), the seed is modified, or ATI updates have occurred, then the content of the Strikes will not be retained.

Can I import my own attacks?

Yes. You can use the import PCAP capture tool to import your own attacks or you can use the Application Manager to create your own.

What does "Strike Error Count" mean in the Security test results?

These are Strikes that encountered an error. For information, please contact our support team at support@ixiacom.com or 1-818-595-2599.

What does "Blocked Open" mean in the Security test results?

The Strike was blocked because the session could not be opened. This will happen when a TCP attack is blocked by a firewall rule (e.g., "Block all traffic on Port 80").

Are your Strikes tested against real servers?

Exploits for high-profile vulnerabilities are validated against real servers before being released in an ATI Update. However, we do not verify every Strike.

Can I designate which port a strike is sent on?

Yes. You can designate the port by modifying the Strike options for the Evasion Profile. To set the Strike options, create a new Strike List; add a new Evasion Profile; add your Strikes; open the Strike Options window; and set the destination port.

How will I know if an existing Strike has been modified by an ATI Update?

You will need to check the release notes for the ATI Updates to see which Strikes have been modified.

How do I set evasion techniques?

You can create evasion techniques by modifying the parameters found in the Evasion Profile section of the Security test's Parameters tab. Please note that security overrides from previous versions of the BreakingPoint system cannot be migrated to version 1.5.1.

Is there an order to how the Security component sends out Strikes?

Yes and no. It depends on whether your Evasion Profile is made up of all individual Strikes or intermixed with StrikeSets. If you have all individual Strikes, then the Strikes will be sent out in the order in which they were added. If Strikesets are included in an Evasion Profile, then the order is random.

How do I simulate an ICMP flood?

After adding a Session Sender component to your test, click the edit component button. Locate the Payload parameter and select ICMP from the Transport drop-down menu. If you are using the Routing Robot component in your test, locate the Packet Templates parameter and select one of the ICMP templates from the Type drop-down menu.

Session Questions

I'm running an Application Simulator test whose ramp-up behavior is Full Open + Data. It looks like the system is only sending DNS queries. Why am I not seeing any TCP packets during ramp up?

During ramp up, the system will be sending DNS flows to the DUT. Using Full Open + Data as the ramp up behavior will cause the system to keep these flows open, so the system will not be able to send TCP packets. Therefore, if you want to be able to send TCP packets during ramp up, you should use the Full Open + Data + Close ramp up behavior.

Why can't I reach the number of sessions I've specified in my Client Simulator or Session Sender test?

For session-based tests, the BreakingPoint system uses the Network Processor, which has a maximum of 28 workers. Each worker can support up to 26,785 sessions/sec (i.e., 750,000 / 28). The Network Processor cannot allocate a partial worker to a test component, so you may not be able to generate the maximum number of possible sessions supported by the BreakingPoint system.

Can I generate a single high-throughput TCP session?

Yes. You can use the Session Sender component to send a single TCP stream at 1 Gbps. You will need to set the following parameters in your Session Sender test: TCP Session Duration (segments) to 1 and Data Rate.Minimum Data Rate to 1000. When setting the Data Rate, verify that the **Data Rate Type** is set to **Constant**.

In my Session Sender test, I have the ramp down behavior set to Half Close, which should omit the last ACK. However, I noticed that the client continues to retransmit the final ACK to the server. Why is this happening, and why is the retransmission coming from the client side, instead of the server side?

This is currently how the BreakingPoint system is designed. This specific issue should be addressed in a future firmware update.

What is the difference between a flow and a connection?

A flow can occur any time a packet is sent. It does not require that a TCP connection be established to be counted as a flow. A connection, on the other hand, requires that the TCP connection be established and finished.

How do I simulate a SYN attack?

There are two ways to do this: you can either select the preset **SYN Flood** for the Session Sender component or you can manually set up a SYN flood using Session Sender. If you are customizing a Session Sender component, select **SYN Only** for the parameter **Session Ramp Up Behavior** and then define **Session Ramp Distribution / Ramp Up Duration**. So, for the time specified for Ramp Up Duration, the system will only send TCP SYN packets.

I have a Session Sender test whose ramp down behavior is full close. However, looking at my test results, I've noticed that the Closed by Reset counter (under the TCP Summary area) has recorded some values. Why is this a non-zero value?

If your test was unable to close all the sessions before the test completed, then you will see these resets recorded in your report. This occurrence typically happens with the Session Sender presets because the ramp down phase was not long enough for the sessions to close.

How do I force sessions from entering the TIME-WAIT state?

You should select the **Open and Close with Reset Response** option for the Steady-State behavior. This will force sessions to wait for the server to end the sessions, and allow the client to respond with a RST.

What is the maximum number of sessions the system supports?

The BreakingPoint Storm allows a maximum total of 20,000,000 sessions on a 10Gb system. The BreakingPoint FireStorm allows a maximum total of 90,000,000 concurrent sessions per chassis across all session-based test components – this includes Session Sender, Application Simulator, and Recreate – in a test. You can use the Maximum Simultaneous Sessions parameter to set the maximum number of sessions allowed per test component.

For example, for the BreakingPoint Storm, if Test A contains a Session Sender and an Application Simulator test component and the value defined for Maximum Simultaneous Sessions for the Session

Sender test component is 3,000,000, then the value defined for Maximum Simultaneous Sessions for the Application Simulator test component cannot exceed 4,500,000 (10Gb) or 2,000,000 (1Gb).

Why does the segment size I've set in the component not match the segment sizes of the traffic on the wire?

When you are running an SSL test, the system will encapsulate the segments; this will increase the segment overhead, which is why you are seeing larger segment sizes on the wire. We typically include an additional 8 or 16 bytes into each segment for these purposes. So, when you are setting the value for the segment size, you should take into consideration that this is the unencrypted size of the segment on the unencrypted side of the SSL proxy.

Why is the Client Connection Rate higher than the Server Connection Rate?

The client and server do not open and close TCP sessions at the same time; therefore, a connection can be half-open depending on whether it is on the client-side or on the server-side. This will affect the connection rate on the server-side and client-side.

The system calculates the Client Connection Rate and the Server Connection Rate based on the following information:

- The client connection is established after the client SYN and the server SYN-ACK.
- The server connection is established after the client ACK.
- The data is sent after a connection has been established both on the client-side and the server-side.
- The server-side connection is closed after the client FIN-ACK and the server FIN-ACK.
- The client-side connection is closed after the client ACK.

You can see that the client connection is opened before the server connection, and it is closed after the server connection has closed.

Session Sender should be able to open a maximum of 5,000,000 (1Gb) or 20,000,000 (10Gb) simultaneous sessions; however, after reviewing the test results, it looks like the component never reaches 5,000,000 (1Gb) or 20,000,000 (10Gb) total sessions. Why is this?

If the test only has one Session Sender component, enough time must be allotted to the ramp up duration for the component to open 5,000,000 (1Gb) or 20,000,000(10Gb) sessions. By using the one Session Sender component to open 5,000,000 (1Gb) 20,000,000 (10Gb) sessions, it will take the component longer to reach the maximum number of sessions than if multiple components had been used.

Since Session Sender can open a maximum of 500,000 (1Gb) or 750,000 (10Gb) sessions per second, the ramp up duration must be set to at least 20 seconds for the component to open a total of 5,000,000 (1Gb) or 20,000,000 (10Gb) sessions.

To calculate the ramp up duration, the following equation was used: 5,000,000(1Gb) or 20,000,000 (10Gb) sessions / 500,000 (1Gb) or 750,000 (10Gb) sessions per second = 20 seconds.

SSL Questions

How do I test SSL/TLS?

You test SSL with either the Application Simulator or Client Simulator component.

How do I set up an SSL/TLS test?

Add an Accept TLS and/or Start TLS action to any TCP flow in the Super Flow editor.

What versions of SSL/TLS are supported?

Application Simulator and Client Simulator support SSLv3 and TLS. TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.

Are CA certificate chains supported?

No. Only one certificate per TCP Super Flow is supported.

What cipher suites are supported?

See Supported Cipher Suites.

Are proxy certificates supported?

Yes, proxy certificates are currently supported.

How many SSL handshakes per second are supported?

43,000 SSL handshakes per second are supported.

How many transactions occur per session?

You can edit Super Flows to create your own transactions.

Why does the payload size I see on the wire not match the MSS I have specified when SSL is enabled?

When SSL is enabled, the segment size is actually treated as the size of the payload before it is encrypted. Encrypting it makes it a few bytes larger; the data is padded to a multiple of 8 or 16 bytes, depending on the encryption algorithm negotiated.

Super Flow Questions

I've imported a response file that is 200 MB; however, I've noticed that the system doesn't use the entire contents of the file. Why is this happening?

The system will only use the first 100 MB of data; therefore, if the file is larger than 100 MB, the system will not use the entire contents of the file.

System Questions

What are the power requirements for the BreakingPoint Storm?

The following are the power requirements for the BreakingPoint Storm: 100-240 VAC, 4 A at 50-60 Hz, and a maximum power consumption of 400 Watts.

What are the power requirements for the BreakingPoint FireStorm?

The following are the power requirements for the BreakingPoint FireStorm: 200-240 VAC, 8.5 A at 50-60 Hz, and a maximum power consumption of 1,800 Watts.

What is the manufacturer MAC address for the BPS Management port?

00:1A:C5

Does the system support ephemeral ports or application specification modifications that are required to match the application data to the IP and TCP/UDP headers?

No. This functionality is currently not supported.

What is the maximum current available for the DUT power receptacle?

120 VAC or 8.3A

Can multiple users use the system?

Yes. Multiple users can be logged into the system at the same time and multiple tests, Tcl scripts, and packet captures can be run simultaneously.

How do I restore factory settings for the BreakingPoint Storm?

You can restore your system to the factory settings manually from the physical box.

First, power cycle the box. Once the Status LED is green, you will need to insert a pin-sized object into the Reset button. Continue to push the Reset button until the Status LED turns red. The Status LED will then turn red and alternatively flash red and green. After this process is done, the chassis will reboot. You will need to configure the chassis after it finishes rebooting.

What is the difference between a factory revert and a previous revert?

A factory revert will roll the system back to the build that was initially installed on it (i.e., the factory build) and revert it back to its factory state; therefore, all settings, tests, and data stored on the system will be removed. A previous revert will roll the system back to the build that was previously installed on your system.

When would I use the Preload for slower connections button on the Login Page?

Use the Preload for slower connections button if your connection is slow. Pressing this button prefetches the application assets and places them into the browser's cache. This reduces the amount of time it takes for the application to load. When you clear your browser's cache, press the Preload for slower connections button again on subsequent logins.

What is the difference between a soft reboot and a restart?

A soft reboot will restart the software processes, whereas restart will power-cycle the box.

How do I know when a firmware update or ATI Update is available?

Check the Ixia Support website periodically for new releases. If your PC is connected to the internet, you may be able to receive an ATI Cloud Update.

What ports need to be open to allow me to manage the system?

You will need to have the following TCP ports available: 80, 443, 8880, and 843.

My system status says "System Not Operational". What should I do?

There are two cases when this may occur: soon after a system has been rebooted, or after the system has not been rebooted for an extended period of time. Typically, after you reboot your system, you should wait at least 5 minutes before running a test. If you try to run a test before this time, the system may display this error. To resolve this error in either case, reboot your system, and wait at least 5 minutes before using the system.

Where is the diagnostics file?

You can download the diagnostics file from the BreakingPoint system Start Page. If you click the **Diagnostics** button, you will be prompted to save a ZIP file to your computer. The zip file contains the diagnostics files for the system.

Why does it take so long for my browser to finish loading information?

Occasionally, there may be some lag between loading different screens in the Control Center. We recommend viewing the Control Center with at least 2 Mb of RAM.

Tcl Questions

I want to run tests that I've created from the Control Center through the Tcl interface. How do I do this?

You can run tests created from the Control Center through the Tcl interface by using the **run** command. For example, you can use the following syntax to run a test you have created and given the name appsimTest1:

```
set bps [bps::connect 10.10.11.219 admin
admin]
set t [$bps createTest-template appsimTest1]
$t run
```

Do I have to use BreakingPoint Systems' Tcl shell?

Our Tcl shells are Standalone Runtime Kits (Starkit), which allow you to wrap an application around it and make it completely self-contained. You can unwrap a Starkit using SDX, which you can get at http://www.equi4.com/starkit/sdx.html.

Do I need to download a new Tcl shell each time I update the system?

Yes.

Do you support Tcl 8.5?

Yes.

How can I determine if there are tests already running on the system?

You can use the chassis object to determine the status of the system. For example, you can use the following syntax::

```
set bps [bps::connect 10.10.10.10 admin admin -onclose
exit]
set chassis [$bps getChassis]
proc echo {args} {
puts $args
}
$chassis configure -ontestprogress {echo "test is running:
"}
set slot 1
set port 0
$chassis reservePort $slot $port
```

How do I get a list of available commands from the Tcl interface?

Call the variable you used to store the `bps::connect`. For example, if you stored the connection in the variable bps, you will need to enter $bps to get a list of available commands.

I get the error "BPS device is version xxxx, while this shell is version xxxx. Please download a new version of the shell from the device". How do I get the new version of the Tcl shell?

This error will display after you have upgraded your system to a newer OS version, but have not downloaded the latest Tcl shell. You will need to go to the system's Start Page and click the **Download Tcl Shell** link to download the newest shell.

When the system boots up, it notifies that a slot is down, but does not indicate which port. Why does it not specify the port number?

The state change is on the slot itself, not on an individual port.

How do I get a list of the parameters and values for a component?

You can use the **configure** command to return a list of parameters and values for a component. For example, if you have a Routing Robot component called RR1, you can use the following syntax to return the values and parameters for the component:

```
RR1 configure; returns a list of parameters and values for Routing
Robot
```

Test Questions

Can I edit a BreakingPoint Systems canned test?

Yes. You can edit a canned test; however, you must save the modified version as a new test.

How many tests can I concurrently run?

The number of tests that you can run concurrently depends on the number of ports you have on your BreakingPoint system. For example, if your BreakingPoint system has 8 ports, you can run 8 tests at a time; if you have 16 ports, then 16 tests can run simultaneously.

How do I run a test without saving the changes I have made to the test?

You can run a test without saving your changes by select **Test > Run** from the Menu bar. However, after you run the test, clicking the **Edit** button on the Real-Time Statistics window will take you back to the saved version of the test. Any changes that you made prior to running the test will be restored to their saved settings.

How can I delay the start of a test component?

Each component has a parameter called Delay Start that enables you to delay the start of a test component by the time specified. This parameter is measured in seconds and supports floating values.

I am trying to run a test, but the run functionality is disabled. Why is this happening?

The ports you are trying to use are in use by another user. You may want to remap your ports on the Device Status page, or wait until the user has finished using the ports. This can also occur if you do not have any ports reserved. Functionality can also become disabled if the test you are attempting to run is invalid due to oversubscribing (for example, if you are attempting to run a 10 Gb test on a 1 Gb blade).

Why is my test not running for the duration (test duration) that was configured ?

Based on the configured lower/upper and step % of load, BreakingPoint calculates how many iterations need to be run based on frame size. For example, lower/upper and step % of load for a test can be expressed as: X. Based on the calculated frame size distribution, BreakingPoint calculates how many frames the test will send (this value=Y). Then total time (T), is divided by X*Y, that is every iteration of the test will run for T/(X*Y) seconds to reach the test duration.In some scenarios, based on the frame size, a test may fail to achieve the objective and the rate for an iteration is not implemented. The time that would have been required to achieve non-implemented rates are deducted from the test total duration time.

Test Component Questions

Can components be run at the same time?

Yes. All test components can be executed with a single test. Tests can contain multiple occurrences of a test component, but bandwidth and hardware resources will affect the number and type of test components that can be added to a test.

How many occurrences of each test type of component does a test support?

Session Sender, Application Simulator, and Recreate support up to 8 components per test. Security and Stack Scrambler support 4 components per chassis. Bit Blaster and Routing Robot support up to 4 components per port.

Traffic Questions

How is packet size calculated?

Subtract 18 from the frame size. This will give you the byte size of the packet.

How is latency measured?

Session-based components only measure latency on TCP packets using the TCP timestamp field. The Bit Blaster and Routing Robot test components measure inter-packet delay, or the amount of time it takes from the last sent packet to the next arriving packet.

What is the latency resolution?

Latency calculations are accurate to +/- 1 millisecond.

How are packets validated?

The Bit Blaster and Routing Robot test components compare all fields in the header (except values that have been modified due to routing or NATing devices) and all bytes of the payload to identify what was sent and received.

Do you support SSL (HTTPS)?

Yes. Any TCP flow can be encrypted with SSL, not just HTTP.

Can I send and receive traffic on the same interface?

Yes. You can send and receive traffic on the same interface if the test component has a different VLAN-enabled domain assigned for the server and the client interfaces.

Update Questions

Note: The Release Notes that for the BPS version you want to upgrade to is a good resource for questions about the steps required before and after updating your BPS software.

I just installed the latest OS update; however, I could not reconnect. What should I do?

Clear the cache on your browser.

Where can I download the latest firmware updates and ATI updates?

All updates can be downloaded from the Ixia website at: https://support.ixiacom.com > **Software Downloads** > **BreakingPoint Software**. Note that you will need to log in to access this section of the support website.

How do I get a Ixia Support website account?

Contact our support team at support@ixiacom.com or 1-818-595-2599.

How are the OS update files for BPS hardware devices named?

Update files use the format X-N.bps. The X refers to the oldest version you can upgrade from, and the N refers to the update file's version.

Will ATI Updates update my existing Strike List with the latest Strikes?

All ATI Updates will populate Smart Strike Lists with current strikes. Standard Strike Lists must be manually updated after applying any ATI upgrade.

CHAPTER 3 BreakingPoint Virtual Edition Feature Support

The tables in this section describe the feature support for BreakingPoint Virtual Edition and BreakingPoint for Amazon Web Services.

Note: See the *BreakingPoint Virtual Edition Installation Guide* for installation information. The guide is available on the Ixia Support website.

Network Neighborhood	BPS VE	BPS on AWS
IPv4/IPv6 Static Hosts	✓	✓
IPv4/IPv6 External Hosts	✓	✓
NAT	✓	NS
VLAN	✓	NS
IPv4/IPv6 Router	✓	✓
DHCPv4 (client/server)	✓	NS
DHCPv6 (client/server)	NS	NS
IPv4 DNS	✓	✓
IPv6 DNS	✓	✓
IPsec IKEv1/IKEv2	NS	NS
LTE(IPv4)	✓	NS
LTE(IPv6)	NS	NS
3G	NS	NS
6RD	NS	NS
DSLite	NS	NS
IPv6 SLAAC	NS	NS

Test Components	BPS VE	BPS on AWS
Live Application Simulator	✓	NS
Application Simulator	✓	✓
Client Simulation	✓	✓
Security	✓	✓ *1
Malware	✓	✓ *1
Session Sender	✓	✓
Stack Scrambler	✓	✓ *2
SSL/TLS	✓	✓
Packet Capture	✓	✓
Impairment	NS	NS
Bit Blaster	✓	NS
Routing Robot	✓	✓
Recreate	✓	✓ *3
SCTP	✓	✓

*1- Some attacks may get blocked by AWS.

*2 - Some invalid IP packet patterns are not compatible with AWS (traffic might get dropped by AWS).

*3 - Limited support. This is because Replay Capture File Without Modification mode replays libpcap formatted capture files without modifying Layer 2 through Layer 7 and AWS requires BPS to use the MAC address that corresponds to the interface that is sending the packets.

BreakingPoint Labs	BPS VE	BPS on AWS
Session Sender Lab	✓	NS
RFC 2544 Lab	✓	NS
Multicast Lab	✓	NS
Lawful Intercept Lab	✓	NS
Device Validation Lab	NS	NS

BreakingPoint Labs	BPS VE	BPS on AWS
Multibox Testing	NS	NS
Resiliency Score	NS	NS
Data Center Resiliency	NS	NS
LTE Lab	NS	NS
DDoS Lab	✓	NS

This page intentionally left blank.

CHAPTER 4 Product Overview

This section provides an overview of the BreakingPoint Products.

BreakingPoint Device Overview

BreakingPoint devices measure and harden the resiliency of every component of your critical infrastructure against potentially crippling attacks and peak application traffic. These devices are modular systems that can accurately recreate a live network environment.

BreakingPoint devices consist of the chassis (physical or virtual) and the user interface called the Control Center. Both components work together to create a comprehensive and user-friendly solution for all network devices. BreakingPoint devices can concurrently simulate TCP sessions, UDP sessions, application traffic, and live security attacks, and ultimately, identify "breaking points" in your network devices.

BreakingPoint Virtual Edition Overview

Ixia's BreakingPoint Virtual Edition is a Virtualized Application and Security Resilience Test Solution. It provides scalable real-world application and threat simulation in an elastic deployment model by leveraging virtualization and industry-standard hardware platforms.

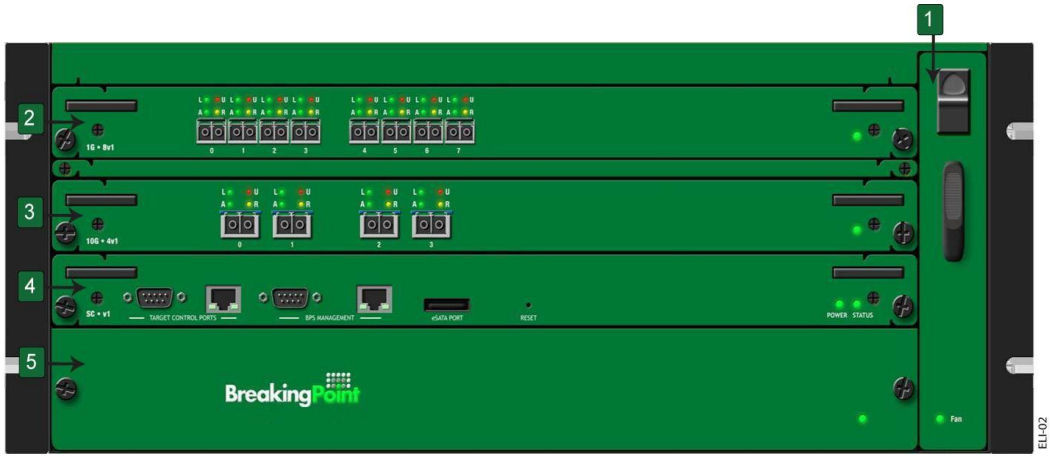
This user guide describes how to use the features that are available in the user interface after logging in. The *BreakingPoint Virtual Edition Installation Guide* explains how to install, run, and manage a BreakingPoint Virtual Chassis. The BreakingPoint Virtual Edition Installation Guide is available on the Ixia Support website.

See BreakingPoint Virtual Edition Feature Support on page 25 for detailed information about the BPS features that are supported when using BreakingPoint Virtual Edition.

BreakingPoint Storm Slot Descriptions

Note: For detailed information on the BreakingPoint Storm installation and setup, see the *BreakingPoint Storm Installation Guide* on the Ixia Support website.

The BreakingPoint Storm is comprised of five slots. The figure below highlights these slots with callouts.



Callout 1 refers to the removable fan tray that is vertically mounted on the right-side of the chassis.

Callout 2 and **Callout 3** refer to the slots dedicated to high-speed data plane processors, or the blades, for the system. When you initially receive the BreakingPoint Storm, these slots will not contain any blades, so you will need to install the blade(s) into the chassis.

Each blade provides fiber-optic data ports (plus four copper on 1Gb systems) that support up to 10 Gbps for 10 Gb blades and 1 Gbps for 1 Gb blades. The fiber-optic connections between the ports on your device under test to the test ports on the chassis establish the transmitting and receiving interfaces for your tests.

Callout 4 points to the system controller, which holds the BPS management ports, target control ports, a reset button, and an eSATA port.

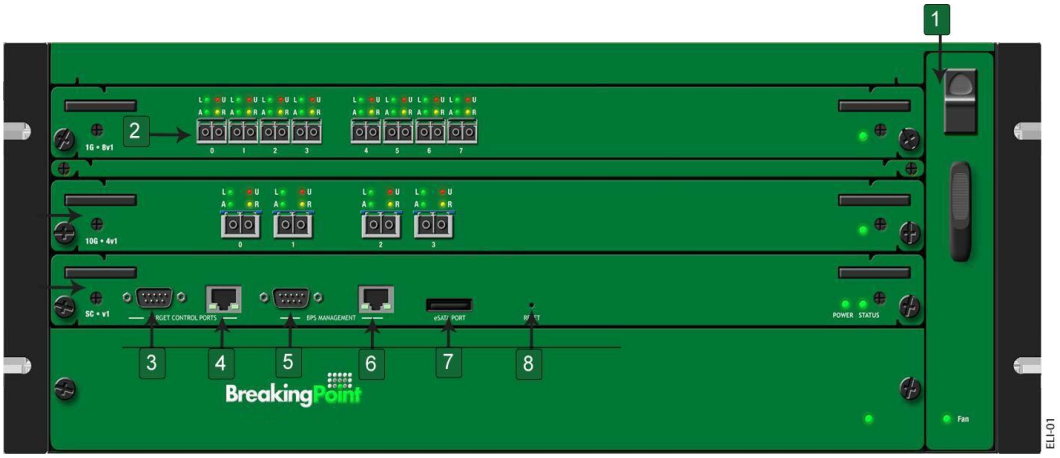
The BPS management ports (serial and Ethernet), located on the system controller, allow you to connect your system to a network and access it through an IP address; the target control ports allow you to automate testing for the device under test; and the reset button restores your system to the default factory settings.

> **Note:** Each BreakingPoint FireStorm blade contains an onboard system controller. When you install a BreakingPoint FireStorm into the chassis, the onboard system controller replaces the dedicated system controller.

Callout 5 refers to the power tray, which contains the power supply for the system.

BreakingPoint Storm Front Panel

The figure below illustrates the front panel of the BreakingPoint Storm. Locate the corresponding callout in the table below for more information about each component.



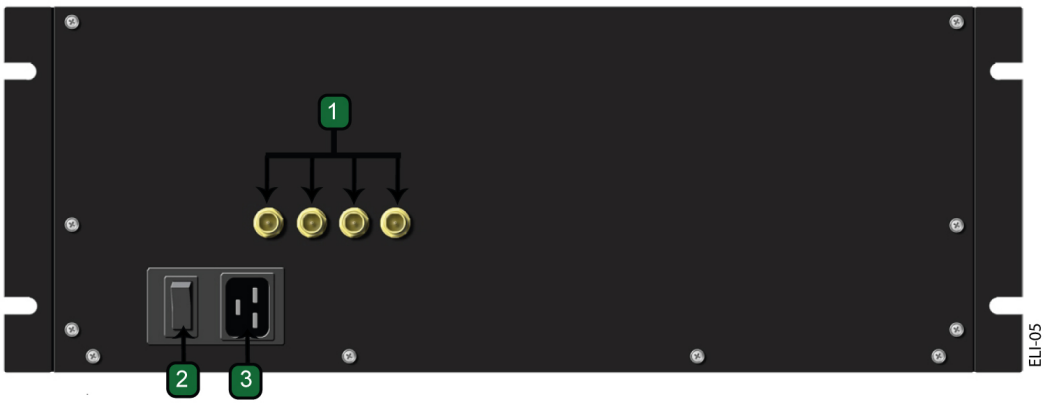
The table below lists the components on the BreakingPoint Storm front panel.

BreakingPoint Storm Front Panel Components

Callout	Component	Description
1	System Fan Tray	Holds the fan tray for the system.
2	Data Ports	Transmits and receives traffic to and from the DUT.
3	Target Control Serial Port	Used to manage and configure settings for the DUT.
4	Target Control Ethernet Port	Used to manage and configure settings for the DUT.
5	BPS Management Serial Port	Manages the BreakingPoint Storm configuration through a serial connection.
6	BPS Management Ethernet Port	Manages the BreakingPoint Storm configuration through an Ethernet connection.
7	eSATA Port	Provides an eSATA connection for an external memory device. This is currently disabled for the 1.3 Release.
8	Reset button	Restores the system to factory settings; this process is irreversible and all tests, imported data, and configurations will be permanently removed from the system.

BreakingPoint Storm Back Panel

The power inlet and power switch are located on the back panel of the chassis, as shown in the figure below. Additionally, there are BNC interfaces that you will be able to use in future releases to link together multiple chassis.



The table below lists the components on the BreakingPoint Storm back panel.

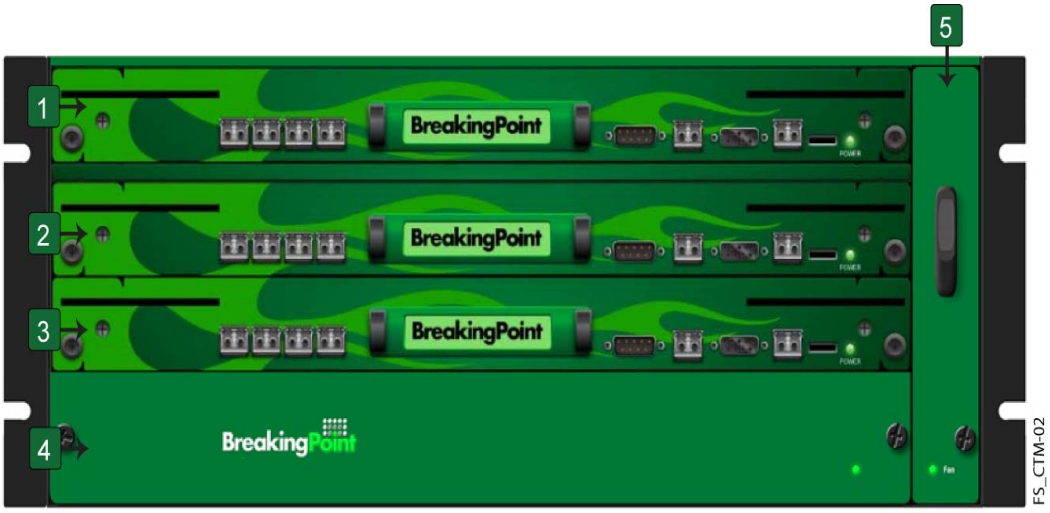
BreakingPoint Storm Back Panel Components

Callout	Component	Description
1	BNC Interfaces	Interfaces that are used to connect multiple chassis together (for clock I/O and trigger I/O)
2	Power Switch	Power breaker switch for the BreakingPoint Storm
3	Power Inlet	Power inlet for the BreakingPoint Storm

BreakingPoint FireStorm Slot Descriptions

Note: For detailed information on BreakingPoint FireStorm installation and setup, see the *BreakingPoint FireStorm Installation Guide* on the Ixia Support website.

The BreakingPoint FireStorm is comprised of five slots. The figure below highlights these slots with callouts.



Callouts 1, 2 and **3** refer to the slots dedicated to high-speed data plane processors (or the blades) for the system. When you initially receive the BreakingPoint FireStorm, these slots will not contain any blades, so you will need to install the blade(s) into the chassis.

Each blade provides fiber-optic and four copper data ports that support up to 10 1 Gbps for 10 Gb blades. The fiber-optic connections between the ports on your device under test and the test ports on the chassis establish the transmitting and receiving interfaces for your tests.

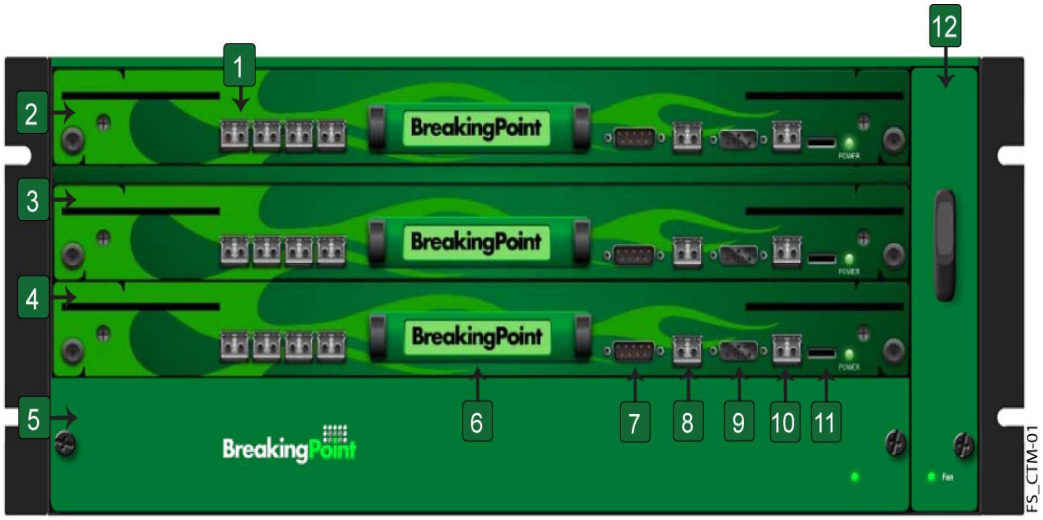
The BPS management ports (serial and Ethernet) allow you to connect your system to a network and access it through an IP address. The target control ports allow you to automate testing for the device under test.

Callout 4 refers to the power tray, which contains the power supply for the system.

Callout 5 refers to the removable fan tray that is vertically mounted on the right-side of the chassis.

BreakingPoint FireStorm Front Panel

The figure below illustrates the front panel of the BreakingPoint FireStorm. Locate the corresponding callout in the table below for more information about each component.



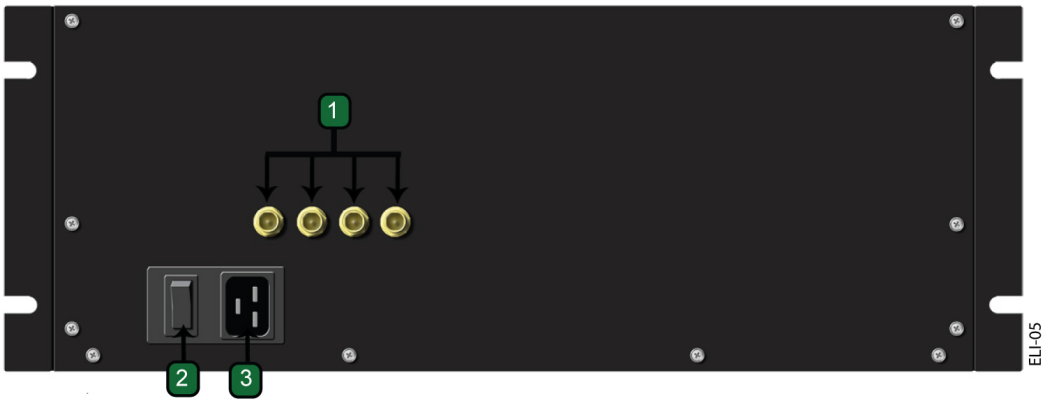
The table below lists the components on the BreakingPoint FireStorm front panel.

Callout	Component	Description
1	Data Ports	Transmits and receives traffic to and from the DUT.
2	System Blade (Slot 2)	Dedicated to high-speed data plane processors (blades).
3	System Blade (Slot 1)	Dedicated to high-speed data plane processors (blades).
4	System Blade (Slot 0)	Dedicated to high-speed data plane processors (blades).
5	Power Tray	Contains the power supply for the system.
6	Hard drive tray	Contains the hard drive enclosure. A flashing power tray indicates that the BreakingPoint FireStorm blade in slot 1 or slot 2 is connected to an improper power supply configuration. When connecting power to your BreakingPoint FireStorm, be sure to use only the power supply that was shipped with your BreakingPoint FireStorm blade.
7	Target Control Serial Port	Used to manage and configure settings for the DUT.

Callout	Component	Description
8	Target Control Ethernet Port	Used to manage and configure settings for the DUT.
9	BPS Management Serial Port	Manages the BreakingPoint FireStorm configuration through a serial connection.
10	BPS Management Ethernet Port	Manages the BreakingPoint FireStorm configuration through an Ethernet connection.
11	USB Port	Provides a USB connection for an external memory device.
12	System Fan Tray	Holds the fan tray for the system.

BreakingPoint FireStorm Back Panel

The power inlet and power switch are located on the back panel of the chassis, as shown in . Additionally, there are BNC interfaces that you will be able to use in future releases to link together multiple chassis.



The table below lists the components on the BreakingPoint FireStorm back panel.

Callout	Component	Description
1	BNC Interfaces	Interfaces that are used to connect multiple chassis together (for clock I/O and trigger I/O)

Callout	Component	Description
2	Power Switch	Power breaker switch for the BreakingPoint FireStorm
3	Power Inlet	Power inlet for the BreakingPoint FireStorm

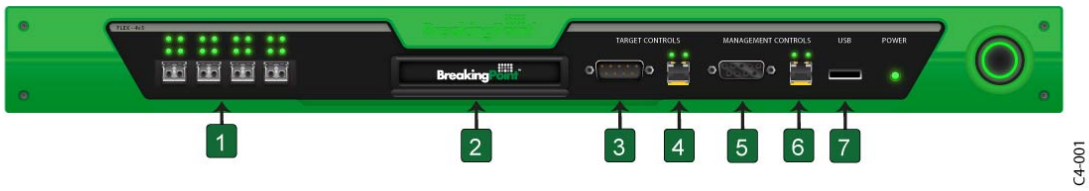
BreakingPoint FireStorm ONE Overview

The BreakingPoint FireStorm ONE is a 1U, rack-mountable appliance. Each BreakingPoint FireStorm ONE provides fiber-optic and four copper data ports. The fiber-optic connections between the ports on your device under test and the test ports establish the transmitting and receiving interfaces for your tests.

The BPS management port (serial and Ethernet) allows you to connect your system to a network and access it through an IP address. The target control ports allow you to automate testing for the device under test.

Note: For detailed information on BreakingPoint FireStorm One installation and setup, see the *BreakingPoint FireStorm One Installation Guide* on the Ixia Support website.

The figure below illustrates the front of the BreakingPoint FireStorm ONE. Locate the corresponding callout in the table below for more information about each component.



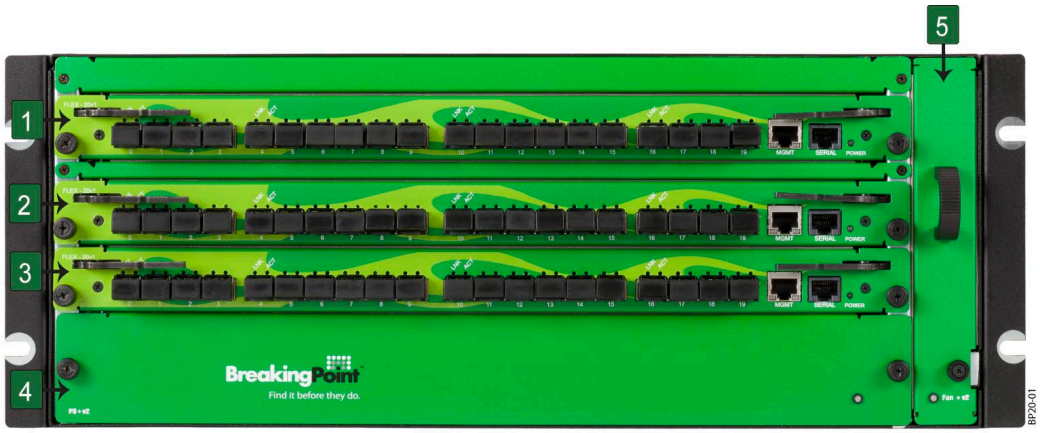
Callout	Component	Description
1	Data Ports	Transmits and receives traffic to and from the DUT.
2	Hard drive tray	Contains the hard drive enclosure. A flashing power tray indicates that the
3	Target Control Serial Port	Used to manage and configure settings for the DUT.
4	Target Control Ethernet Port	Used to manage and configure settings for the DUT.
5	BPS Management Serial Port	Manages the BreakingPoint FireStorm ONE configuration through a serial connection.

Callout	Component	Description
6	BPS Management Ethernet Port	Manages the BreakingPoint FireStorm ONE configuration through an Ethernet connection.
7	USB Port	Provides a USB connection for an external memory device.

BreakingPoint 20 Slot Descriptions

Note: For detailed information on the BreakingPoint 20 installation and setup, see the *BreakingPoint 20 Installation Guide* which is available on the Ixia Support website.

The BreakingPoint 20 is comprised of five slots as shown in the figure below. The figure below highlights these slots with callouts.



Callouts 1, 2 and **3** refer to the slots dedicated to high-speed data plane processors (or the blades) for the system. When you initially receive the BreakingPoint 20 chassis, these slots will not contain any blades. You will need to install the blade(s) into the chassis.

Each blade provides 20 universal 10GigE/1GigE SFP+ ports that support up to 40 Gbps per blade. The fiber-optic connections between the ports on your device under test and the test ports on the chassis establish the transmitting and receiving interfaces for your tests.

The BPS management ports (serial and Ethernet) allow you to connect your system to a network and access it through an IP address.

Callout 4 refers to the power tray, which contains the power supply for the system.

Callout 5 refers to the removable fan tray that is vertically mounted on the right-side of the chassis.

BreakingPoint 20 Front Panel

The figure below illustrates the front panel of the BreakingPoint 20.

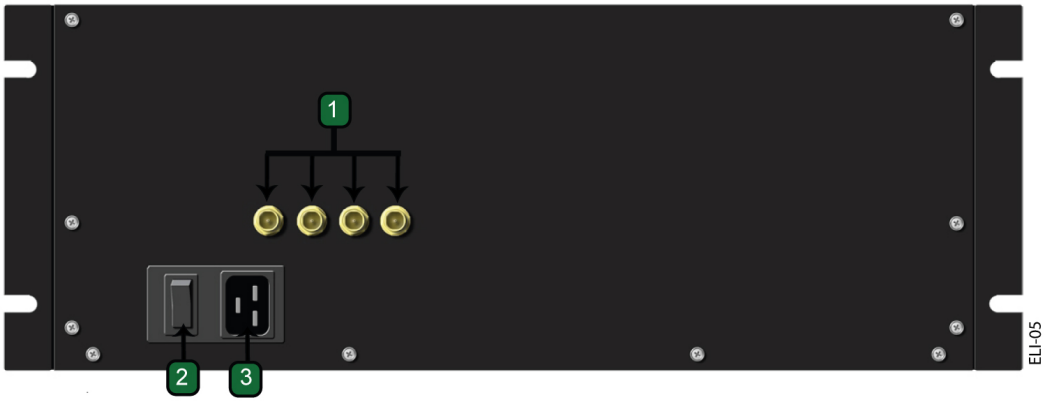
The table below lists the components on the BreakingPoint 20 front panel.

BreakingPoint 20 Front Panel Components

Callout	Compnent	Description
1	Data Ports	Transmits and receives traffic to and from the DUT.
2	BPS Management Ethernet Port	Manages the BreakingPoint 20 configuration through an Ethernet connection.
3	BPS Management Serial Port	Manages the BreakingPoint 20 configuration through a serial connection.
4	System Fan Tray	Holds the fan tray for the system.

BreakingPoint 20 Back Panel

The power inlet and power switch are located on the back of the chassis, as shown in the figure below. Additionally, there are BNC interfaces that you will be able to use in future releases to link together multiple chassis.



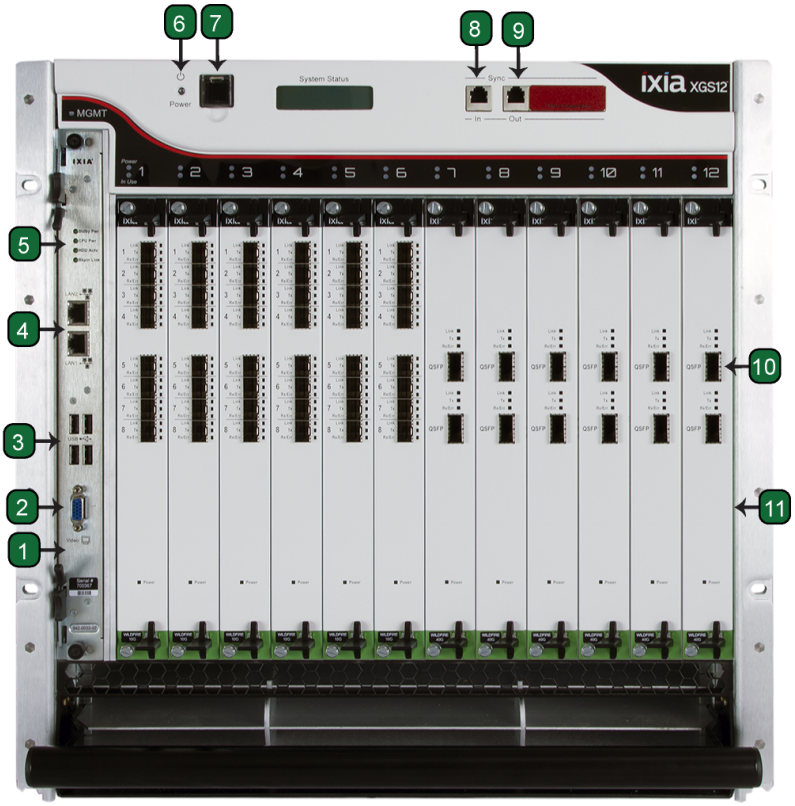
The following table lists the components on the BreakingPoint 20 back panel.

Callout	Component	Description
1	BNC Interfaces	Interfaces that are used to connect multiple chassis together (for clock I/O and trigger I/O)
2	Power Switch	Power breaker switch for the BreakingPoint 20
3	Power Inlet	Power inlet for the BreakingPoint 20

Ixia PerfectStorm Fusion Front Panel Overview

Note: For detailed information on Ixia PerfectStorm Fusion installation and setup, see the *XGS12 Chassis Platform Assembly Guide* that is available on the Ixia Support site.

The figure below illustrates the front panel of the PerfectStorm Fusion. Locate the corresponding callout in the table below for more information about each component.



The following table lists the components on the Ixia PerfectStorm Fusion front panel.

Callout	Component	Description
1	Integrated Controller Module	Provides management for the chassis and the Web-based user interface.
2	Video Port	HD-DB15 Super VGA.
3	USB Ports	4 USB dual type A, 4-pin jack connectors provide USB connections for external memory devices.
4	BPS Management Ethernet Ports	RJ-45 10/100/1000Mbps Gigabit Ethernet Management Ports manage the Ixia PerfectStorm Fusion configuration through an Ethernet connection.
5	Front Panel LEDs	Standby Power – Indicates whether the 5V of Standby power is available. CPU Power – Indicates whether the CPU card power is available. HDD Activity – Indicates activity by internal hard drive. Backplane Link – Indicates whether the PCIe link to the backplane is up.

Callout	Component	Description
6	Power LED	When the Power LED is flashing, the board is being detected or initialized.
		The Power LED is illuminated when the board is powered.
7	Power Switch	On/Off momentary power push button.
8	Sync In	Single Sync In jack with a 4-pin RJ11.
9	Sync Out	Single Sync Out jack with a 4-pin RJ11.
10	Data Ports	Transmits and receives traffic to and from the DUT.
11	Slot	Dedicated to high-speed load modules (blades).

Ixia PerfectStorm Fusion Back Panel Overview

The power inlet and power switch are located on the back of the chassis, as shown in the figure below.

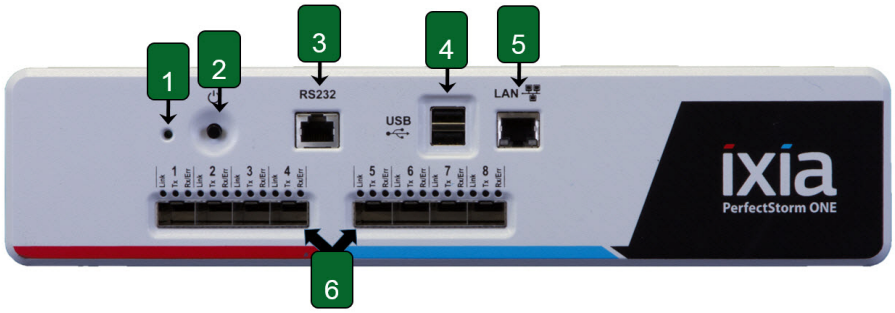
The following table lists the components of the Ixia PerfectStorm Fusion back panel.

Callout	Component	Description
1	Fan Module	Holds the fan module for the system.
2	Rear Cover	Covers the power supply module for the system.

Ixia PerfectStorm One Front Panel Overview

The PerfectStorm ONE appliance consists of a single-slot chassis integrated with a load module. The appliance provides the platform to seamlessly unify the IxLoad and BreakingPoint software applications into a single and more powerful system, and at the same time provides a portable solution in a compact form factor appliance.

Note: For detailed information on Ixia PerfectStorm One installation and setup, see the Ixia PerfectStorm One Getting Starting Guide on the Ixia Support website.

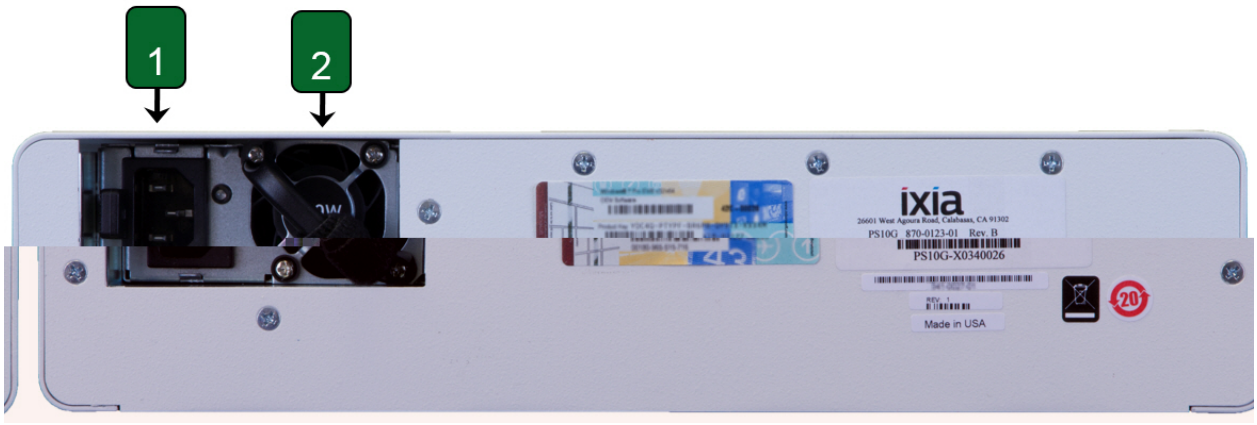


The following table lists the components on the Ixia PerfectStorm One front panel.

Callout	Component	Description
1	Power LED	Unit power indicator.
2	Power Button	Power on/off button.
3	Serial Port	RS-232 serial port which is used to access the Administrative Console.
4	USB Ports	Used to perform FlixOS upgrade.
5	Management Network Port	1G Ethernet management port.
6	Application Traffic Network Ports	40G, 10G or 1G Ethernet traffic ports depending on licensing.

Ixia PerfectStorm One Back Panel Overview

The power inlet and fan are located on the back of the chassis, as shown in the figure below.



The table below lists the components of the Ixia PerfectStorm One back panel.

Callout	Component	Description
1	Power Inlet	Power Inlet for the chassis.
2	Fan	Fan for the chassis.

Control Center Overview

The Control Center is a Web-based user interface where the testing environment can be created, tests can be run, and reports can be viewed. The Control Center is accessible through an Adobe Flash-enabled Web browser. Mac users with OS 10.8.2 can use Firefox or Google Chrome as their browser.

Note:

- The table below lists the browsers supported by BreakingPoint. Browser versions that are more current than the versions listed in the table may work, but have not been tested at this time. Beta versions of HTML browsers are not supported.
- Note that Chrome is supported on Windows and MAC operating systems.
- In BPS 3.5.1, and higher releases, logging in to the management UI will not be allowed from HTML browsers that have a maximum SSL version setting that is limited to only sslv3 (ssl3).

Supported Browsers

Browser	Recommendation for Windows	Recommendation for MAC
Google Chrome	68.0.3440.75 (32-bit)	67.0.3396.99(64-bit)

Browser	Recommendation for Windows	Recommendation for MAC
Firefox	61.0.1 (64-bit)	61.0.1 (64-bit)
Microsoft Edge	38.14393.2068.0	N/A
Safari	Not supported	11.1.2
Internet Explorer	Not supported	Not supported

Enabling JavaScript

You must have JavaScript enabled to view the Control Center. Consult your browser documentation for information on how to enable JavaScript.

Browser Resources

Please note that if you have several browser windows open simultaneously, or if you have multiple instances of the Control Center open, this may cause lagging or delayed responses from the system. This is normal behavior for the Control Center if multiple browser resources are being used.

Note: BreakingPoint recommends clearing your cache and refreshing your browser after upgrading or reverting to any release of the BreakingPoint.

You must have popups and JavaScript enabled to view the Control Center.

Navigational Overview

This section provides an overview of the navigational areas in the Control Center. The Control Center is divided into two main areas: the menu bar and the navigational buttons. See the figure and table below for a tour of the interface.

Control Center Overview

Control Center Overview

Callout	Name	Description
1	Menu Bar	Provides point and click access to the main areas of the user interface.
2	Device Status Icon	Provides access to the Device Status area so that you can reserve ports while no tests are running or access to the Real-Time Statistics window if there is a running test.
3	Navigational Buttons	Provides access to areas within the user interface.
4	Firmware Version Information	Provides firmware version and update information.

Features Overview

BreakingPoint Storm Features below lists the features available with the BreakingPoint System.

BreakingPoint Storm Features

Feature	Description
Application Profile	Create a container for a set of Super Flows that Application Simulator will use to generate test traffic.

Feature	Description
Super Flows	Create and customize Super Flows that define the characteristics of the application traffic.
DUT Profile	Create custom profiles that contain the connection settings and the interface speeds for a device under test.
E-mail Test Results	Send completed test results to your e-mail account.
Lawful Intercept Test Lab	Detect and capture specific information flows out of a large field of untargeted flows. Enables you to easily construct a scenario with configurable traffic (with both random and specific keywords) and real-world background traffic.
Long Term Evolution (LTE) Test Lab	Allows you to test your LTE devices by emulating a mobile telecommunications environment complete with mobile phone users of various types, connecting cell towers, and a variety of services.
Multicast Test Lab	Emulate multicast clients and servers for performance testing of external clients, servers, and routers.
Multi-box Testing	Use a single management interface to control multiple boxes and share one IP source address pool across multiple boxes.
Network Neighborhood	Set up network addressing for test traffic.
Quick Tests	Run Quick Tests to get an instant snapshot of how well your device performs based on standard industry testing metrics.
Real-Time Stats	Get instant feedback on the test progress with interactive and live graphs.
Recreate Traffic	Capture live network traffic and replay traffic to the device under test.
Reports	Export test results in PDF, HTML, RTF, XLS, CSV, and ZIP (CSV).
Resiliency Score Test Lab	Consists of a set of standardized tests for measuring the resiliency of your network devices, allowing you to determine their true level of security, performance, and stability.
SSL Support	Encrypt and/or decrypt any TCP traffic with SSL/TLS.
Strike List	Access thousands of Strikes and dozens of Strike options from which you can create custom Strike Lists.

Feature	Description
Strike List Import and Export	Import and export Strike List from one system to another.
ATI Updates	Instantly obtain the latest ATI Updates from the Ixia Support website.
System Updates	Download OS updates from the Ixia Support website.
Tcl Interface	Use the Windows, Linux or Mac OS Tcl shell to automate device testing with Tcl scripting.
Test Components	Use these customizable virtual devices to simulate Layer 2-7 traffic.
Test Pass/Fail Criteria	Define custom pass/fail test criteria.
Test Import and Export	Import tests or export tests and share them with other systems.
VLAN Tagging	Tag network traffic with single or double VLAN tags.

Switching From IxLoad Mode to BreakingPoint Mode

All PerfectStorm Fusion load modules (blades) are capable of operating in IxLoad or BreakingPoint mode. When booting up, all PerfectStorm Fusion load modules default to IxLoad mode. A red square in the upper right corner of the load module on the Device Status screen indicates that the module is in IxLoad mode. A green square indicates that the module is in BreakingPoint mode.

To transition multiple load modules to BreakingPoint mode, each load module must be allowed to completely transition before the process for the next load module can begin. Transitioning multiple modules simultaneously is prohibited.

Note: Load modules retain the mode they were in prior to being rebooted.

To transition from IxLoad mode to BreakingPoint mode:

1. Click a port on the load module to be transitioned. A message asking if you want to switch to BreakingPoint mode will be displayed.
2. Click Yes on the message that is displayed.

Note: The following graphics will be displayed during the transition process. As these graphics are displayed, no user action is required. Please wait until the square located in the upper right corner of the load module is green before continuing.

Graphics Displayed During Transition

Wait until a green square is displayed in the upper right corner of the load module. Once the green square is displayed, you can begin using the load module or you can begin the transition process for another load module.

The transition from IxLoad mode to BreakingPoint mode takes approximately five minutes for each load module.

This page intentionally left blank.

CHAPTER 5 Getting Started

Welcome to the Getting Started section of this guide. This section provides an overview of the tasks you must complete to set up your test environment within the BreakingPoint Control Center.

Task 1: Establishing a BreakingPoint Session

The Breaking Point (BPS) application has been integrated into Ixia's Web App. This integration enables you to access the BreakingPoint Control Center from the Ixia Web App user interface. You can also configure certain BreakingPoint chassis properties through the Ixia Web App user interface. New user accounts can now be created and managed through the Ixia Web App user interface as well.

In order to establish a BreakingPoint session and operate in the BreakingPoint Control Center, you must first log into the Ixia Web App.

Note:

- By default, BPS uses local authentication for user access. There is also the option of enabling TACACS+ Authentication on page 89 for user access.
- A session is an individual instance of a running test. You can run multiple sessions at one time, and manage all your current sessions from a single window. You manage sessions on the Sessions page, which displays after you login.

To establish a BreakingPoint session:

1. Open a web browser.
2. In the URL field, type the IP address or hostname of the Ixia chassis where the Ixia Web App server components are installed, followed by the port number that the Ixia Web App server is listening on (the default is 8080), and then press Enter.
   a. For example: 192.168.100.56:8080
   b. The Login page is displayed.
3. In the **Username** field, type your user ID.
4. In the **Password** field, type your password.
5. If you want the browser to automatically fill in the Username and Password field for future logins, check the **Remember Me** box.
6. Click **Login**.
   The Ixia Web Platform Dashboard displays.
7. Select the **BreakingPoint** Application icon.
8. Select option "a" shown below:
   a. **BreakingPoint New Session** - Start a new session to configure or run tests.
   b. **Administration Launch** - Provides access to system updates, backup/restore functionality and other operations. For earlier versions of BPS, select **Administration** from the menu and top left section of the page.
   c. **Results Launch** - Provides access to the list of previous test runs. You can view the detailed report for each run and also export it.

You are now in the BreakingPoint Control Center which is described in the Task 2: BreakingPoint Control Center Overview below.

Task 2: BreakingPoint Control Center Overview

Follow the steps described in Task 1: Establishing a BreakingPoint Session on the previous pageto access the BreakingPoint Control Center.

The BreakingPoint Control Center is a web-based user interface where the testing environment can be created, tests can be run, and reports can be viewed. The BreakingPoint Control Center is accessible through an Adobe Flash-enabled Web browser.

Note:

- Please note that if you have several browser windows open simultaneously, or if you have multiple instances of the BreakingPoint Control Center open, this may cause lagging or delayed responses from the system. This is normal behavior for the BreakingPoint Control Center if multiple browser resources are being used.
- BreakingPoint recommends clearing your cache and refreshing your browser after upgrading or reverting to any release of the BreakingPoint.

Navigational Overview

This section provides an overview of the navigational areas in the BreakingPoint Control Center. The BreakingPoint Control Center is divided into two main areas: the BreakingPoint Control Center Menu bar and the navigational buttons as shown in the image below.

BreakingPoint Control Center Overview

The table below lists the elements of the BreakingPoint Control Center.

BreakingPoint Control Center Overview

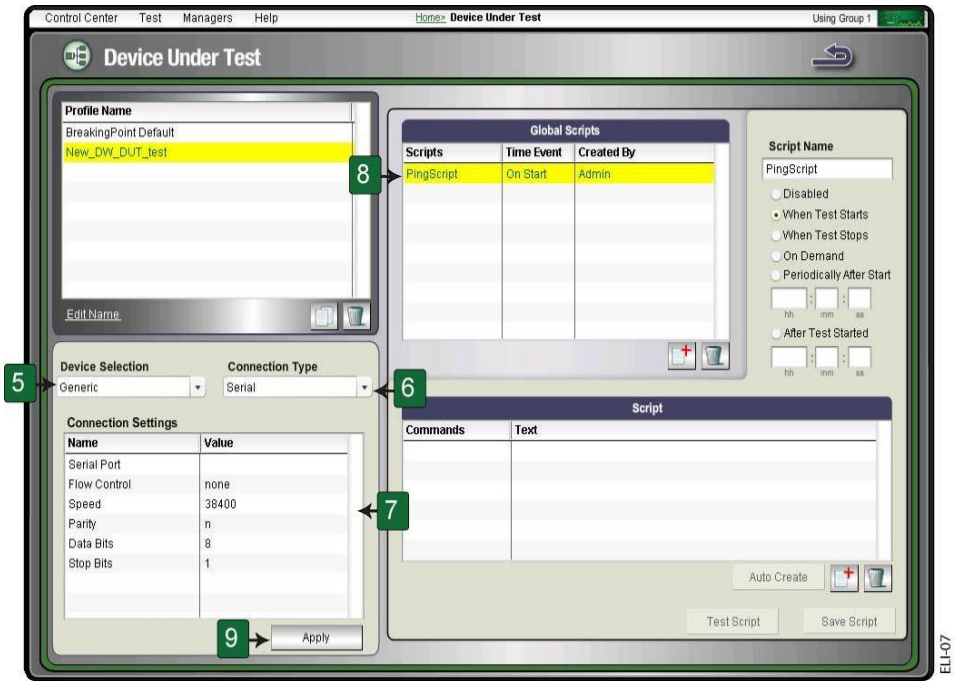
Callout	Name	Description
1	Menu Bar	Provides point and click access to the main areas of the user interface.
2	Device Status Icon	Provides access to the Device Status area so that you can reserve ports while no tests are running or the Real-Time Statistics screen if there is a running test.
3	Navigational Buttons	Provides access to areas within the user interface.
4	Firmware Version Information	Provides firmware version and update information.

Task 3: Creating a Device Under Test Profile

This section describes how to create a DUT Profile. A DUT Profile defines the connection settings for the device under test – such as the device's connection type, connection parameters, link type, and global commands. BreakingPoint uses these settings to connect to the device under test for remote scripting. For more information on DUT Profiles, see the DUT Profiles on page 95 section.

Note: BreakingPoint provides a default DUT Profile called BreakingPoint Default. This DUT Profile cannot be modified or deleted; however, it can be cloned and customized for your device.

Creating a DUT Profile



To create a DUT Profile:

1. Select **Control Center > Device Under Test** from the BreakingPoint Control Center Menu bar.
2. Select a profile from the **Profile Name** list to clone.
3. Click the **Clone the selected DUT** button.
4. Enter a name for the DUT Profile in the **Name** field and click the **OK** button.
5. Click the **Device Selection** drop-down button and select a device type. (Optional)
   a. Each device type has its own set of global commands. Select the device type that best fits your device.
6. Click the **Connection Type** drop-down button and select Telnet, SNMP, SSH, or Serial.
   a. If you have selected **Serial**, the DUT must be plugged into the chassis through the BPS Management serial port. If you have selected Telnet or SSH, the DUT must be plugged into the chassis through the BPS Management Ethernet port.

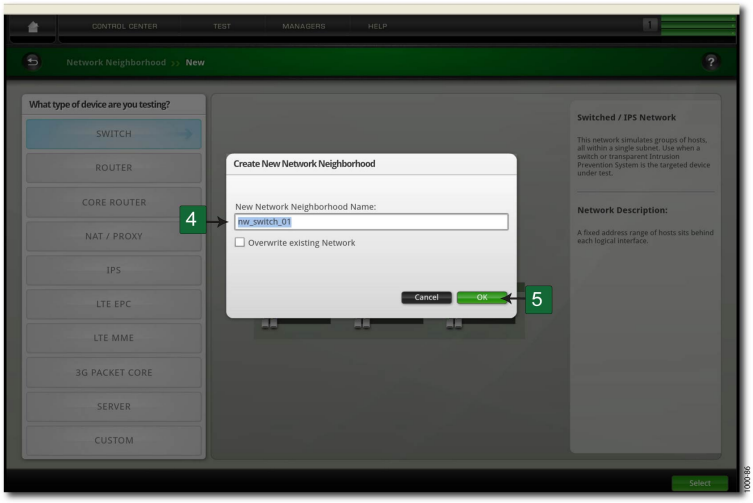
7. Define the connection parameters for the DUT under the **Connections Settings** area.
8. Enable or disable any global commands from the **Global Commands** list.
9. All cloned DUT Profiles will inherit the active global commands from its parent DUT Profile.
10. Click the **Apply** button.

Task 4: Creating a Network Neighborhood

The Network Neighborhood contains the addressing rules available for each test interface. Each test interface has a set of subnets to define the addressing rules for test traffic originating from each test interface.

Creating a Network Neighborhood

This section describes how to create a Network Neighborhood.



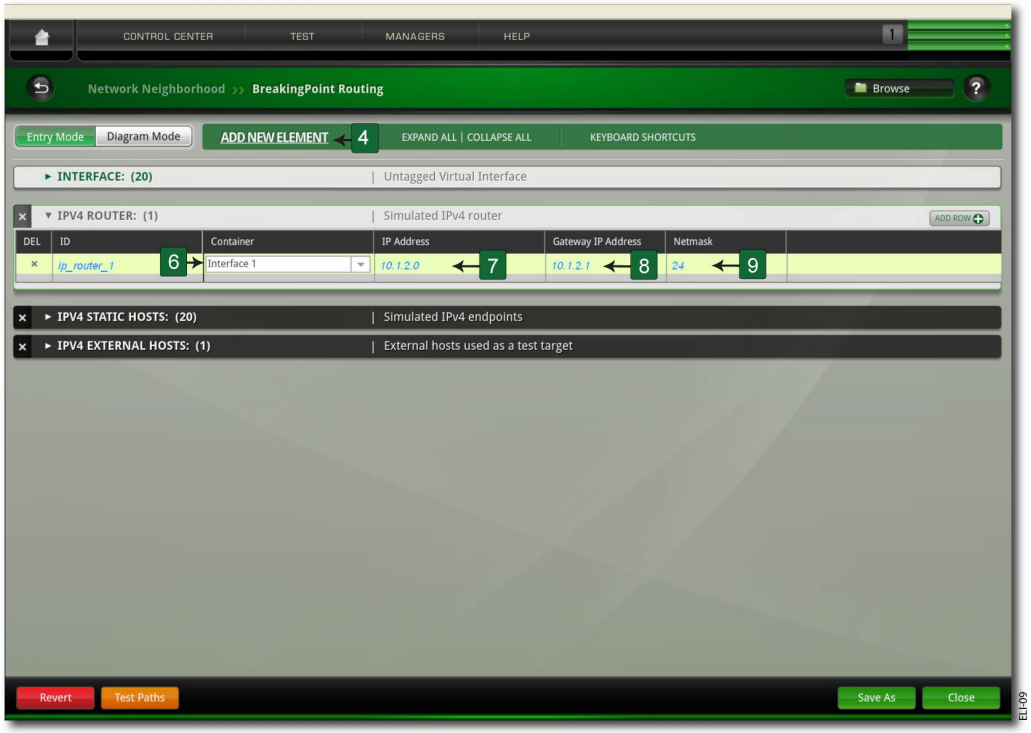
To create a Network Neighborhood:

1. Select **Control Center** > **New Neighborhood** from the BreakingPoint Control Center Menu bar.
2. Select the type of device you are testing.
3. Click the **Select** button.
4. Enter a name for the Network Neighborhood in the New Network Neighborhood **Name** field.
5. Click the **OK** button.

Note: The first time you create a Network Neighborhood, the Network Neighborhood Introduction page will be displayed. This page provides helpful information about the Network Neighborhood screen. Click the **Close** button on this page to continue with setting up your Network Neighborhood.

Defining a Subnet

This section describes how to add a subnet to a non-VLAN tagging subnet on a non-external interface. For information on external device addressing or VLAN-enabled addressing, see External Interface Addressing on page 157.



To define a non-VLAN subnet:

1. Select **Control Center** > **Open Neighborhood** from the BreakingPoint Control Center Menu bar.
2. Locate the Network Neighborhood from the Network Neighborhood list.
3. If the Network Neighborhood you want to select does not appear in the list, enter a portion of the name of the Network Neighborhood into the **Filtered Search** field and click the **Search** button.
4. Select a Network Neighborhood and then click **Open**.
5. Click the **Add New Element** link to add the type of element (or elements) you want to use in your network configuration if it is not displayed. For example, select **IP Infrastructure** > **IPv4 Router**.
6. Click the arrow to expand the tab for the first element in the Network Neighborhood.
7. If you are defining a subnet for the external interface (for endpoint testing), see External Interface Addressing on page 157.
8. Select Interface 1 in the **Container** field of the IPv4 Router interface.
9. Enter an IP address in the **IP Address** field of the IPv4 Router tab. Use the format x.x.x.x, where x is a number between 0-255.

10. Enter a gateway IP address in the **Gateway IP Addr**ess field of the IPv4 Router interface. Use the format x.x.x.x, where x is a number between 0-255.
11. Enter a netmask in the **Netmask** field of the IPv4 Router tab.
12. Click the **Add Row** button in the IPv4 Router tab.
13. Repeat steps 5 - 12for each additional interface you want to add to this element.
14. Click **Save** when you are done.

Adding a Test Interface

Each test interface in the Network Neighborhood corresponds to a data port on the chassis. When you add an interface to a Network Neighborhood, the system will automatically number the interface based on the order in which it was added.

If you delete any of the interfaces, the system will automatically resequence the interfaces. The succeeding interfaces (following the deleted interface) will be renumbered to the preceding interface's value (for example, "6" will become "5").

Note: There can be up to twenty-four test interfaces in a Network Neighborhood and one external interface.

To add a test interface to a Network Neighborhood:

1. Select **Control Center > Network Neighborhood** from the BreakingPoint Control Center Menu bar.
2. Select a Network Neighborhood from the **Network Neighborhoods** list.
3. Click the **Add Row** button.
4. Once you have added the interface to the Network Neighborhood, you can add and modify addressing information.

Task 5: Making Port Reservations

The number of tests that you can run concurrently depends on the number of available ports that the BreakingPoint system has. For example, a single-blade BreakingPoint device with four available ports can only run four tests at a time. A two-blade chassis with sixteen total available ports can run sixteen tests simultaneously. However, in order to run all sixteen tests concurrently, you will need to assign each available port to a different Active Group.

In order to run tests on BreakingPoint, you must make port reservations. A port reservation occurs when you click on a port to reserve it under your account.

When you click on a port to reserve it, the system will lock the port reservation under your account. Locking a port reservation will also reserve all other ports under your account as well; however, only the ports with locked reservations can be used to run tests.

Note: In order to run two tests concurrently, each set of ports must be assigned. If you have a single port and not a pair, you will receive an error when you try to run the test. The error will state that the Interface has not been assigned.

There are three ways to reserve a blade:

- Reserving an unreserved blade
- Force reserving a reserved blade
- Simultaneously reserving or unreserving a blade

Reserving an Unreserved Blade

Unreserved blades may be reserved simply by selecting the Active Group to which you would like to assign the blade, and then clicking on the port you would like to reserve. This will lock the port reservation. Right-clicking on a port and selecting Reserve all ports on this slot will reserve all ports on the blade.

Note: A key icon along with the Active Group will appear on all the ports on the blade that you reserve.

An important thing to remember when reserving your ports is the order in which you reserve them. Whenever you reserve a port, the system will automatically map that port to an interface on the chassis.

For example, if you reserve ports 0 and 1, then port 0 will map to interface 1 and port 1 will map to Interface 2. You can use these interfaces to run tests. If an interface is not mapped to a port, then you cannot use that interface to run tests.

If you want to remap the ports to different interfaces, you can click on the **Port Mapping** options, located on the **Device Options** screen, and manually remap the ports.

Note: Only reserved ports will can be mapped to interfaces.

To reserve ports on an unreserved blade:

1. Select **Control Center** > **Device Status** from the BreakingPoint Control Center Menu bar.
2. Click the **Active Group** drop-down menu.
3. Select the Active Group to which you would like to assign the ports.
4. Click on the port(s) you would like to reserve.
5. A lock will appear over the reserved port. All other ports will be tagged with an icon denoting the port's Active Group. These ports, even though they have not been manually reserved by you, will be reserved under your account.

Force Reserving a Blade

If another user has reserved the ports on a blade, you can force reserve all the ports on that blade by clicking on any of the ports. During a force reserve, the system will alert you that the ports are reserved by another user and ask if you want to force reserve all the ports on that blade. If you force reserve the port at this point, the system will reserve all the ports on that blade under your account.

Note: You cannot force reserve ports if there is a test or system process running on any of the ports on the blade. This system will alert you that there is a process running on that slot.

You should check the port notes before you force reserve the port(s) because other system users may not want you to remove their port reservations. If available, the port notes will appear as a yellow note icon located below the port.

As a best practice recommendation, you should add a port note to your reserved ports. For example, you may want to note that you will be running tests on these ports everyday between 2 and 4 p.m. This may prevent other users from removing your port reservations.

To force reserve ports:

1. Select Control **Center > Device Status** from the BreakingPoint Control Center Menu bar.
2. Click on the port(s) you would like to reserve.
3. You can only force reserve ports that do not have tests or system processes running on them.
4. Click **Yes** when the dialog window displays, asking if you would like to force reserve all the ports in the slot.
5. The port(s) that you clicked on will show a key icon, denoting that this port has been reserved by you. All other ports will be tagged with an icon showing the active group to which the ports belong.

Simultaneously Reserve or Unreserve All Ports On A Blade

When you right-click on a port, you can conveniently reserve or unreserve all ports on that slot without having to individually select them.

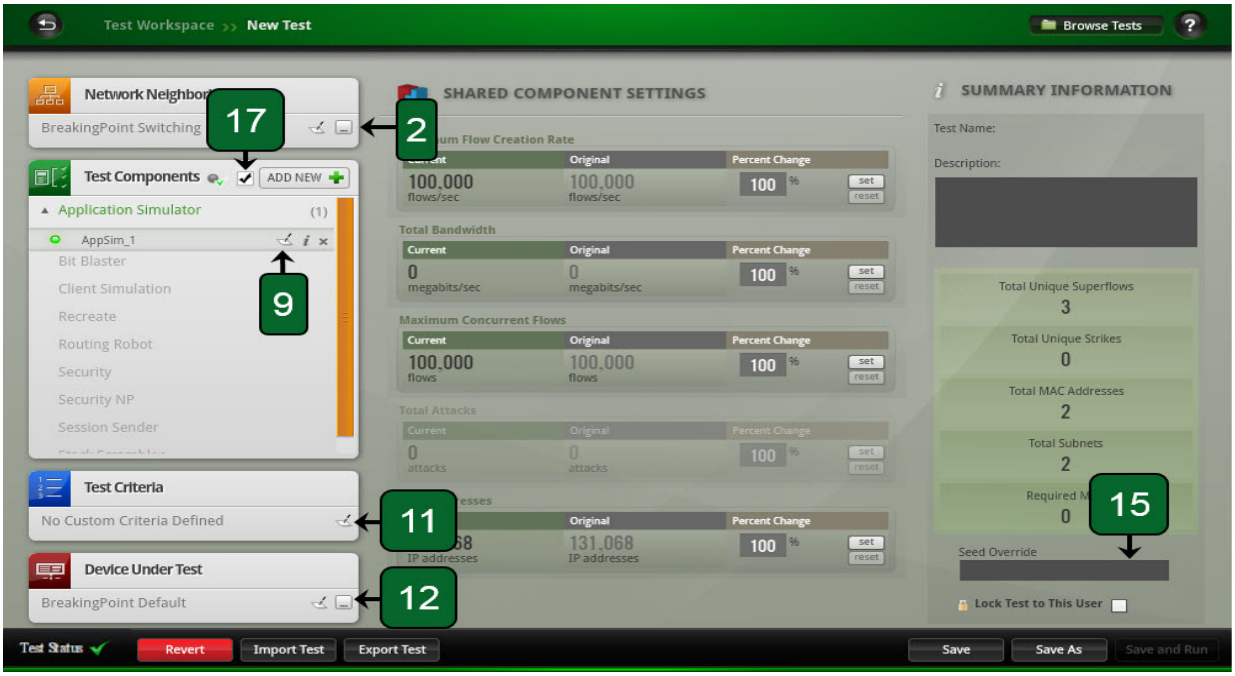
To simultaneously reserve or unreserve all ports on a blade:

1. Select Control Center > Device Status from the BreakingPoint Control Center Menu bar.
2. Click the Active Group that you would like to use from the drop-down menu.
3. Right-click on the slot that has the ports you would like to reserve or unreserve.
4. Select the Reserve/Unreserve all ports on this slot option.

Task 6: Creating a Test

This section describes how to create a test from start to finish; this includes selecting the Network Neighborhood and DUT Profile, adding a test component, configuring the test component, and running the test.

Creating a Test



To create a test:

1. In the Control Center, select **Test > New Test** from the BreakingPoint Control Center Menu bar.
2. Click the **Browse Available Network Neighborhoods** icon in the **Network Neighborhood** section.
3. Select a Network Neighborhood from the list.
   a. If you do not see the Network Neighborhood you want to use, enter a portion of the name into the Browse Network field.
4. Click the **Select** button. The name of the Network Neighborhood you selected will be displayed in the Network Neighborhood section.
5. Click the **Add New** in the Test Components section.
6. Select a component from the list and click the Select button.
7. Enter a name and description for the component. (Optional)
   a. Select the Use Template check box to select a pre-configured test component. (Optional)
8. Click **Create**. The name of the component you selected will be displayed in the Test Components section.
9. Click the **Edit component** icon to make changes to the test component. (Optional) Component modifications include:
   a. Including or excluding component results in the test report
   b. Modifying test component parameters
   c. Assigning network component tags to the component
   d. Rendering the component State Active or Inactive
10. Click **Return to Test Workspace**.

11. Click the **Edit test criteria** icon in the Test Criteria section and create the pass/fail criteria for the test. For more information on pass/fail criteria, see the section Test Pass/Fail Criteria in this guide.
12. Click the **Browse available DUTs** icon in the **Device Under Test** section.
13. Select a DUT from the **Browse DUT** list.
    a. If you do not see the DUT you want to use, enter a portion of the name into the Browse DUT field.
14. Click **Select**. The name of the DUT you selected will be displayed in the Device Under Test section.
15. Click the **Edit DUT** icon link to modify the DUT Profile. Once you have made your changes, click **Return** button to return to the Test Workspace screen. For more information on DUT Profiles, see Task 3: Creating a Device Under Test Profile on page 54.
16. Enter adjustments into the Percent Change field of the Shared Component Settings section. (Optional)
17. Enter a value in the Seed Override field. (Optional)

> **Note:** Use the Seed Override to modify the seed for Security, Application Simulator, and Stack Scrambler tests. The Seed Override enables you to control whether static or dynamic content will be generated. If you explicitly set the seed, the system will recreate the same application flows each time the Super Flow is run. If you do not explicitly set a seed, the system will automatically randomize a seed for the Super Flow each time it is used.

18. Use the Seed Override to modify the seed for Security, Application Simulator, and Stack Scrambler tests. The Seed Override enables you to control whether static or dynamic content will be generated. If you explicitly set the seed, the system will recreate the same application flows each time the Super Flow is run. If you do not explicitly set a seed, the system will automatically randomize a seed for the Super Flow each time it is used.
19. Optionally click the **Component Enable** button. A wizard displays which allows you to easily disable or enable specific test components. When filter criteria is entered, only the components with names that match a portion of the entered criteria will be displayed. Click the check box next to a component name to disable/enable it. Click the **Submit** button to accept the changes or click the **X** to cancel.
20. Click **Save As**.
21. Enter a name for the test in the **Name** field.
22. Click **Save and Run** to save and run the test.

Seed Override

The Seed Override is used to modify the seed for the test. The seed is used in Security, Application Simulator, and Stack Scrambler tests whenever there is a value that can be randomized. The purpose of the seed is to provide randomness and/or predictability. Establishing a set value for the seed will give you the ability to create a test with results that are reproducible.

From the Test Panel, you can enter a numerical value into the Seed Override field to override the seed.

This page intentionally left blank.

CHAPTER 6 Administration

This section covers the following topics:

Accessing the Administration Pages

Your BreakingPoint system provides three complimentary areas of administration.

Ixia Web Platform Administration - Used to administrate IxOS and some BPS chassis level settings. See the *IxOS Getting Starting Guide* for more information on IxOS.

BreakingPoint Administration **-** Allows you to perform administrative tasks such as creating user accounts, utilizing the database optimization feature and downloading ATI Updates.

BPS Control Center Administration **-** The BreakingPoint Control Center Administration page is where you can import tests, export tests, and manage BPS card licensing.

You should also be aware of the Gear icon  options that allow you to perform various administrative tasks.

Gear Icon Menu Options

Upon logging into the Ixia chassis, the Gear icon  (which is located at the top right corner of the Ixia Web Platform Dashboard) provides several system level menu options. After accessing the BreakingPoint Application, the Gear icon that appears in the BreakingPoint Dashboard provides a different set of system level and BPS specific options.

This documentation will use the following terms to differentiate between the specific Gear options.

- Ixia Web Platform Dashboard Gear
- BPS Application Dashboard Gear

Ixia Web Platform Dashboard Gear

Upon logging into the Ixia chassis, the Gear icon (which is located at the top right corner of the Ixia Web Platform dashboard) provides the following options:

Administration

- BreakingPoint - BPS admin options such as software and ATI updates, restore system configuration, storage management, and BPS system restart.
- System - Chassis admin options such as WebPlatform updates, authentication options, add users and license management.

My Account

Options to configure your user account settings such as password and email address.

REST API

- Documentation - An index of REST API Services.
- Python Lib - Downloads the Python Library files that can be used with REST API.

Note: For BPS RESTful API documentation see RESTful API. For details on downloading and using the Python Libraries, see Python REST Overview and Examples.

Help

- Docs - Online Help for the Web Platform .
- System Diagnostics - Allows you to generate BPS, Port and Chassis System logs.

About

Provides current application version and disk usage information.

Logout

Logs you out of the system.

BPS Application Dashboard Gear

After accessing the BreakingPoint Application, the Gear icon that appears in the BreakingPoint Dashboard provides the following options:

Administration

- BreakingPoint - BPS admin options such as software and ATI updates, restore system configuration, storage management, and BPS system restart.
- System - Chassis admin options such as WebPlatform updates, authentication options, add users and license management.

My Account

Options to configure your user account settings such as password and email address.

REST API

- Documentation - BPS RESTful API help documentation.
- Services - An index of REST API Services.
- BPS REST API - Allows you to download the Python Library files that can be used with REST API, see Getting Started with Python REST Calls for more information.

Help

- Ixia Help - Ixia Web Apps Help documentation.
- Ixia BreakingPoint EULA - Link to the Ixia Software End User License Agreement.
- BPS Control Center - BPS help documentation.
- Downloads
  - Strike Center - Links to the Ixia Support website where you can log in to get ATI Updates
  - Download Tcl Shell - Provides links to the Tcl Shell for several operating systems
  - Enhanced Shell - Provides link information to get the Enhanced Shell for several operating systems
  - BPS Robot Library - Downloads BPS Robot Library files
- Report an issue
- System Diagnostics

About

Provides current application version and disk usage information.

Logout

Logs you out of the system.

Web Platform Administration

To access Web Platform Administration:

1. Log into the Ixia chassis.
2. Click **Launch** under Administration or select the **Web Platform Gear** icon  **> Administration > System**.

The following options will display.

System Settings Tab

Updates - Allows you to check for and install Web Platform Updates

Authentication - Allows you to select Local or TACACS+ authentication.

Maintenance - Allows you to enable TLS v1.0 and Restart the BPS system.

Note: If you restart the system, all active sessions and running tests will be terminated.

Server Certificate - Allows you to upload your custom X.509 certificate for the web server to use for SSL/TLS authentication. The example provided on this page in the BPS UI describes in detail how to configure the following fields:

- PKCS#12 File: The PKCS#12 bundle with the certificate, private key and optionally the certificate chain.
- Bundle Password: This is the password that is optionally used to protect the PKCS#12 bundle.
- Alias: This is the certificate friendly name (alias) in the PKCS#12 bundle.

Users Tab

Allows to manage (add, delete and modify) user settings.

License Manager

Allows you to manage IxOS and other Ixia Product Licensing.

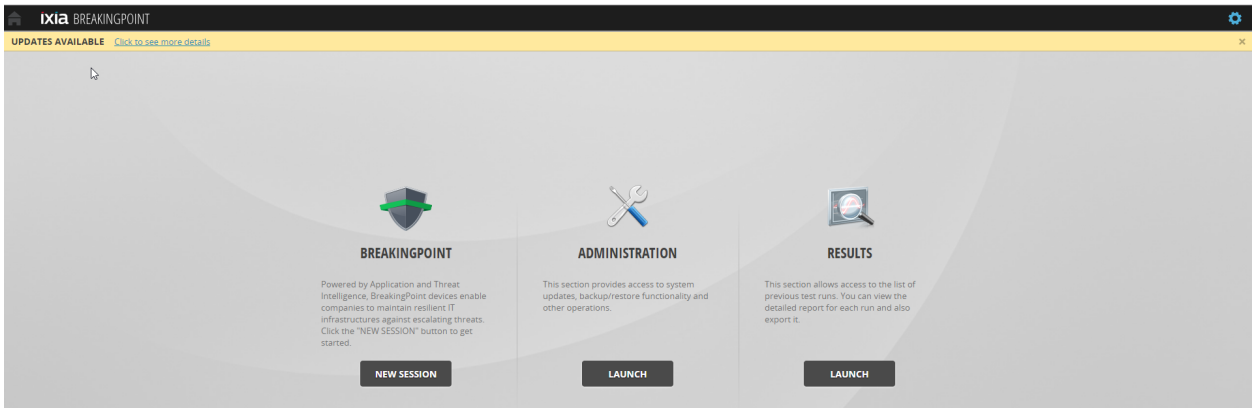
To license BreakingPoint load modules (cards), see BPS Control Center Administration and System Functions.

BreakingPoint Administration

To access BreakingPoint Administration:

1. Log into the Ixia chassis.

After logging in to the Ixia chassis and selecting the BreakingPoint Application, the BreakingPoint Dashboard will display as shown below.



The upper left corner of the page provides information about currently available updates and a link to the **System Settings** > **ATI Updates** page.

The following options are available in the BreakingPoint Dashboard.

BreakingPoint - Click **New Session** to configure and run BPS tests.

Administration - Click **Launch** to access the BPS Administration Page.

Results - Click **Launch** to view test results and generate Reports.

The upper right corner of the page provides the BPS Application Gear that provide BPS and System administration menu options.

2. When you click **Launch** under **Administration** or select **BPS Application Gear** **> Administration > BreakingPoint**, the following options display.

ATI Updates

See Updating the System.

BPS Software Updates

See Updating the System.

Storage

The Storage section provides the following options.

Backup User Data - Use this options to backup all user data including tests and test results. See Backup User Data on the facing page for details.

Restore User Data - Use this option to restore all saved user data. This action will replace all of your existing data with data from the backup file. See System Restore below for details.

Compact Storage - Use this option to eliminate the space occupied by any unused or deleted items to free up storage space for new items. Fragmentation of BPS physical storage may occur when tests are created and then deleted resulting in free space regions on the disk. Usually these free space regions are reused when a new test is created. However, if it absolutely necessary to recover disk space, Ixia recommends deleting a number of tests/reports and then running Defragment Database.

Current BPS disk usage is displayed in the table available at: **Administration** > **System Settings** > **Updates** > **System Information**.

Note: Compact Database operations are very resource and time intensive. The process may take several hours and should only be run when absolutely necessary and at a time when the system is not in use. Defragmentation of BPS physical storage that is not very fragmented may still take several hours to complete.

Purge Reports - Use this command to delete all reports. Note that tests will not be deleted.

System Restore

This feature is used to restore user data including tests and results. Any configuration changes or test results created after the backup was made will be lost. After restoring the system, you must restart it to bring up the restored configuration.

Note: Tests cannot be running while a restore is in progress.

To restore a backup:

1. If you plan to restore from a physical (USB) drive, attach the drive to the chassis.
2. Log in to the system with an admin account.
3. Stop any tests that are running.
4. Click **Administration | System Settings**.
5. In the System information area, click **Restore**.
6. Configure the restore settings. (See the Backup/RestoreOptions on the next page for parameters and descriptions.)
7. Select the backup that you want to restore, then click **Restore**.
8. When the restore is complete, the system needs to be restarted to load the restored configuration.
9. Stop any running tests, and inform other users that you are restarting the system.
10. Click **Restart** to restart the system.

Maintenance

This settings provides a **Restart** button that can be used to restart your BPS system.

Note: If you restart the system, all active sessions and running tests will be terminated.

Backup User Data

This feature is used to backup user data. Note that that this feature does not backup system data.

Notes on Backing Up User Data:

- Backups may take a long time (sometimes more than an hour) , so plan accordingly.
- Tests cannot be running while a backup is in progress.
- You can backup to: USB drive (flash drive or disc drive connected over USB), NFS network drive or to a local file on your computer.
- When backing up to NFS, it is recommended to have at least a 1 Gbps link to the NFS Server before beginning the Backup or Restore procedures.
- To backup to an NFS drive, the drive must be mountable without user credentials. There is no way to supply NFS user credentials through the system.
- Ixia recommends that you back up to FAT32 or EXT3-formatted drives. You cannot backup to FATor NTFS-formatted drives.
- The backup drive must support long file names.
- The first partition on the backup drive must be one of the supported file system types (such as FAT32 or EXT3).
- If the backup process prompts you to select the partition table type and the choices are GUID, Apple, BSD, or Master Boot Record (MBR), select MBR.

To back up the system:

1. If you plan to backup to a physical (USB) drive, attach the drive to the chassis.
2. Log in with an admin account.
3. Click **Administration > System Settings > Storage**.
4. Click **Backup User Data**.
5. Configure the backup settings. (See the Backup/Restore Options table below for parameters and descriptions.)
6. Click **Backup** to start the backup.

Backup/RestoreOptions

Parameter	Description
Backup Destination or Restore Source	Local computer
	NFS Share - A NFS drive. • Specify the IP address of the NFS host. • Specify the NFS host root patch.
	External USB Drive - Physical drive connected to one of the chassis USB slots.

BreakingPoint Control Center Administration

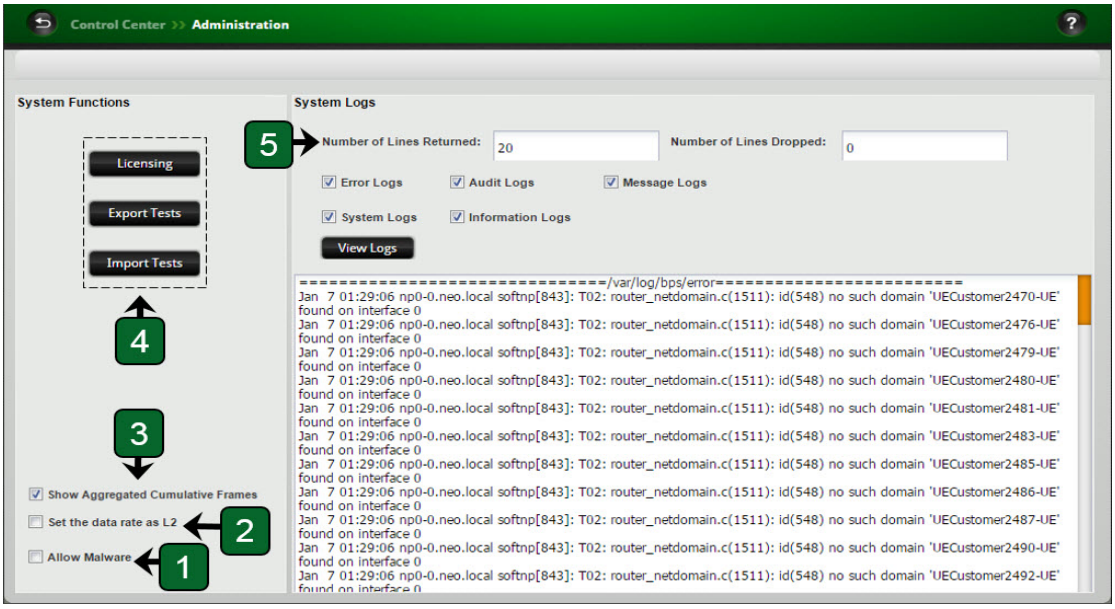
BreakingPoint Control Center Administration page is where you can import tests, export tests, and manage BPS card licensing.

To license IxOS, see IxOS and other Ixia Product Licensing.

To access the BreakingPoint Control Center Administration:

1. Log into the Ixia chassis.
2. In the Ixia Web Platform window, select the BreakingPoint Application.
3. In the BreakingPoint Dashboard, click BreakingPoint **New Session**.
4. The BreakingPoint Control Center is displayed.
5. Select **Control Center > Administration**. The Administration page is displayed.

Administration Page



The Administration Page Elements table below lists and describes sections of the Control Center Administration Page.

Administration Page Elements

Callout	Name	Description
1	Allow Malware	Select this option if the user will be allowed to run a test that sends malware to a DUT.
2	Set the Date Rate as L2	Determines whether or not the Bandwidth Tx/Rx statistics include Layer 1 data. If enabled, Bandwidth statistics do not include Layer 1 data. If disabled, Bandwidth statistics include Layer 1 data.
3	Show Aggregate Cumulative Frames	When this option is enabled (the default mode), Real-Time Statistics will report all packets that arrive at a port regardless as to how the packets reached the port. When this option is disabled, Real-Time Statistics will only report packets that match the configuration of the current network neighborhood. **Note:** The state of this options does not affect a capture - BPS will always capture all packets that arrive at a port.
4	System Functions	Provides controls for importing licenses and for exporting and importing multiple tests. Click **Licensing** to view current licensing information. From the displayed Licenses table you can click **Import Licsense** to import a license.
5	System Logs	Stores logs of the system's activity.

Licensing

Load Module Licensing

To license BreakingPoint load modules (cards), see BPS Control Center Administration and System Functions.

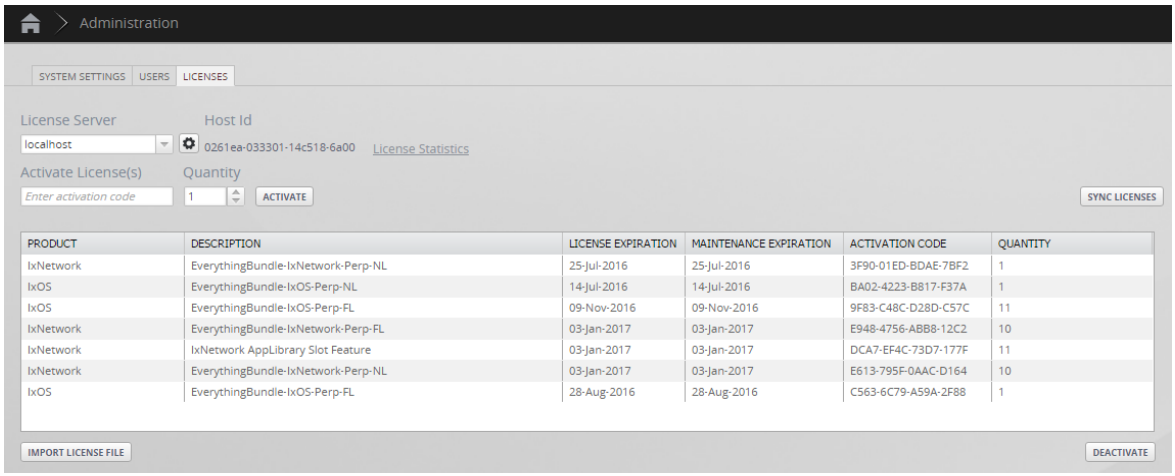
Note: An IxOS license provides functionality to start IxServer. IxServer is required to run BPS Load Modules.

IxOS and other Ixia Product Licensing

Note: The Licenses tab is only available on PerfectStorm running on Native IxOS. PerfectStorm running on Windows IxOS will not have this option.

The Chassis Management Console is a licensed Ixia application and should have a registered license before it can be used. The licenses are stored on a license server.

The LICENSES tab allows you to select the usage of the external license server. You need to activate the licenses in order to use them.



You can perform the following tasks:

- Activate Licenses
- Deactivate Licenses on the facing page
- Manage License Servers
- Sync Licenses
- Import License File
- View License Statistics

Activate Licenses

To activate licenses, do the following:

1. Click the Breaking Application Dashboard Gear  and select **Administration**.
2. Click the LICENSES tab.
3. In the **License Server box**, type or select the name of the license server.

You use a local or an external license server. If you select an external server, then enter the IP address of the server that you plan to use to activate your Chassis Management Console licenses.

4. In the Activate Licenses(s) box, type the activation code.
5. In the adjacent box, type or select the number of licenses you want to activate and then click **Activate**. A message appears in the Notifications pane, indicating "License successfully activated".

The activated license appears in the Licenses table (described below) and the information for each license is listed.

Parameter	Description
Product	The name of the Ixia application for which the license is activated.
Description	The type of license.
License Expiration	The displayed date indicates the license expiration.
Maintenance Expiration	The maintenance end date for licenses.
Activation Code	The license activation code.
Quantity	The number of activated licenses.

Deactivate Licenses

To deactivate the licenses, do the following:

1. Click **Deactivate**. A New Quantity column is added to Licenses table.
2. In the New Quantity column, type the number of licenses you want to deactivate for the selected product.
3. Click **Perform Deactivation**. The selected licenses are deactivated.
4. Click **CANCEL** if you do not want to deactivate any license.

Manage License Servers

1. Click  to view the license servers.
2. In the **License Server box**, type the IP address or host name of the license server.
3. Click **Ad Server**. The server name is added to the **License Servers List**.

4. To remove a license server from the list, select the server and click **Delete Server**.
5. Click **Close** to exit from the **Manage License Servers** window.

Sync Licenses

If licenses are renewed, you can sync all the license changes made on the local license server by clicking **Sync Licenses**. After syncing, a message appears in the Notifications pane, saying Licenses successfully synced. As an example, supposing the maintenance end date for a given activation code is extended, you can update the local license file with the new dates by using the Sync Licenses button.

If there is internet connectivity, this synchronizes the local license server with the Ixia backend, yielding an up-to-date maintenance end date.

In the case of no internet connectivity, the steps for offline license synchronization must be taken. For details on this, see the Ixia Licensing Management User Guide (Sync Licenses - Offline section).

Import License File

On the Licenses tab, click **Import License File**. The Import License File window appears.

Follow the instructions provided in the window to import a license file from a remote computer to your local computer.

Note: You will need the Host ID during the license import process.

View License Statistics

The following licensing statistics are available when you click the License Statistics link.

Parameter	Description
Feature	The name of the feature for which the license is activated.
Maintenance End Date	The maintenance end date for licenses.
Borrowable	The number of borrowable licenses.
Available	The number of available licenses.
User	The name of the user.
Host	The name of the host machine.
IP	The IP address of the chassis.
Count Taken	The number of licenses.
Borrowed	The number of licenses that are borrowed.
Duration	The duration of time for which license is active.

Parameter	Description
Hours Used	The number of hours for which the license has been used.
Days Left	The number of days till which the license is active.

Backup and Restore User Data

This section describes how to Backup and Restore User Data.

Backup User Data

This feature is used to backup user data. Note that this feature does not backup system data.

User Data includes:

- user accounts
- custom tests
- test reports

Notes on Backing Up User Data:

- Backups may take a long time (sometimes more than an hour), so plan accordingly.
- Tests cannot be running while a backup is in progress.
- You can backup to: USB drive (flash drive or disc drive connected over USB), NFS network drive or to a local file on your computer.
- When backing up to NFS, it is recommended to have at least a 1 Gbps link to the NFS Server before beginning the Backup or Restore procedures.
- To backup to an NFS drive, the drive must be mountable without user credentials. There is no way to supply NFS user credentials through the system.
- Ixia recommends that you back up to FAT32 or EXT3-formatted drives. You cannot backup to FATor NTFS-formatted drives.
- The backup drive must support long file names.
- The first partition on the backup drive must be one of the supported file system types (such as FAT32 or EXT3).
- If the backup process prompts you to select the partition table type and the choices are GUID, Apple, BSD, or Master Boot Record (MBR), select MBR.

To perform a backup:

1. If you plan to backup to a physical (USB) drive, attach the drive to the chassis.
2. Log in with an admin account.
3. Click **Administration > System Settings > Storage**.
4. Click **Backup User Data**.
5. Configure the backup settings. (See the Backup/Restore Options table below for parameters and

descriptions.)

6. Click **Backup** to start the backup.

Backup/RestoreOptions Table

Parameter	Description
Backup Destination or Restore Source	Local computer
	NFS Share - A NFS drive. • Specify the IP address of the NFS host. • Specify the NFS host root patch.
	External USB Drive - Physical drive connected to one of the chassis USB slots.

Restore User Data

This feature is used to restore user data including tests and results. Any configuration changes or test results created after the backup was made will be lost. After restoring the system, you must restart it to bring up the restored configuration.

Be aware that a restore operation is actually replacing the old/existing database.

Note:

- If restoring a database that was created in release 8.01 or 8.10, user accounts are included in the restored objects but these user accounts need to have the passwords re-entered in order to be used in release 8.11 or higher. If the database was created in release 8.11 or higher, user accounts are also included in the restore but passwords are retained and do not need to be re-entered.
- Tests cannot be running while a restore is in progress.

To restore a backup:

1. If you plan to restore from a physical (USB) drive, attach the drive to the chassis.
2. Log in to the system with an admin account.
3. Stop any tests that are running.
4. Click **Administration | System Settings**.
5. In the System information area, click **Storage**.
6. Configure the restore settings. (See the Backup/RestoreOptions Table above for parameters and descriptions.)
7. Select the backup that you want to restore, then click **Restore**.
8. When the restore is complete, the system needs to be restarted to load the restored configuration.
9. Stop any running tests, and inform other users that you are restarting the system.
10. Click **Restart** to restart the system.

Restore Factory Configuration

The ability to restore the system to factory defaults was added for the XGS-HSL2/XHS-HSL12 chassis and PerfectStorm One. This functionality allows customers to revert all installed software to the version that the system was shipped with.

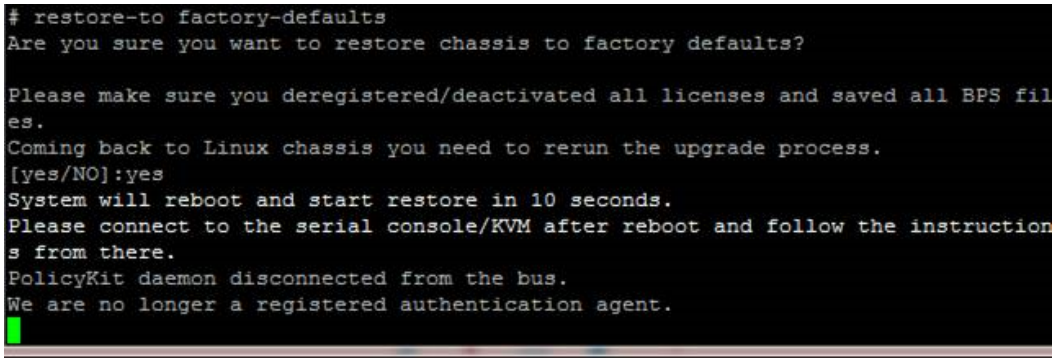
Note: This option is not available for systems that have been upgraded to Native IxOS in the field.

To restore the factory configuration:

1. Stop any running tests.
2. Connect to the chassis using a serial cable (speed: 115200, data bits: 8, stop bits 1, parity : none, flow control : none) for the PerfectStorm One appliance or a VGA/keyboard for XGS12-HSL/XGS2-HSL chassis.

   User admin/ password: admin
3. Run the following command:

   `restore-to factory-defaults`
4. Follow the onscreen instructions. Type `yes` and press the **Enter** key when you are ready to begin the restore.



Updating the System

Periodically, Ixia releases updated firmware and versions of the installed applications (the labs).

The following update options and information is available.

Multi-version Feature

Daily Malware Updates

Updating an ATI Package

Offline Updates

BPS Software Uppdates

BPS Software Updates

Note: The *BreakingPoint Release Notes* document that matches your current software version, provides the most up-to-date specifics about updating your BreakingPoint system. The information in this section provides general guidelines and background information on the process. Please review the release notes before updating your BPS software.

Note: Ixia recommends creating a backup before updating the system.

File Formats

- Firmware update files for PerfectStorm running in a Native IxOS system are provided in the following format, "update_lxc.x.x.bps".
- For PerfectStorm running on Windows IxOS and for legacy Firestorm/Storm systems, firmware files use the following naming convention: BPS-X-N.bps where N represents the update's firmwareversion, and X represents the oldest firmware version N will work with. Please see the documentationfor your pre- 8.11 release if you need to install a pre-8.11 version.
- Firmware update files for BPS VE are provided in the following format, "update_vm.x.x.bps".

Update Process

When using the Native IxOS, the BPS Multi-version feature allows the Ixia chassis to retain multiple versions of BPS software.

To update the firmware:

1. Log in with an admin account.
2. Click the **BreakingPoint** application icon.
3. Click the **Launch** button under the **Administration** section.
4. Click **BPS Software Updates**.
5. Click **Update System**.
6. Use one of the following options to select the update file:
   a. Browse to a Local File.
   b. Indicate a Web Location for the update file.
   c. Select a Server Location from the drop-down indicator. This option is used in a scenario where the firmware successfully uploaded to the chassis but the installation failed for some reason (for example, licensing issues). The firmware can be installed from the chassis using this option.
7. Click **OK.**
   a. If there are no tests running, click **OK** to confirm that you are aware that the system will automatically restart after the update has completed. Clicking **OK** will also begin the update process.
   b. If there are tests running and you want them to continue, click **Cancel**. When the tests have finished (or you have stopped them), complete this procedure to update the system.

Note: Restarting the system takes 2-5 minutes.

After the system has restarted, clear your browser's cache before you start a BPS test or lab.

Multi-version Feature

For Native IxOS platforms, the process of updating the system software also retains a copy of the previously installed BPS software. The BPS Multi-version feature gives you the ability to switch between the active software version and earlier installed versions (results database and users will be cleared in the switch process). You can also "update" to a version that is earlier than the currently active version.

The installed versions of BPS firmware can be controlled using the following Multi-version commands.

1. Access and log into the BPS CLI.

For example:

```
ssh admin@10.218.36.110 -p 8022
password = admin
```

2. Run one of the BPS Version Control commands described below.

- `show bps installed-versions`

  Displays all installed versions. An asterisk and bold text indicates the active version.
- `show bps active-version`

  Displays only the active version.
- `set bps active-version <version>`

  Sets the active version. Ensure to create a backup before you set the active version and that there are no actively running tests.

Note that the following occurs when this command is run:

a. The set command is verified.
b. The chassis web server is restarted.
c. When switching to a another BPS version, all results and user accounts will be cleared (**Restore User Data** functionality can be used to restore the users from a previous backup).

`uninstall bps <version>`

Uninstall a non-active version.

Daily Malware

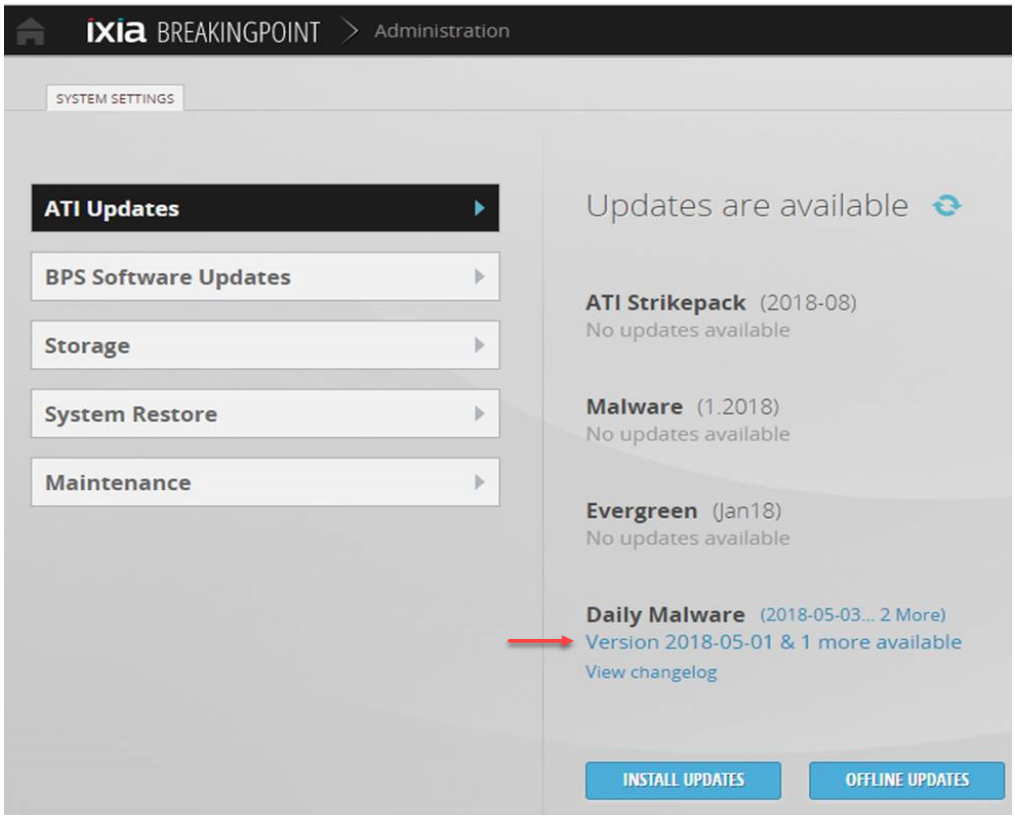
Daily Malware is a daily package update containing malware strikes for a particular day. It allows you to access new malware samples as seen in the wild on a daily basis in order to keep pace with the rapid changes that occur with threats. It is a complementary (not overlapping) malware feed to the monthly ATI Malware package (which is downloaded from the Customer Support Portal and installed offline).

Note: Daily Malware can only be obtained from an ATI Cloud Update (therefore it will only be available to users that have BPS version 8.40.2 or higher).

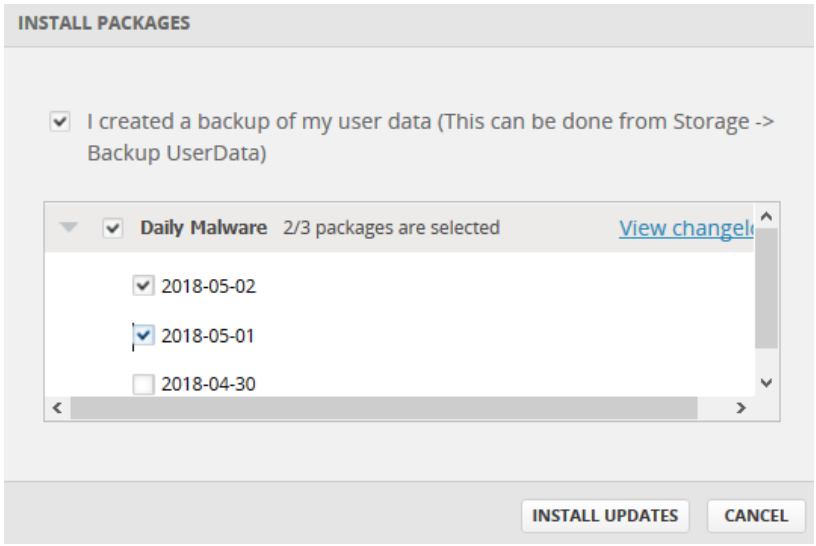
How to Install/Uninstall Daily Malware

To install Daily Malware:

1. In the BPS UI, select the **Launch** button under **Administration**.
2. Select **ATI Updates**.



3. Click the "**Version**..." link below **Daily Malware**. The Install Packages window will display.

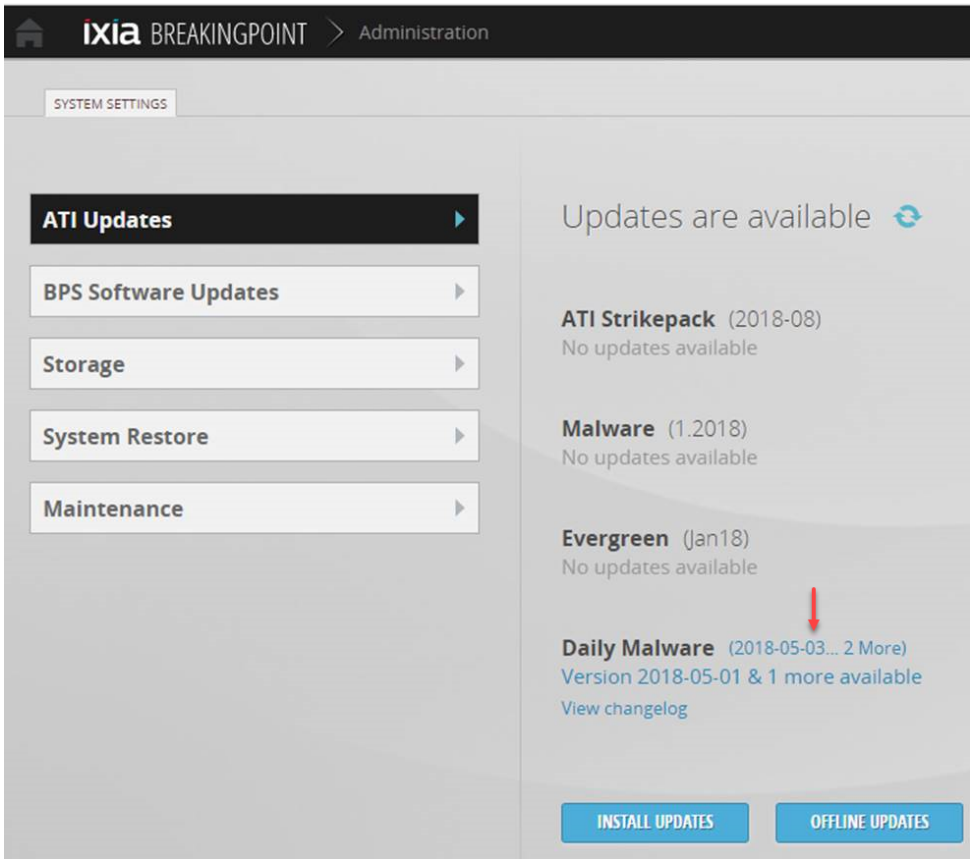


4. Select the **Backup Confirmation** check box to confirm that a system backup has been performed.

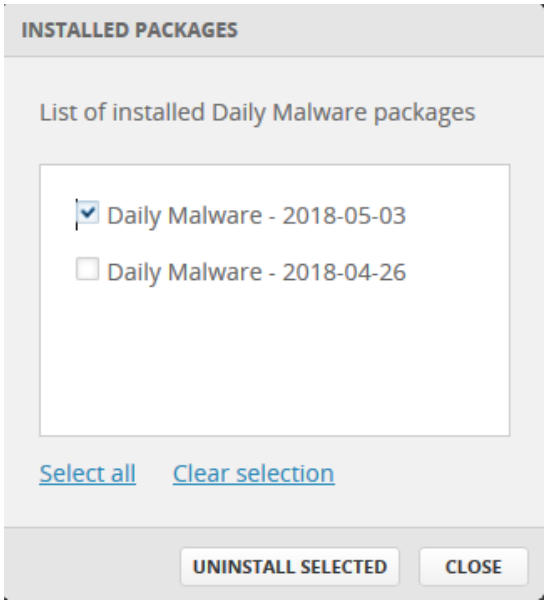
5. Click the check box that is next to the Daily Malware packages that you want to install.
6. Click the **Install Updates** button.

To uninstall Daily Malware:

1. In the BPS UI, select the **Launch** button under **Administration**.
2. Select **ATI Updates**.



3. Click the Installed Version link that is next to **Daily Malware**. The Installed Packages window will display.



4. Select the check box next to the Daily Malware packages that you want to uninstall.
5. Click **Uninstall Selected**.

Categories and Platforms

Daily Malware provides 14 Strikelists based on Category (4) and Platform (10) which are described below.

Note: Some Strikelists can be in more than a single category. For example, a Strikelist could be listed in both the Ransonware and Financial categories.

Category

- Ransomware
- APT (Advanced Persistent Threat)
- Retail
- Financial

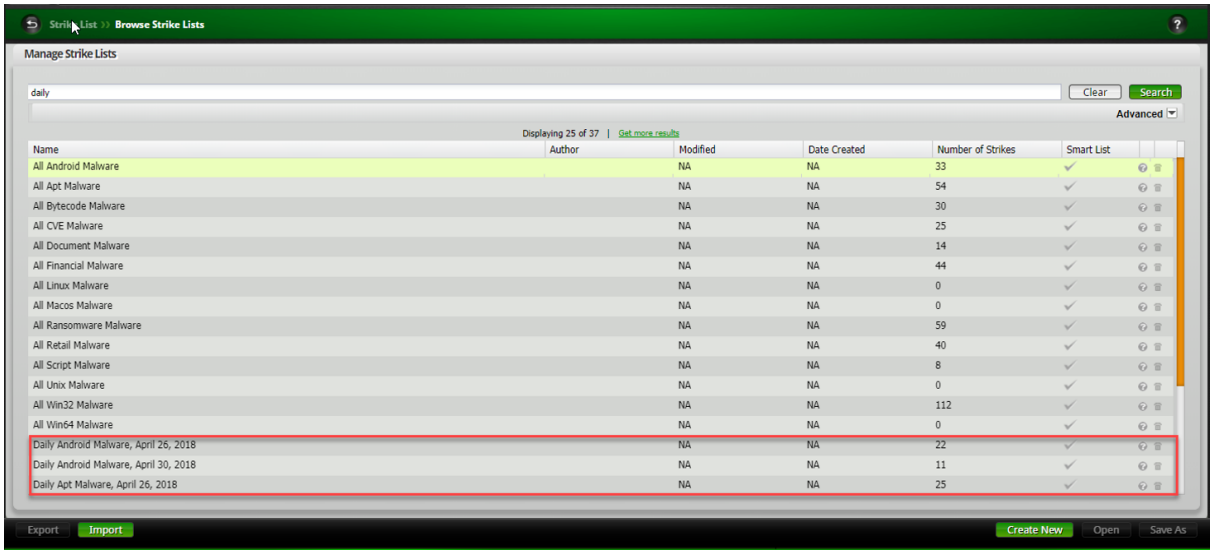
Platform

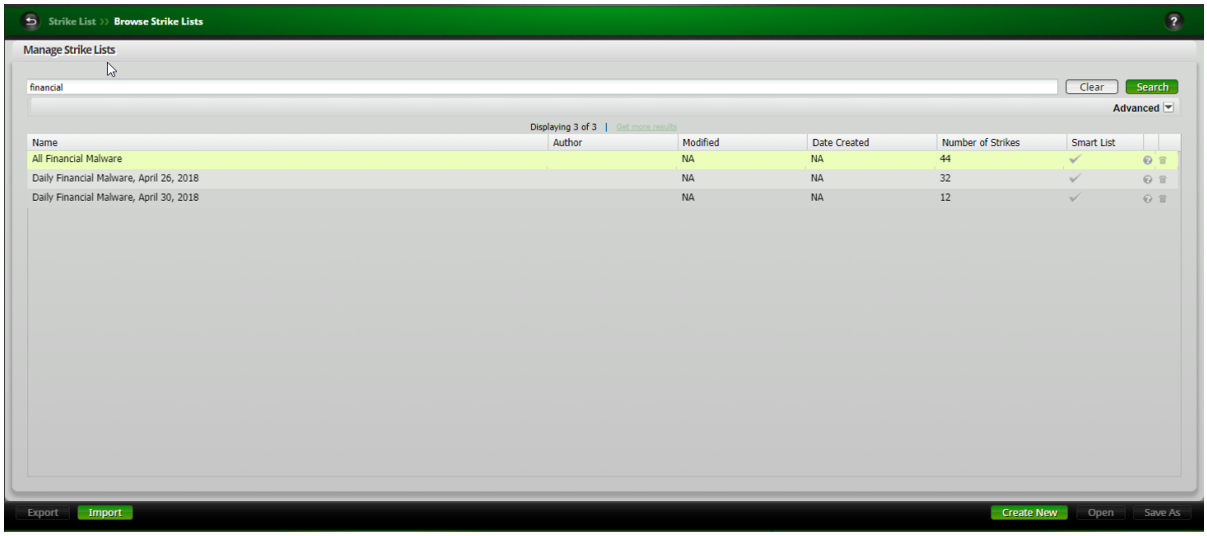
- Exploit
- Win32
- MacOS
- Linux
- Unix
- Document
- Android
- Script

- ByteCode
- Win64

After installing a Daily Malware package, you can locate the installed strikes by accessing **BreakingPoint New Session** > **Managers** > **Strikelists** and then searching for "Daily". You will see the Daily Malware that has been installed (see example in the 1st image below). You could also search for "Financial" to see only the Financial category strikelists (see example in the 2nd image below). Similar searches can be executed for the other Category and Platform keywords listed above.

To get more information about the Strikelists that are displayed in your search results, double-click the Strikelist or select it and then click the **Open** button.





Updating an ATI Package

The Application and Threat Intelligence (ATI) program provides frequent ATI updates ensuring delivery of the industry's most up-to-date application and threat intelligence. ATI packages include Strikepacks, Malware, Evergreen and Daily Malware.

There are 2 methods available for updating an ATI Package:

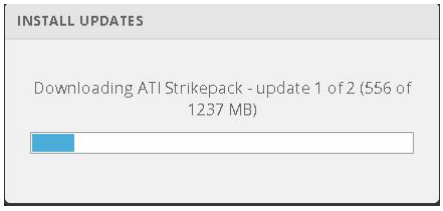
- ATI Cloud Updates - Install ATI updates from the Cloud. When BPS is connected to the internet, you will receive automatic notification of available ATI updates in the BPS UI, and access to the update files.
- ATI Offline Updates - Manually download ATI update files from the Ixia support website. Run the Offline Updates procedure.

Installation Behavior for Base and Incremental ATI Strikepack Updates

- A Base update is a full update which contains all ATI content produced to date. Therefore, a Base update can be a large file (approximately 1GB) as opposed to an Incremental update which will be smaller. Two Base updates will be made available per year.
- An Incremental update contains delta files from the baseline to the incremental version. For example, the Base update for BPS 8.50 is ati-2018-14. Incremental update ati-2018-16 includes all content released between updates #14 and #16 (including ati-2018-15 content). An Incremental update is much smaller than a Base update and can be installed 2 to 3 times faster than a Base update (depending on your BPS platform - Storm, PerfectStorm, etc.).

Note: The speed of your connection to the Cloud can also affect the time required to complete an update.

Note: When you install an Incremental update, there is a requirement for the corresponding Base build to be installed. If the required Base build is not installed on your system, it will be automatically installed before the installation of the Incremental update. In this scenario, you will see a progress bar (similar to the one shown below) that indicates the number of ATI Strikepacks being downloaded/installed. Upon completion, you will receive a confirmation which displays the ATI Strikepacks that were successfully installed.



Note: Currently, the installed ATI Strikepack version can only be downgraded by using the Offline Updates on page 88 procedure to install any older full Base build. For example, if you install an Incremental build and then decide to downgrade, perform the following steps:

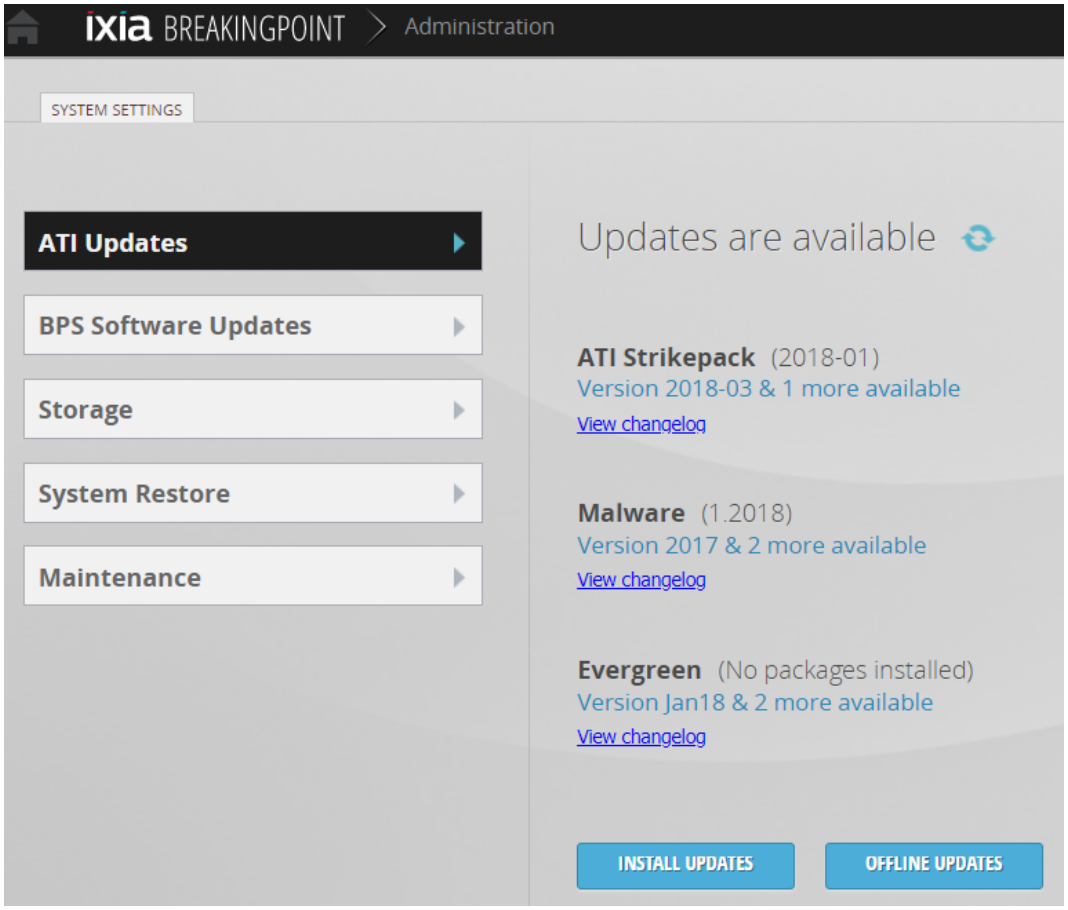
1. Download any full Base build that was released earlier than the currently installed Strikepack from the Ixia Customer Support portal.
2. Install the full Base build using the Offline Updates on page 88 procedure.

Note that downgrading to an Incremental build is not supported and will fail.

ATI Cloud Updates

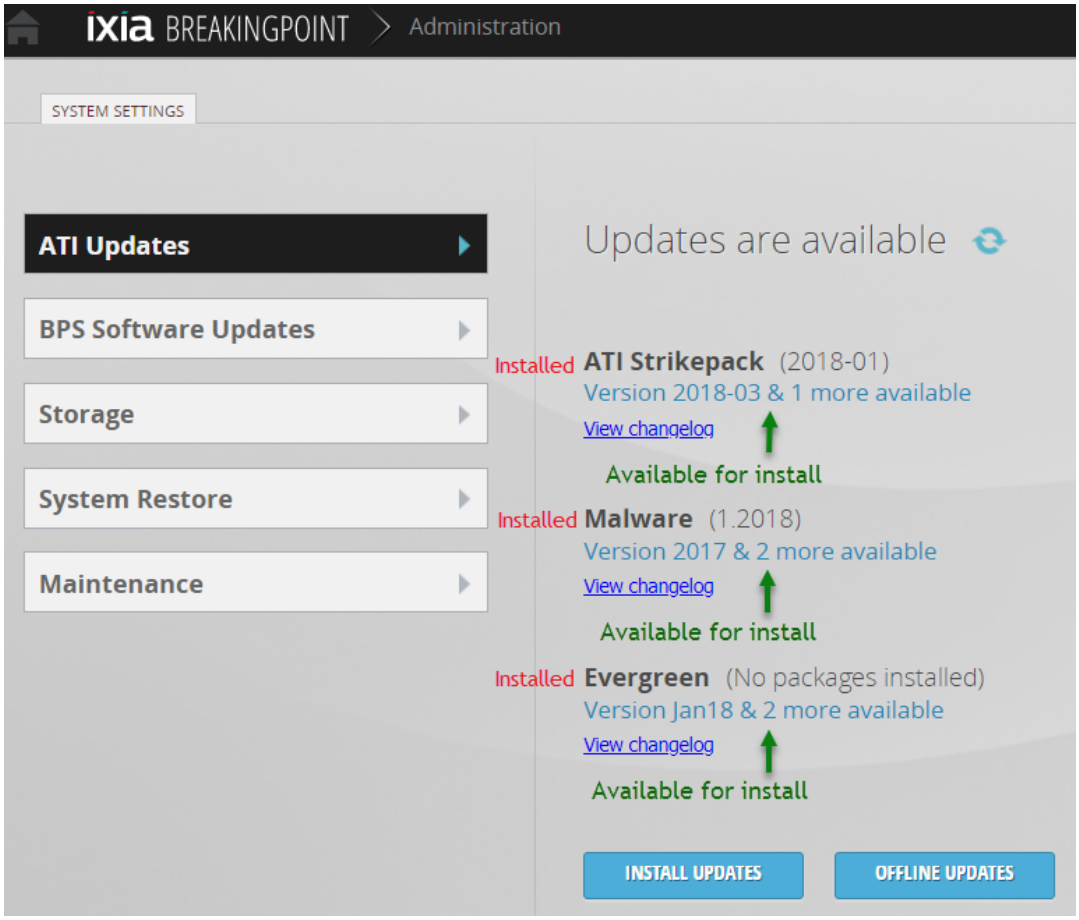
When BPS is connected to the internet, you will receive automatic notification of available ATI updates in the BPS UI as shown in the image below.

Click the "**Click to see more details**" link OR click the **Launch** button under the **Administration** section, then click **ATI Updates**. A page similar to the one shown in the image below will display.



The status of the updates that are currently installed, and the updates that are available for installation are displayed.

Click the **Refresh** button ( ) to refresh the page.

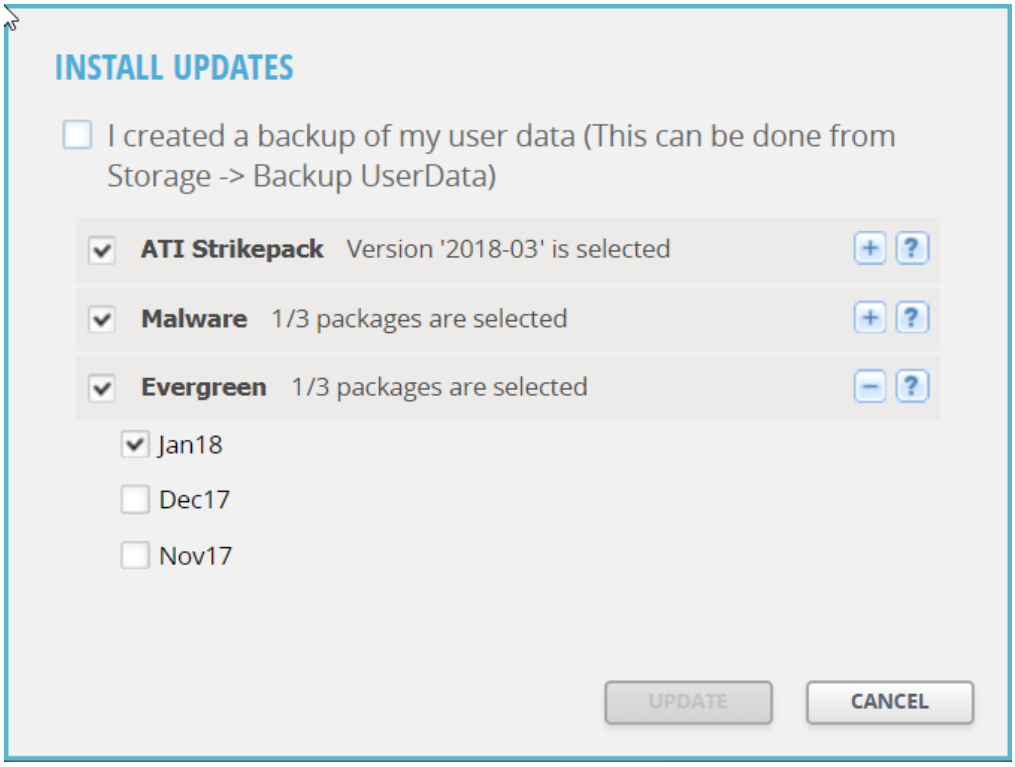


Each section of the window (**ATI Strikepack**, **Malware** and **Evergreen**) displays the following:

- The installed version.
- A description of the versions available for installation. Clicking this link will jump to the Install Updates dialog box. Based on the link you select, the corresponding section will automatically expand allowing you to see the available update details.
- A **View changelog** link. Click this ink to view the details about the contents of a package.

Install Updates Dialog Box

The Installed Updates dialog box displays after you click a "Version Available" link or the **Install Updates** button in the **ATI Updates** section of Administration.



A **Backup Confirmation** check box is available at the top of the dialog box. The **Update** button will not become available until this boxed is checked.

You can click the check box next to **ATI Strikepack**, **Malware** or **Evergreen** to select/deselect the options that are currently selected in these sections. For example, using the dialog box displayed above, deselecting the check box next to **Evergreen** will uncheck/disable the "jan18" update. Selecting the **Evergreen** check box will check/enable the "jan18" update.

The "+" and "-" buttons allow the sections to be expanded/collapsed respectively. The "?" button jumps to the changelog view.

To install ATI Updates:

1. Select the **Backup Confirmation** check box to confirm that a system backup has been performed.
2. Select the ATI Strikepack that you want to install.
3. Select the Malware and Evergreen packages that you want to install.
4. Click the **Install Updates** button.

Offline Updates

ATI Update Strikelists and Packages can be obtained from the Ixia website at: https://support.ixiacom.com > **Software Downloads** > **BreakingPoint Software**. Note that you will need to log in to access this section of the support website.

To update an ATI Package:

1. Log in with an admin account.
2. Click the **BreakingPoint** application icon.
3. Click the **Launch** button under the **Administration** section.
4. Click **ATI Updates**.
5. Click **Offline Updates**.
6. Select the box indicating that a system backup has been performed.
7. Browse to select a **Local file**, indicate a **Web Location** or **Server Location** for the update packages. Server Location is used in a scenario where the firmware successfully uploaded to the chassis but the installation failed for some reason (for example, licensing issues). The firmware can be installed from the chassis using this option.
8. Click **Update.**

TACACS+ Authentication

This section describes the Ixia Breaking Point System (BPS) support for TACACS+ (Terminal Access Controller Access-Control System Plus) authentication. TACACS+ is an access control network protocol for routers, network access servers and other networked computing devices.

TACACS+ Introduction

By default, BPS is configured in local authentication mode with one initial user, which is admin.

- The local BPS admin user is referred to as the default administrator and cannot be deleted.
- The local admin user account is accessible even when BPS is using TACACS+ authentication. This is done as a fail-safe in the event that the remote server is unreachable due to either a communication or misconfiguration error.
- When BPS is in TACACS+ mode, users that are not the local admin, must authenticate with the TACACS+ server before gaining access to the BPS UI.
- When TACACS+ Authentication mode is enabled, users cannot be added using the BPS user interface (UI). Users, and their access level, must be configured at the TACACS+ server. Reference your TACACS+ server documentation for configuration information.

Note: Changing the authentication mode from or to Local or TACACS+ requires a BreakingPoint System restart.

Supported Platforms

Currently TACACS+ authentication is supported on the FireStorm, PerfectStorm and BreakingPoint Virtual Edition BPS platforms.

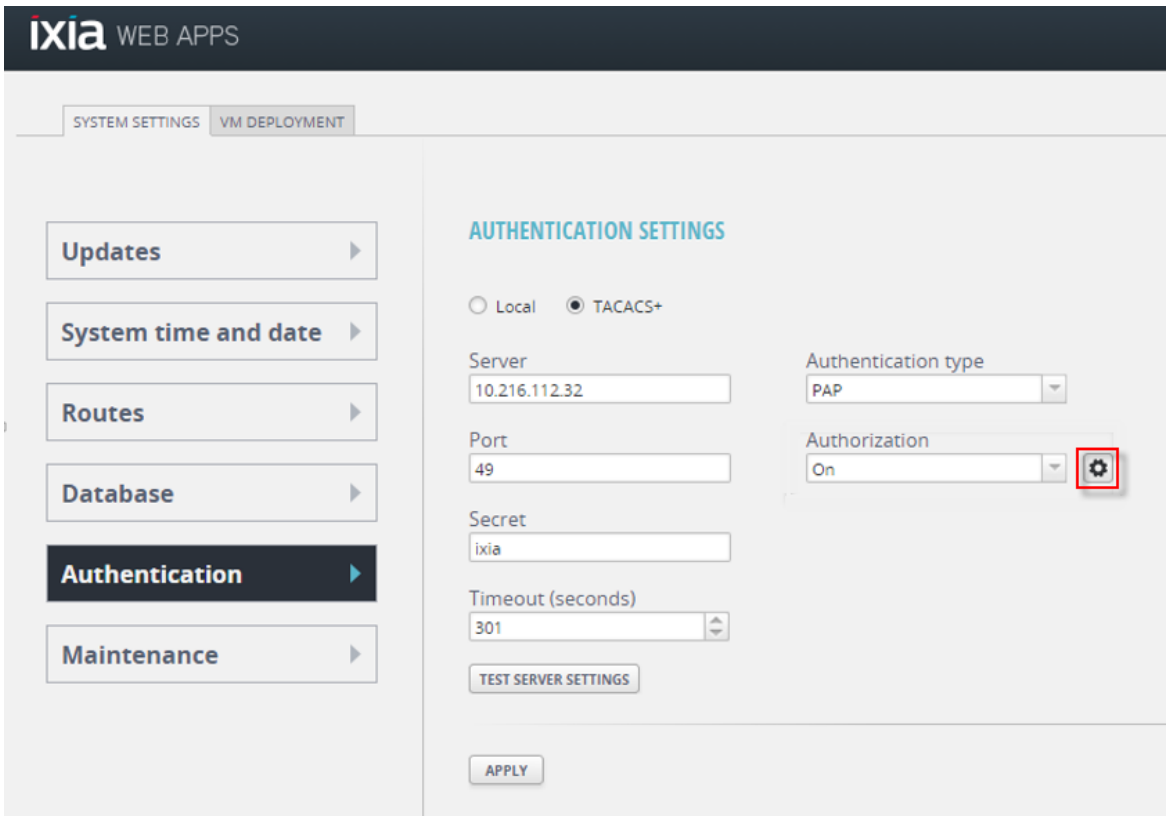
Authentication Settings

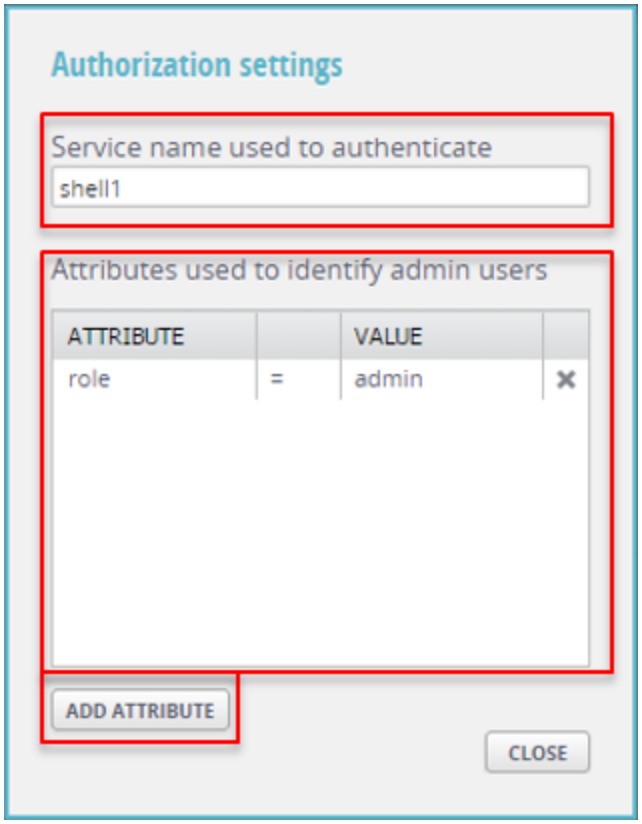
To support TACACS+ authentication, an Authentication tab has been added to the BPS user interface as shown in the image below.

The Authentication tab allows a local admin user to configure the TACACS+ authentication and authorization fields.

Note: In the first phase of Authentication support, BPS will not support TACACS+ accounting functionality.

Note: Remote authentication must be enabled on both BreakingPoint and on the remote server. Reference your TACACS+ server documentation for information on configuring and enabling your TACACS+ server. For reference, RFC 1492 (http://www.faqs.org/rfcs/rfc1492.html) describes TACACS+.





The following list provides a description of the fields shown in the images above.

Authentication Option	Description
Authentication Settings	Select Local for the local BPS authentication mode. Select TACACS+ to enable TACACS+ authentication mode. **Note:** Only the Admin user can modify this setting. See TACACS+ Privileges on the next page for more information.
Server	Enter the IP address or hostname of the TACACS+ server.
Port	Specify the TCP port the TACACS+ server is listening on.
Secret	The secret passphrase which BPS will use to encrypt the entire TACACS+ payload and send it to the TACACS+ server. The TACACS+ Server should also use the same passphrase to decrypt the payload. This passphrase will be shared with the server prior to sending the encrypted details.
Timeout	Specify the amount of time BPS will wait before timing out if it has not received a response from the TACACS+ server.

Authentication Option	Description
Authentication Type	Select the types of authentication mechanism used. **Note:** Currently, BPS only supports PAP and CHAP authentication.
Authorization	Specify whether there is a need to authorize users using a TACACS+ server. The options are Off and On. • **When authorization is Off**: BPS will not send authorization packets to the TACACS+ server and all of the users (TACACS+ users) are treated as Regular users regardless as to whether they are defined as having an admin role at the TACACS+ server. • **When authorization is On**: The following options must be configured in order for a user to authenticate as an admin with the TACACS+ server (click the "dial" icon at the right side of the Authorization field shown in the image above): Service Name: Indicate the service name required for an admin user to authenticate. Attribute Values: Indicate the attribute sets required to recognize admin users.
Test Server Settings	Select this option to trigger a TACACS+ server test. BPS tests the TACACS+ server settings by opening a TCP connection with the TACACS+ server using the Authentication Server and Port settings you have defined. A FIN termination request is sent after a successful server connection is made.
Apply	Click **Apply** to save and apply the configuration settings. **Note:** Applying changes will require a BPS restart.

TACACS+ Privileges

The following table describes the privileges that can be assigned to users.

User	Privilege Level
Admin	This special user is the only user that can access and modify the Authentication mode.
Local Admin Users	These users can access the Administration tab but they cannot change the Authentication mode.
TACACS+ Admin Users	These users can access and modify the Administration and TACACS+ Server Settings but they cannot change the Authentication mode.
TACACS+ Regular Users	These users cannot access the BPS UI Administration tab.
Local Regular Users	These users cannot access the BPS UI Administration tab.

System Functions

The Systems Functions area provides controls for managing licenses and exporting and importing multiple tests.

The System Functions table shown below lists and describes each available function.

System Functions

Function	Description
Export Tests	Exports all existing tests. During this process, do not close the browser that is processing this request. Be aware that exporting all existing tests may require a large amount of time to process.
Import Tests	Imports all existing tests. During this process, do not close the browser that is processing this request. Be aware that importing all existing tests may require a large amount of time to process.
Licensing *	Provides a view of the installed licenses and provides options to import new licenses.

Note: * Ixia BreakingPoint has added a licensing enforcement mechanism to its firmware. This mechanism allows all customers under current ATI maintenance to install new firmware and ATI updates, but prevents new updates from being installed once current maintenance agreements have expired. Should you encounter any difficulties with licensing during the upgrade process, contact BreakingPoint Support at 1-818-595-2599. If your ATI maintenance agreement has expired, contact your BreakingPoint sales representative.

System Logs

There are six logs that track the various events and errors that occur on the system:

- Audit
- Error
- Information
- Message
- System
- Web

The information listed in these logs are used for support related issues. Typically, when you make a support request, the BreakingPoint Systems support team will require that you send in the information stored in these logs. To send in a compressed file of these logs, go to the Start Page and click the Diagnostics button. Save the file called diagnostics-xxx.bug to a location on your computer and send that file to the support team.

The information stored in the logs are not intended to be deciphered. Any system messages intended for you will automatically display as popup messages.

My Preferences

The settings on the My Preferences tab enable you to configure global options that affect how Control Center functions.

My Preferences below lists and describes each available function.

My Preferences

Function	Description
Set data rate as L2	Determines whether or not the Bandwidth Tx/Rx statistics include Layer 1 data or not. If enabled, Bandwidth statistics do not include Layer 1 data. If disabled, Bandwidth statistics include Layer 1 data.

CHAPTER 7 Device Under Test Profiles

This section covers:

DUT Profiles

A DUT Profile defines the connection settings for the device under test – such as the connection method, connection parameters, interface speed, and global scripts. BreakingPoint will use these settings to establish a connection to the device under test (DUT) for automation purposes.

Note: Each test must have a DUT Profile selected for it; however, if you do not plan on using device automation, you can select the default BreakingPoint Systems DUT Profile.

Valid Connection Parameters

Connection Parameters on the next page lists the valid connection parameters for serial, SNMP, SSH, and telnet connection types.

Connection Parameters

Connection Type	Parameter	Valid Values
Serial	Flow Control	none, rtscs, or xonxoff
	Speed	50, 75, 110, 134, 150, 200, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
	Parity	'n' for none, '0' for even, or '1' for odd
	Data Bits	7 or 8
	Stop Bits	0 or 1
SNMP	Host	Server IP address
	Port	SNMP port (default 161)
	Version	1, 2, or 3
	Login ID	Server login ID
	Password	Server password
SSH	Host	Server IP address
	Port	SSH port (default 22)
	Login ID	Server login ID
	Local IP	BPS Management Port IP address (default mgmt)
Telnet	Host	Server IP address
	Port	Telnet port (default 23)
	Local IP	BPS Management Port IP address (default mgmt)

Note: For telnet, if the local IP is set to mgmt, BreakingPoint will communicate with the device under test using its management interface. However, if the local IP is set to an IP address, then BreakingPoint will use the DUT's control interface to communicate with the DUT. In the latter case, you must connect an Ethernet cable between the BPS management port to the DUT, otherwise, the BPS will not have a way to locate a route to the DUT.

Note: The Ixia BreakingPoint PerfectStorm does not accept incoming Telnet connections. Ixia recommends using SSH to establish an external connection to the PerfectStorm.

Creating a DUT Profile

A DUT Profile defines the device's connection type, connection parameters, interface speeds, and global scripts. BreakingPoint will use these settings to establish a connection to the DUT for automation, or scripting purposes. You can use Expect scripts (also known as global scripts) to automate your device testing; for example, you can create scripts that will create VLANs.

See Task 3: Creating a Device Under Test Profile on page 54 for detailed information.

Global Scripts

Global scripts are also known as Expect scripts. These global scripts allow you do things like reboot your device, monitor DUT statistics, and create VLANs via software control. BreakingPoint Systems provides templates which you can use as a basis for your own scripts.

Each device type comes with a set of templates that are specific to that device. For a list of templates, see the Global Scripts Templates on page 1469

When creating global scripts, keep the following factors in mind:

- Each global script is specific to the device selection. For example, if you create a global script for the Cisco IOS device type, then only the DUT Profiles using the Cisco IOS device selection can access the global script.
- There must be a serial or Ethernet connection between the Target Control port and the DUT. For more information on Target Control ports, see the BreakingPoint system Installation Guide.
- Each line in the global script must begin with a command (i.e., expect, send, expect-close, etc.).
- You must click **Save** to save any changes you have made to a global script.
- Only one global script can use the On Start option per DUT Profile.

Commands

Commands below lists the commands that can be used for creating scripts.

Commands

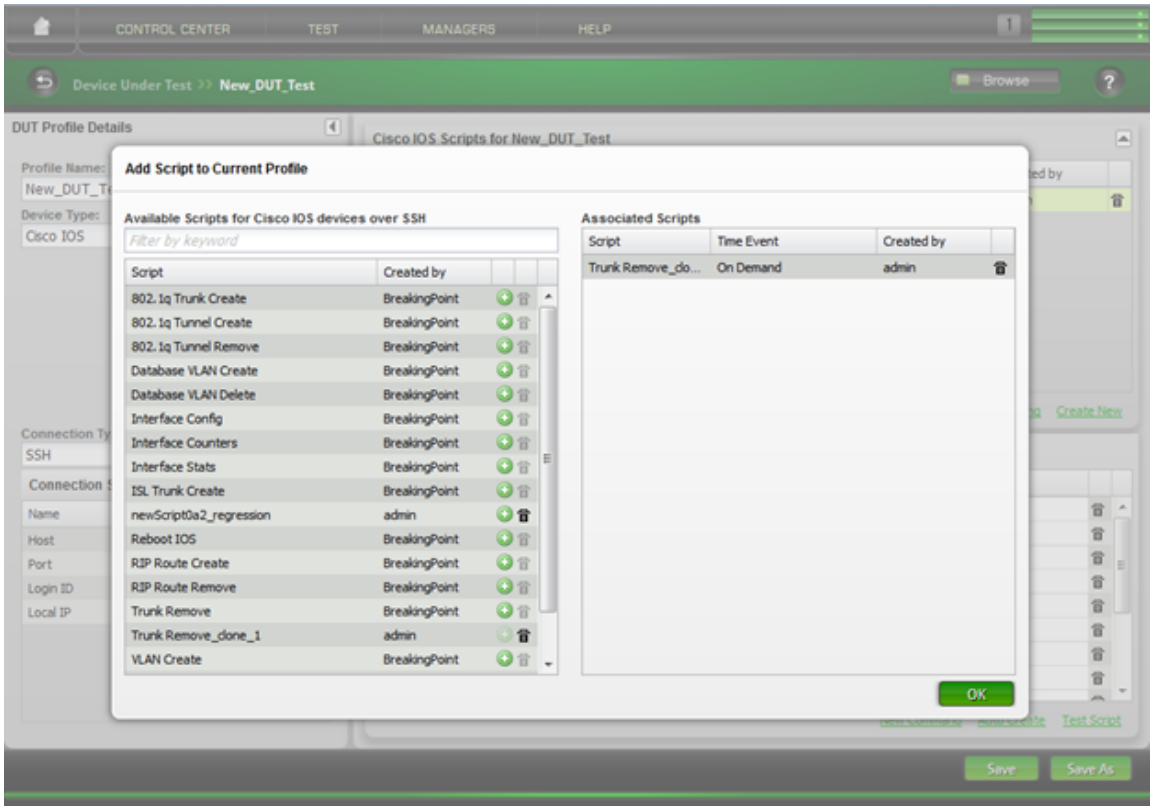
Commands	Description
Expect	Waits for a string from a process (e.g., Expect "name")
Send	Takes a string and sends it to a process (e.g., Send "myLoginID"\r).
Expect-Close	Waits for the server to close the connection.
Wait	Delays the script from executing for n milliseconds.
Power Cycle	Restarts the device.

Using an Existing Global Script

To use an existing global script:

1. Click **Add Existing** under the Scripts area (displayed at: **Control Center** > **Device Under Test** - **Open** or **Create New** DUT).
2. Click the **Add Script to Profile** icon for any script listed in left pane of the window. Please note that this list is filtered based on **Device Type** and **Connection Type** settings.

Note: Any changes made to the Script Commands will be applied to the Global Script across all profiles that are using it.



Creating a Global Script

Creating a global command requires familiarity with the syntax of Expect; however, if you are unfamiliar with Expect, a script can be created by recording commands from the command line. For more information on recording a script, see the section Auto Creating Global Scripts on the facing page.

To create a global script:

1. Select **Control Center > Device Under Test** from the Menu bar.
   a. Select a DUT Profile from the Profile Name list and click **Open**, or double-click the profile.
2. Click **Create New** under the Scripts area.
3. Enter a name for the script in the **Name** field.
4. Click **OK**. The newly created Global Script will be automatically added to the current DUT Profile.

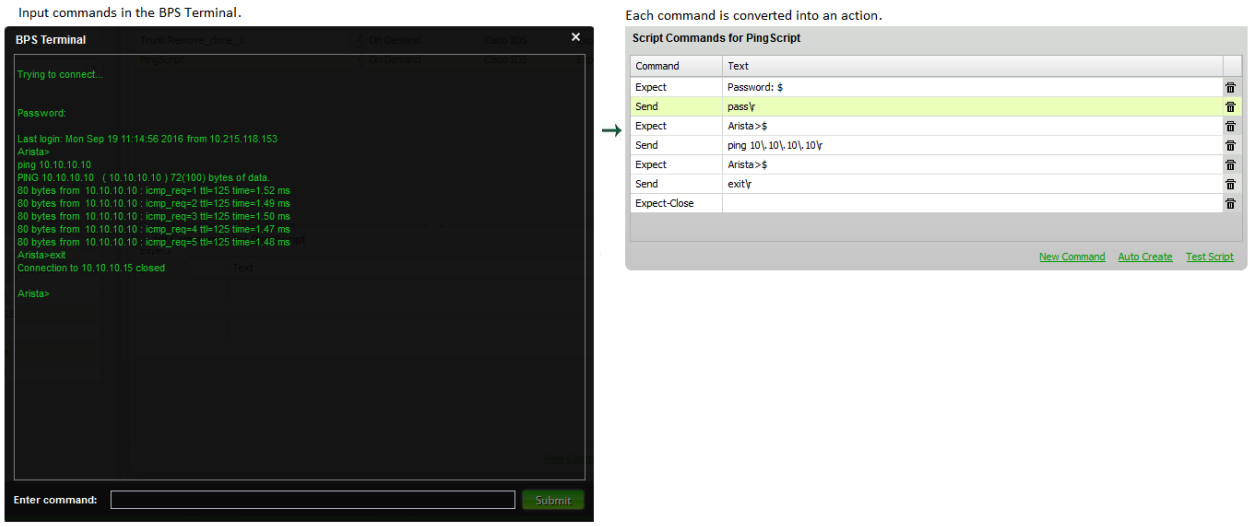
5. Click **New Command** under the Script Commands area.
6. Select a command from the Command column (see Commands for reference).
7. In the Text field that is to the right of the selected command, type parameters for the command if they are required.
8. Repeat steps 5-7 until all desired commands have been added.
9. Specify when the Global Script will run by clicking the **Edit Time Event** icon  located to the right of the script name, and then selecting one of the following options:
   a. Select **When Test Starts** to run the script when the test starts.
   b. Select **When Test Stops** to run the script once the test finishes.
   c. Select **On Demand** to run the script from the Real Time Statistics page at any time while the test is running.
   d. Select **Periodically After Start** to run the script at certain time intervals and enter the time interval at which the script should be executed. Use the following format: hours: minutes: seconds.
   e. Select **After Test Started** to run the script after the test has started and enter the amount of time the test should run before the script is executed.
10. Click **Save** or **Save as** to save the script.
11. To test the script, select the script from the Global Scripts list and click the **Test Script** button.

Auto Creating Global Scripts

Global scripts can be recorded from the command line. Each command entered in the text console is recorded by the system and translated into an Expect string.

For example, let's say the destination 10.10.10.10 should be pinged at the start of a test. Using the BPS Terminal, we can ping the destination and record the commands that have been entered into the text console. After we exit the BPS Terminal, the system will translate our commands into Expect scripts, as shown in BPS Terminal below.

BPS Terminal



To auto create a global script:

1. Select **Control Center > Device Under Test** from the Menu bar.
   a. Select a DUT Profile from the Profile Name list and click **Open**, or double-click the profile.
2. Select an existing script from the Global Scripts list or create a new one.

Note: The next step requires running **Auto Create** on a selected script. Be aware that this action will clear the script's existing script commands before any Auto Create commands are recorded.

3. Click **Auto Create**. The console will be displayed.
4. Type and **Submit** commands in the console.
5. **Submit** an `exit` command in the console and close the BPS Terminal when you are finished. Your commands will appear as actions in the Script Commands list.
6. Specify when the Global Script will run by clicking the **Edit Time Event** icon located to the right of the script name, and then selecting one of the following options:
   a. Select **When Test Starts** to run the script when the test starts.
   b. Select **When Test Stops** to run the script once the test finishes.
   c. Select **On Demand** to run the script from the Real Time Statistics page at any time while the test is running.
   d. Select **Periodically After Start** to run the script at certain time intervals and enter the time interval at which the script should be executed. Use the following format: hours: minutes: seconds.
   e. Select **After Test Started** to run the script after the test has started and enter the amount of time the test should run before the script is executed.
7. Click **Save** or **Save as** to save the script.
8. To test the script, select the script from the Global Scripts list and click the **Test Script** button.

CHAPTER 8 Network Neighborhood

This section covers:

Network Neighborhood Overview

The Network Neighborhood defines the possible addresses the system can use for its generated test traffic and determines how the system will allocate those addresses for use. All addresses used in test traffic generated by BreakingPoint must follow the protocol rules as though the addresses were a real host existing within a real subnet on the network.

What is a Network Neighborhood?

A Network Neighborhood defines the addressing rules that are available for each test interface. You designate addressing information for each test interface by creating component tags. Each component tag defines the host addresses that can be used in the test traffic, as well as the subnet and routing information for those hosts. The addressing will fill the entire subnet, but you can limit the number of addresses by defining a range for the network.

When you create a test, you will assign a component tag to each test interface used by a test component. For each test component, the component tag assigned will determine the client addressing and server addressing. When the system generates the test traffic, it will derive the source and destination addresses from the component tag. Component tags are covered in detail later in this chapter.

How Does the Network Neighborhood Work?

To simplify this feature, think of the Network Neighborhood as a large pool of addresses and interfaces. Each component tag breaks down the Network Neighborhood into smaller pools. Each component tag has at least one subnet that sets the guidelines for the addresses that can be in that pool. You can further restrict the number of addresses within the subnet by assigning a range.

The Network Neighborhood determines:

- The type of network in which the device is operating (e.g., routed, switched, NAT, or VLAN).
- The addresses that can be used for the Ethernet, source, and destination IP addresses.

After the system looks at the Network Neighborhood you have selected for the test, it will look at the component tags that are selected for each interface. These component tags contain the subnets that the system will use to derive its addressing.

How Do I Set Up a Network Neighborhood?

There are a few decisions you must make before creating a Network Neighborhood. These decisions will help you determine what values you should define for each parameter. For a list of parameters, see the section.

First, decide what type of network you want to create (e.g., routed, switched, NAT); this will affect the subnet definition.

Next, determine the number of host addresses you need. This will determine whether you select a virtual router or host.

Finally, figure out which test components you will be using. The type of test component you use will determine the type of network (e.g., switched, routed, etc.) you will need to create.

Note: The following video link provides access to a tutorial on **Configuring a Network Neighborhood**:
https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

Network Neighborhood Control Buttons

provides descriptions of each control button on the Network Neighborhood screen. You can use these buttons to navigate the Network Neighborhood screen.

Network Neighborhood Control Buttons

Button Name	Description
Return Button (arrow)	Returns you to the previous screen
Browse	Browse all available Network Neighborhoods.
Help (?)	Takes you to the help system.
Entry Mode	Displays the editable console that allows you to configure the network to be used in your test.
Diagram Mode	Displays a diagram of the network as configured.
Add New Element	Allows you to select a network element to add to the network configuration.
Expand All | Collapse All	Allows you to expand or collapse the details of all visible interfaces.

Button Name	Description
Keyboard Shortcuts	Displays available keyboard shortcuts.
Add Row	Add a new interface to the Network Neighborhood.
Revert	Revert to last saved state.
Test Paths	Manually define the client/server interface connections.
Save	Saves your test.
Save As	Saves your test under a different name without running it.
Close	Closes the screen.

Network Neighborhood Elements

The Network Neighborhood page is comprised of Elements. Elements are similar to various types of network devices.

Each element corresponds to a port on the BreakingPoint system. As you add elements to your Network Neighborhood, you can configure them to have the same or different network settings as the other elements that share the same interface.

As you add and view Network Elements in the user interface, note the following color assigments of Network Element labels.

Network Element	Label Color
IP Infrastructure	White
IP Configuration	Olive
Endpoint	Black
LTE	Copper
Mobile Configuration	Tan
3G	Light Blue

Maximum Number of Hosts and Virtual Routers

PerfectStorm does not limit the number of static hosts, virtual router entities, or the IP address count. Instead there is a design limit of 1024 internal rules. PerfectStorm uses an internal router so that received packets are sent to the correct network processor, and the maximum number of rules is 1024. This means that the maximum number of tags (per card) is 1024. Furthermore, a unique rule is created for each test component. For example, if you have a network neighborhood with 20 tags, and two components each using all 20 tags, the result will be 40 rules.

Firestorm supports 64 IPv6 or 256 IPv4 ranges per VLAN.

IP Address Allocation Options

The following test components allow the user to select an IP allocation algorithm.

- Application Simulator
- Client Simulator
- Recreate
- Malware
- Stack Scrambler

The following test components use a component IP selection algorithm.

- Routing Robot
- Bit Blaster

User Specified IP Allocation Algorithm

IPv4 and IPv6 static hosts have the option of using a Random or Sequential IP address generation algorithm. When IP addresses are generated the following rules apply:

- IP addresses will be assigned in incrementally starting from min IP to max IP as defined in the network neighborhood
  - IP addresses will wrap around within the defined range
- Source and destination IP addresses will be independent of each other

Component IP Selection Algorithm

The BPS traffic engine selects an IP address randomly as described in the steps below.

1. A new Super Flow session is created.
2. For each flow in a Super Flow, a new tuple (an ordered set of values) is allocated based on the configuration.
3. IP addresses are allocated. For example, for an HTTP flow:
   a. A client IP address is randomly picked from the configured range.
   b. A server IP address is randomly picked from the configured range.
   c. The server port is set to 80 (HTTP).
   d. The client port is picked at random from the configured range.
   e. An attempt is made to add the tuple to the flow table. If it is already in use, a different IP address is selected. If a unique tuple cannot be found after 100 tries, the Super Flow session is deleted and BPS will try again later.
4. Repeat step 2 for all flows in the Super Flow.

This process ensures that all tuples are allocated before any packets are sent.

MPLS Functionality

This section of the user guide provides background information on the MPLS feature. MPLS-Settings are a user defined group of MPLS tags (see MPLS Parameters). MPLS-Settings are also a unit, meaning that any TCP Connection using the settings will have all of the tags configured (in order) in each of the packets that are sent from a specific source IP throughout a single session. Notice that BPS associates MPLS tags with source IP addresses. On the receive side, BPS does not validate or take action on MPLS tags. When a host has more than one MPLS-Settings list assigned to it, the lists are used in a round-robin fashion.

- A MPLS Settings List is defined with MPLS tags
- MPLS Settings Lists can then be applied to IPv4 Static Hosts and IPv6 Static Hosts

To create a MPLS Settings List:

1. In Network Neighborhood, select **Add New Element** > **IP Infrastructure** > **MPLS**. A MPLS network element will appear in a MPLS section of Network Neighborhood.
2. In the **ID** field, you can name the MPLS Settings List or accept the default name (mpls_settings #) by performing the next step.
3. Within the new MPLS network element, click the **MPLS Tag(s)**.. link.
4. Configure the values for the **Label**, **Experimental Field** and **TTL** fields.
5. Click **Add New Tag** if you wish to add and configure additional tags.
6. Click **Accept** to save the MPLS Settings List.

If you want to add additional MPLS network elements, click the **Add Row** button (which is above the **MPLS** section).

To apply a MPLS Settings List to a IPv4 Static Host or IPv6 Static Host:

1. Click the Static Host's **MPLS Settings List** link.
2. Click the drop-down list to select a MPLS Settings List.
3. Click **Add New** if you wish to add additional MPLS Settings Lists to the configuration.
4. Click **Apply** to save the configuration.

MPLS Known Limitations

The following components are not supported in a Network Neighborhood containing MPLS-Settings:

- Routing Robot
- Bit Blaster (including the RFC2544 Lab test)
- Security
- Stack Scrambler

Note the following functionality and limitations:

- Tests containing MPLS-Settings are not allowed to run on Storm/Firestorm blades.
- The Recreate Test component can capture files having MPLS-Tags only when the Mode Option -
- Replay Capture File Without Modification is set.
- MPLS-Settings and IP Mappings: Static mapping of specific MPLS-Settings and a specific IP

- Source (and destination) holds true for a single session. A combination of IP/MPLS-Settings can be statically mapped to different MPLS-Settings/IP for a different session.
- Number of supported settings:
  - Only the first 8 MPLS-Tags that are defined in a MPL-Settings will be used.
  - Only the first 128 MPLS-Settings will be used per IPv4/IPv6 host from configured SettingsLists.
  - Only the first 256 MPLS-Settings will be used globally/overall.
- A MPLS-Settings name cannot contain a semicolon ";".
- A blank in a MPLS tag field will be interpreted as "0".
- If the TTL field of a MPLS tag is configured as "0", the effective value will be "64".
- Tags configured in MPLS_Settings need to be "Accepted" (equivalent to being saved) in order to take effect.

Element Groups

Each Element belongs to an Element Group. This section describes each Element Group and lists the Elements that belong to them. Each Element parameter is also described.

IP Infrastructure Element Group

The elements in this group define the layout and structure of the simulated network.

IP Infrastructure Elements

- Interface
- VLAN
- MPLS
- IPv4 DHCP Server
- IPv4 Router
- IPsec IPv4 Router
- IPv6 DHCP Server Parameters
- DHCPv6 Server Statistics
- DS-Lite B4 Parameters
- DS-Lite AFTR Parameters

Interface Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Number	Interface number
MTU	Maximum Transmission Unit (IP)
MAC Address	Base 48-bit hexadecimal MAC address.

Parameter	Description
Duplicate MAC Address	Select to use one MAC address for all hosts.
VLAN Key	Determines whether to route packets based on inner or outer VLAN in double-tagged VLAN scenarios.
Ignore Pause Frames	Disregard received Ethernet pause frames.
Description	User-defined description of the network element. The contents of this field are included when searching, so it can be used as a way to enhance the ability to find specific Network Neighborhoods.
Impairments	Corruptions applied to outbound packets.
Packet Filter	Capture filter rules applied to inbound packets.

VLAN Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Inner VLAN ID	Defines the inner VLAN ID for frames. (For Q-in-Q routing)
Outer VLAN ID	Defines the outer VLAN ID for frames. (For Q-in-Q routing)
Tag Protocol Identifier	The outer VLAN tag will use this TPID for outbound packets. Inbound packets can match any supported TPID.
MTU	Maximum Transmission Unit (IP)
MAC Address	Base 48-bit hexadecimal MAC address.
Duplicate MAC Address	Select to use one MAC address for all hosts.

Parameter	Description
Description	User-defined description of the network element. The contents of this field are included when searching, so it can be used as a way to enhance the ability to find specific Network Neighborhoods.

MPLS Parameters

Parameter	Description
ID	Name of the MPLS-Settings. The default naming structure is, **mpls_settings** #.
MPLS Tags	Configure between 1-8 MPLS tags per MPLS-settings. Each MPLS tag can be configured with a **Label**, **Experimental Field** (defines QOS treatment), and **TTL**.

IPv4 DHCP Server Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Base Lease IP Address	Defines the minimum IP lease address.
Count	The number of hosts in the set.
Lease Time	The number of seconds the DHCP server will advertise an address it gives to a client until the client has to renew it.
Accept Simulated Clients Only	Instructs the DHCP server to only offer leases to BreakingPoint clients.
DNS Settings	A selection from the DNS Settings network element list.
IP Address	Defines the router IP address.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.

Note: The following options are only available on the PerfectStorm Platform.

- IPv6 DHCP Server Parameters
- DHCPv6 Server Statistics

IPv6 DHCP Server Parameters below (PerfectStorm Platform Only) provides descriptions of the parameters available in IIPv6 DHCP Server Parameters. You can use these parameters to define the network settings for each IPv6 DHCP Server element in your test. DHCPv6 Server Statistics on the next page (PerfectStorm Platform Only) provides a description of statistics provided for a DHCPv6 Server.

In addition to the DHCPv6 server parameters described below, please be aware of the following:

- The DHCPv6 Server element has to have an associated IPv6 host element (and only one), connected to the same interface. This implies that the DHCPv6 Server IPv6 address has to be an address within an address range of an IPv6 Host element.
- When using VLAN, both the DHCPv6 Server element and its associated IPv6 host element have to be placed in the same VLAN container.
- The default gateway for the DHCPv6 Server has to be the same as the default gateway of its associated IPv6 host element.
- Multiple DHCPv6 severs cannot be directly connected to the same interface. However, multiple DHCPv6 serves can be directly connected to the same interface, if each server is included in a different VLAN container.

IPv6 DHCP Server Parameters

Parameter	Description
ID	The ID of the logical interface.
IPv6 Address	Defines the DHCP server IPv6 address.
Container	The device or network that this element resides behind or within.
Prefix Length	Defines the prefix length associated with the DHCP server interface.
Gateway IPv6 Address	Defines the default gateway address associated with the DHCP server interface.
Pool Base IPv6 Address	For DHCPv6 with IA Type: **IANA/IATA** - Defines the first IPv6 address leased by the server (the least significant bit of the host address is incremented by 1) **IAPD** - Defines the first IPv6 prefix advertised by the server (the least significant bit of the network address is incremented by 1)
Pool Address Size	Number of leased addresses.

Parameter	Description
Pool Prefix Length	The pool prefix length advertised by the DHCP server.
First Advertised DNS	The first DNS server advertised by the DHCP server.
Second Advertised DNS	The second DNS server advertised by the DHCP server.
IA Type	The identity association type supported by the IPv6 address pools. Available options: IANA, IATA and IAPD.
Default Lease Time	The duration of an address lease, in seconds, if the client requesting the lease does not ask for a specific expiration time. The default value is 86,400 (one day); the minimum is 300; and the maximum is 30,000,000.
Maximum Lease Time	The maximum lease duration (in seconds). The default value is 86,400 (one day), the minimum is 300, and the maximum is 30,000,000.
Offer Lifetime	The lifetime duration (in seconds) that is assigned to a lease after the Advertise message is sent. The lease will be freed after this time unless the server receives a Request message for it. The default value is 10, the minimum is 5, and the maximum is 60.

DHCPv6 Server Statistics

Statistic Name	Description
Solicits Received	Number of SOLICIT messages received from DHCP clients.
Advertisements Sent	Number of ADVERTISE messages sent by this DHCP Server, in response to SOLICIT messages received from DHCP clients.
Requests Received	Number of REQUEST messages received from DHCP clients.
Confirms Received	Number of CONFIRM messages received from DHCP clients to determine whether the IPv6 addresses it was assigned are still valid.
Renewals Received	Number of RENEW messages received from DHCP clients to extend the lifetimes on the assigned addresses and/or to update other configuration parameters.

Statistic Name	Description
Rebinds Received	Number of REBIND messages received from DHCP clients to extend the lifetimes on the assigned addresses and/or to update other configuration parameters, in the event that the client did not receive a response to a RENEW message.
Replies Sent	Number of REPLY messages sent by this DHCP Server in response to SOLICIT, REQUEST, RENEW, REBIND, INFORMATION-REQUEST, CONFIRM, RELEASE, and DECLINE messages received from DHCP clients.
Releases Received	Number of RELEASE messages received from DHCP clients.
Declines Received	Number of DECLINE messages received from DHCP clients.
Information-Requests Received	Number of INFORMATION-REQUEST messages received from DHCP clients.
Relay Forwards Received	Number of RELAY-FORW messages received from relay agents.
Relay Replies Sent	Number of RELAY-REPLY messages sent to relay agents.
Total Addresses Allocated	The total number of IPv6 address leases that have been renewed by this server.
Total Addresses Renewed	The total number of IPv6 address leases that have been renewed by this server.
Current Addresses Allocated	The number of IPv6 addresses that this server is currently leasing to clients.
Total Prefixes Allocated	The total number of IPv6 prefixes that have been allocated by this server.
Total Prefixes Renewed	The total number of IPv6 prefixes leases that have been renewed by this server.
Current Prefixes Allocated	The number of IPv6 prefixes that this server is currently leasing to clients.

IPv4 Router Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	Defines the router IP address.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.

IPsec IPv4 Router Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within. Supported container types: IPv4 Router, Interface.
IP Address	Defines the router IP address. This is the outer IP from the IPsec header. It will be incremented with 0.0.0.1 for each IPv4 Static Host count. In other words, for each IPv4 host IP there will be a different tunnel/outer IP.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
IKE Peer Address	Defines the IKE Peer IP Address. Note that If the "1-to-1 IKE peers" check box from the IPsec Configuration element is enabled, this value will be incremented by 0.0.0.1. If the "1-to-1 IKE peers" checkbox is disabled, the defined IP Address will remain the same for all tunnels.
Config	IPsec configuration.

IPv6 Router Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.

Parameter	Description
IPv6 Address	Defines the router IPv6 address.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Prefix Length	The length (in number of bits) of the service provider's 6rd IPv6 prefix set in the subnet mask.
DHCP-PD Client Config	Container for a DHCPv6 Client Configuration element with IAPD IA Type configured. If specified, the router acts as a DHCP client and it is requesting a prefix to be assigned. The hosts behind this router (subscriber networks) which will use PD need to have the IPv6 address allocation mode set to DHCPv6 and no Alloc config referenced.

The following options are only available on the PerfectStorm Platform.

- DS-Lite B4 Parameters
- DS-Lite AFTR Parameters

DS-Lite B4 Parameters below (PerfectStorm Only) provides descriptions of the parameters available in DS-Lite B4 elements. You can use these parameters to define the network settings for each DS-Lite B4 element in your test. In addition to the DS-Lite parameter descriptions below, please be aware of the following:

- When emulating multiple AFTRs per IP range, the AFTR Count is limited to 1 for the AFTR.
- Three DS-Lite Templates are available:
- DS-Lite-simulated-AFTR
- DS-Lite-simulated-B4
- DS-Lite-simulated-B4-AFTR-Loopback

The following items are not supported with this feature:

- RFC2544 L2-3 test component
- BitBlaster L2-3 test component
- RoutingRobot L2-3 test component
- Emulated Router

DS-Lite B4 Parameters

Parameter	Description
ID	The ID of the logical interface.
Container	The device or network that this element resides behind or within.
B4 Count	Defines the number of B4 devices.

Parameter	Description
B4 IPv6 Address Allocation Mode	Defines the method that is used to allocate the IPv6 address. Note that "Static" indicates that the starting IPv6 address will be the address set in the B4 IPv6 Base Address field.
B4 IPv6 Base Address	The IPv6 address of the first B4. This address will be incremented by 1 for each B4 Count.
B4 Gateway IPv6 Address	Defines the IPv6 gateway address for the B4.
B4 Prefix Length	Defines the prefix length of the B4 IPv6 address (in bits).
AFTR IPv6 Base Address	Defines the IPv6 address of the first AFTR device. If AFTR Count is greater than 1, this address will be incremented by 1 for each Count.
AFTR Count	Defines the number of AFTR devices in the topology. The implicit increment for the AFTR IPv6 Base Address is 1 for each count.
Host IPv4 Address Allocation Mode	Options include: **Static:** The starting IPv4 address will be the Base IP Address configured in the IPv4 STATIC HOSTS Element which has a Container field value that matches this DS-Lite B4's ID. The address Count and Netmask will also be derived from the IPv4 STATIC HOSTS Element. **Static/CPE Defined:** The starting IPv4 address will be the Host IPv4 Base Address value defined in this DS-Lite B4 Element, however, the address Count and Netmask will be derived from the IPv4 STATIC HOSTS Element which has a Container field value that matches this DS-Lite B4's ID
Host IPv4 Base Address	Defines the method that is used to allocate the IPv6 address. "Static":
Subnet IPv4 Increment	This field is only configurable if the Host IPv4 Address Allocation Mode is set to "Static/CPE defined" (otherwise the field will be grayed out). This increment is only used if the B4 Count value is greater than 1. **Example:** B4 Count value = 2, Host IPv4 Base Address = 192.168.0.10, Subnet IPv4 Increment = 0.0.2.0, Count Value (from IPv4 Static Hosts Element range which has a Container that match this DS-LITE B4's ID) = 7. **Result:** The 1st B4 will have hosts 192.168.2.10 – 192.168.0.16. The 2nd B4 will have hosts 192.168.2.10 – 192.168.2.16.

DS-Lite AFTR Parameters below (PerfectStorm Only) provides descriptions of the parameters available in IPv6 Router elements. You can use these parameters to define the network settings for each IPv6 Router element in your test.

DS-Lite AFTR Parameters

Parameter	Description
ID	The ID of the logical interface.
Container	The device or network that this element resides behind or within.
AFTR Count	Number of DS-Lite AFTRs. BPS can simulate one AFTR per IP range.
AFTR IPv6 address allocation mode	Defines the method that is used to allocate the IPv6 address. Note that "Static" indicates that the starting IPv6 address will be the address set in the AFTR IPv6 Base Address field.
AFTR IPv6 Base Address	Defines the IPv6 gateway address for the AFTR.
B4 IPv6 Base Address	Defines the IPv6 address of the B4 interface.
B4 Count	Number of DS-Lite B4s.
AFTR Prefix Length	Defines the prefix length of the AFTR IPv6 address (in bits).

IP Configuration Element Group

The elements in this group define common configurations that can be shared across multiple endpoint elements.

IP Configuration Elements

- IPv4 DNS Configuration
- IPv6 DNS Configuration
- IPsec Configuration
- IPv6 SLAAC Client Configuration (PS One only)
- DHCPv6 Client Configuration (PS One only)
- DHCPv6 Request Options (PS One only)
- DHCPv6 Timeout and Retransmission (PS One only)

The IPv4 DNS Configuration Parameters table shown below provides descriptions of the parameters available in IPv4 DNS Configuration elements. You can use these parameters to define the network settings for each IPv4 DNS Configuration element in your test.

IPv4 DNS Configuration Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
DNS Server	The address of the DNS to use when resolving hostnames.
DNS Domain	The default DNS domain used in domain name resolution.

The IPv6 DNS Configuration Parameters table shown below provides descriptions of the parameters available in IPv6 DNS Configuration elements. You can use these parameters to define the network settings for each IPv6 DNS Configuration element in your test.

IPv6 DNS Configuration Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
DNS Server	The address of the DNS to use when resolving hostnames.
DNS Domain	The default DNS domain used in domain name resolution.

The IPsec Configuration Parameters table shown below provides descriptions of the parameters available in IPsec Configuration elements. You can use these parameters to define the network settings for each IPsec Configuration element in your test.

IPsec Configuration Parameters

Parameter	Description
ID	The ID of the logical interface.
IKE version	Defines IKE version to use for running the test (IKEv1 or IKEv2).
IKE mode	Defines the IKE mode for IKEv1: main mode or aggressive mode. This field must be empty when IKEv2 is used.
1-to-1 IKE peers	When enabled, remote IKE (outer) IP addresses are incremented with .1 for each tunnel.
PSK	Pre-shared key: any text string up to 255 bytes long.

Parameter	Description
Left ID	Identification (left side) "Identification Type - Local ID Default value if field is blank: Local IKE peer IP Address (ID Type ID_IPV4_ADDR) Existent ID Types: - ID_IPV4_ADDR: specifies a single four (4) octet IPv4 address - ID_FQDN: specifies a fully-qualified username string (i.e. @foo.bar.com) - ID_USER_FQDN: specifies a fully-qualified username string (i.e. piper@foo.bar.com) - ID_DER_ASN: NOT SUPPORTED - specifies the binary DER encoding of an ASN - ID_KEY_ID: NOT SUPPORTED - The ID_KEY_ID type specifies an opaque byte stream which may be used to pass vendor-specific information necessary to identify which pre-shared key should be used to authenticate Aggressive mode negotiations.
Right ID	Identification (right side) "Identification Type - Remote ID Default value if field is blank: Remote IKE peer IP Address (ID Type ID_IPV4_ ADDR) Existent ID Types: - ID_IPV4_ADDR: specifies a single four (4) octet IPv4 address - ID_FQDN: specifies a fully-qualified username string (i.e. @foo.bar.com) - ID_USER_FQDN: specifies a fully-qualified username string (i.e. piper@foo.bar.com) - ID_DER_ASN: '"NOT SUPPORTED"' - specifies the binary DER encoding of an ASN - ID_KEY_ID: '"NOT SUPPORTED"' - The ID_KEY_ID type specifies an opaque byte stream which may be used to pass vendor-specific information necessary to identify which pre-shared key should be used to authenticate Aggressive mode negotiations.
IKE DH	IKE Diffie-Hellman group
IKE Encryption	IKE encryption algorithm
IKE Integrity	IKE integrity algorithm
IKE PRF	IKE pseudo-random function
IKE lifetime	The number of seconds before attempting to initiate an IKE rekey.
ESP Encryption	ESP encryption algorithm
ESP Integrity	ESP integrity algorithm

Parameter	Description
ESP lifetime	The number of seconds before attempting to initiate an IPsec rekey.
PFS	Diffie-Hellman group used for Perfect Forward Secrecy when rekeying IPsec SAs.
NAT Traversal	Encapsulate IPsec traffic in UDP. (Note that NAT Traversal is supported only for IKEv2.)
Debug Logging	Enables extra verbose logging. Enabling this option may negatively impact performance.
Initiation rate	Maximum number of tunnels to initiate every second.
Max. outstanding	Maximum number of tunnels that can be pending negotiation at any given time.
Tunnel setup timeout	Abort negotiation and declare it failed if a tunnel is not successfully negotiated after this interval (in seconds).
Retransmission interval	Interval between IKEv2 request retransmission (seconds).
Rekey margin	Amount of time before SA expiration when rekeying should start.
Initial contact	Send NOTIFY_INITIAL_CONTACT in IKEV2_EXT_IKE_AUTH packets.
Preemptive initiation	Initiate all configured tunnels at the start of test.
Enable Xauth	Enable extended authentication for IKEv1 Remote Access scenarios.
User Name	Username for IKEv1 extended authentication - see Generating a Sequence of Values for IKEv1 Extended Authentication on the facing pagefor information on generating a sequence of usernames.
Password	Password for IKEv1 extended authentication - see Generating a Sequence of Values for IKEv1 Extended Authentication on the facing pagefor information on generating a sequence of passwords.
Send wildcard TSr with modecgf	Enable wildcard TSr configuration when modecfg is enabled.
DPD Enable	Enable dead peer detection messages for IKEv1.
DPD Delay	The number of seconds between consecutive dead peer detection messages. The default value is 10, the minimum is 1, and the maximum is 4294967295.

Parameter	Description
DPD Timeout	The number of seconds to wait before deleting a tunnel because the device on the other end of the tunnel is not responding. The default value is 10, the minimum is 1, and the maximum is 4294967295.

Generating a Sequence of Values for IKEv1 Extended Authentication

Username for IKEv1 extended authentication

You can configure a format string that generates sequences of user names for the selected IPsec configuration element. The format string can include a combination of text and format characters.

You can intermix text strings with format characters. The format characters are **% : x i**.

Format character description:

- % start of complex increment string
- : field separator within the complex increment string
- x end of complex increment string (hexadecimal values)
- i end of complex increment string (decimal values)

User Name and Password Increment Format:

%[<start>][:[<modulo>][:<divisor>]]i|x , where

f (index, start, modulo, divisor) = ((start + index) / divisor) % modulo

Index: The tunnel index.

Start: Specifies the first value used in the % variable sequence, as well as its width. For example: 001 will generate values of 001, 002, 003, etc. Also, the number of characters in 'start' sets the minimum number of (left-zero-padded) characters in the output, as in "foo%0001i"

Modulo: Specifies how many values will be generated before the first value is repeated. For example, a value of 3 will generate three values and then repeat them. Effective default 'modulo' is infinity (if missing or 0). For example: a 'start' value of 001 and a 'modulo' value of 3 will generate 001, 002, 003, 001, 002, 003, etc.

Divisor: Specifies how many times each value will be repeated. Effective default 'divisior' is 1 (if missing or 0). For example, a 'start' value of 001 and a 'divisor' value of 2 will generate 001, 001, 002, 002, 003, 003, etc. When the string is concluded with an 'i' format character, the generated value will use decimal numbers. When the string is concluded with an 'x' format character, the generated value will use hexadecimal numbers.

Note: Format string with increment options are not available for Storm/FireStorm platforms.

User Name Examples

If you enter user%1id, BPS creates the strings user1id, user2id, and so forth.

You can use more than one Increment variable in a string. For example, if you enter U%1iid%1i, BPS creates the strings U1id1, U2id2, etc.

Another example is represented by the following complex increment string - host%100::3i-user%0:0:2x, that will create the following values: host100-user0, host100-user0, host100-user1, host101-user1, host101-user2, host101-user2, host102-user3, host102-user3, host102-user4, host103-user4, etc.

Password for IKEv1 extended authentication

You can configure a format string that generates sequences of passwords for the selected IPsec configuration element. The format string can include a combination of text and format characters.

You can intermix text strings with format characters. The format characters are **% : x i**.

Format character description:

- % start of complex increment string
- : field separator within the complex increment string
- x end of complex increment string (hexadecimal values)
- i end of complex increment string (decimal values)

Increment Format

See User Name and Password Increment Format.

Password Example

If you enter pass%1iword, BPS creates the strings pass1word, pass2word, etc.

You can use more than one Increment variable in a string. For example, if you enter P%iword%2i, BPS creates the strings P0word2, P1word3, P2word4,etc.

Note: Format string with increment options are not available for Storm/FireStorm platforms.

Note: The following options are only available on the Ixia PerfectStorm platform.

- IPv6 SLAAC Client Configuration Parameters
- DHCPv6 Client Configuration Parameters
- DHCPv6 Client Statistics
- DHCPv6 Request Options
- DHCPv6 Timeout and Retransmission Parameters

IPv6 SLAAC Client Configuration Parameters on the facing page (PerfectStorm Only) provides descriptions of the parameters available in IPv6 SLAAC Client Configuration elements. You can use these parameters to define the network settings for each IPv6 SLAAC Client Configuration element in your test. In addition to the IPv6 SLAAC Client Configuration parameter descriptions below, please be aware of the following:

- The interface identifiers are based on the EUI-64 identifier (derived from the interface's built-in 48-bit IEEE 802.3 address).
- The prefix that is advertised must have a prefix length of 64 (i.e. /64). Otherwise, the prefix information will be ignored.
- DAD is available only as part of the SLAAC implementation.
- When global duplicate address are detected, BPS provides a fallback mechanism. When duplicate link-local addresses are detected, the hosts become inactive.
- When parsing RAs, the lifetime options are not considered, so the system bounds addresses to an interface for an infinite time.
- A node waits for 2 seconds after sending a Neighbor Solicitation message before considering its tentative address to be unique.
- A Router Solicitation retry will occur if no response is received within 2 seconds.
- SLAAC does not work when **Duplicate MAC Address** is enabled on the interfaces.
- The advertised prefix information is ignored in the following scenarios:
- When the Autonomous flag is not set.
- When the prefix is the link-local prefix.

The following items are not supported with this feature:

- Security test component
- RFC2544 L2-3 test component
- BitBlaster L2-3 test component
- RoutingRobot L2-3 test component
- Emulated Router

IPv6 SLAAC Client Configuration Parameters

Parameter	Description
ID	The ID of the logical interface.
DAD (Duplicate Address Detection)	Enable Duplicate Address Detection. If this option is enabled, the DAD Fallback Base IP Address field must be configured or the DAD Fallback Random Address must be enabled.
DAD Fallback Base IP Address	This field is available when DAD is enabled. The hosts bits (last 64) will be used as the base host bits of the fallback address. When the DAD Fallback Random Address option is enabled, this field is ignored.
DAD Fallback Random Address	This field is available when DAD is enabled. The fallback address is generated using random host bits. If this option is not enabled, the DAD Fallback Base IP Address field is mandatory.

Parameter	Description
Stateless DHCPv6 Client Configuration	Container for a DHCPv6 Client Configuration element. If specified, the SLAAC configured hosts will use the Stateless DHCPv6 service to obtain configuration information such as the addresses of DNS recursive name servers.

The DHCPv6 Client Configuration Parameters tableshown below (PerfectStorm Only), provides descriptions of the parameters available in DHCPv6 Client Configuration elements. You can use these parameters to define the network settings for each DHCPv6 Client Configuration element in your test. DHCPv6 Client Statistics on the facing page (PerfectStorm Only) provides a description of the DHCPv6 client statistics. In addition to the DHCPv6 Client Configuration parameter descriptions below, please be aware of the following:

- NN templates for DHCPv6-Client-Stateful and DHCPv6-Loopback are available on the system.

DHCPv6 Client Configuration Parameters

Parameter	Description
ID	The ID of the logical interface.
Setup Rate	Setup Rate is the number of clients to start in each second.
Max Outstanding	Sessions are setup at the configured speed until there are this number of requests in progress, at which point new requests are added when others are completed.
Renew Time	The user-defined lease renewal timer. The value is defined in seconds and will override the lease renewal timer if it is not zero and is smaller than the server-defined value.
Timeout and Retry Config	DHCPv6 Timeout and Retransmission configuration.
Request Options Config	DHCPv6 Request Options Configuration container.
DUID-Type	DHCP Unique Identifier Type. Available options: [DUID-LLT] DUID Based on Link-layer Address Plus Time and [DUID-LL] DUID Based on Link-layer Address.
IA Type	Identity Association Type - Available options: [IANA] Identity association for Non-temporary Addresses [IATA] Identity Association for Temporary Addresses [IAPD] Identity Association for Prefix Delegation

Parameter	Description
IA T1	The suggested time at which the client contacts the server from which the addresses were obtained to extend the lifetimes of the addresses assigned (Initial Renew messages will be sent after this timer expires).
IA T2	The suggested time at which the client contacts any available server to extend the lifetimes of the addresses assigned (Initial Rebind messages will be sent after this timer expires and no renew messages were properly replied).

DHCPv6 Client Statistics

	Start Column	Description
DHCP Client Sessions	Sessions Initiated	Number of clients that where started. This corresponds to number of started DHCP Solicit sessions.
	Sessions Succeeded	Number of addresses that were successful configured.
	Setup Success Rate	The number of setup successes per second since setup process has started.
	Sessions Outstanding	The number of sessions initiated that are nor succeeded nor failed.
	Sessions Failed	Number of clients for which the requested DHCP address allocation failed.
	Teardown Initiated	Number of clients that where stopped. This corresponds to number of initiated release sessions.
	Teardown Succeeded	Number of addresses that were successful released.
	Teardown Failed	Number of interfaces that could not be configured until the REL_ MAX_RC was exceeded.
	Addresses Discovered	The number of DHCPv6 addresses learned.

	Start Column	Description
DHCPv6 Client Messages	Solicits Sent	Number of Solicit messages sent.
	Advertisements Received	Number of Advertise messages received.
	Advertisements Ignored	Number of received Advertise messages that were ignored.
	Requests Sent	Number of Request messages sent.
	Info Requests Sent	Number of Info Request messages sent.
	Reply Received	Number of Reply messages received.
	Renews Sent	Number of Renew messages sent.
	Rebinds Sent	Number of Rebind messages sent.
	Releases Sent	Number of Release messages sent.

The DHCPv6 Request Options element is used to identify a list of options in a message between a client and a server. DHCPv6 Request Options below (PerfectStorm Only) provides descriptions of the parameters available in DHCPv6 Client Configuration elements. You can use these parameters to define the network settings for each DHCPv6 Client Configuration element in your test.

DHCPv6 Request Options

Parameter	Description
ID	The ID of the logical interface.
Server Identifier	Option value: 2; The Server Identifier option is used to carry a DHCP Unique Identifier [DUID], identifying a server which is located between a client and another server.
Preference	Option value: 7; The Preference option is sent by a server to a client to affect the selection of a server by the client.
DNS Resolvers	Option value: 23; The DNS Recursive Name Server option provides a list of one or more IPv6 addresses of DNS recursive name servers to which a client's DNS resolver MAY send DNS queries. The DNS servers are listed in the order of preference for use by the client resolver.
DNS List	Option value: 24; The Domain Search List option specifies the domain search list the client is to use when resolving hostnames with DNS. This option does not apply to other name resolution mechanisms.

The DHCPv6 Timeout and Retransmission Parameters tableshown below (PerfectStorm Only) provides descriptions of the parameters available in DHCPv6 Timeout and Retransmission elements. You can use these parameters to define the network settings for each DHCPv6 Timeout and Retransmission element in your test.

DHCPv6 Timeout and Retransmission Parameters

Parameter	Description
Initial Solicit Timeout	RFC 3315 Initial Solicit timeout value in seconds.
Max Solicit Timeout	RFC 3315 Max Solicit Timeout value in seconds.
Solicit Attempts	RFC 3315 Max Solicit retry attempts.
Initial Request Timeout	RFC 3315 Initial Request Timeout value in seconds.
Max Request Timeout	RFC 3315 Max Request Timeout value in seconds.
Request Attempts	RFC 3315 Max Request retry attempts.
Initial Renew Timeout	RFC 3315 Initial Renew Timeout value in seconds.
Max Renew Timeout	RFC 3315 Max Renew Timeout value in seconds.
Initial Rebind Timeout	RFC 3315 Initial Rebind Timeout value in seconds.
Max Rebind Timeout	RFC 3315 Max Rebind Timeout value in seconds.
Initial Release Timeout	RFC 3315 Initial Release Timeout value in seconds.
Release Attempts	RFC 3315 Max Release retry attempts.
Initial Info Request Timeout	RFC 3315 Initial Info Request Timeout value in seconds.
Max Info Request Timeout	RFC 3315 Max Info Request Timeout value in seconds.
Info Request Attempts	RFC 3315 Info Request retry attempts.

Endpoint Element Group

The elements in this group represent the logical devices that will be transmitting or receiving traffic on the network.

Endpoint Elements

- IPv4 External Hosts
- IPv6 External Hosts
- IPv4 Static Hosts
- IPv6 Hosts
- IPv4 DHCP Hosts

- 6RD Customer Edge Routers
- User Equipment

The IPv4 External Host Parameters table shown below provides descriptions of the parameters available in IPv4 External Host elements. You can use these parameters to define the network settings for each IPv4 External Host element in your test.

IPv4 External Host Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Tags	A list of logical names for the host or the UE.
Base IP Address	Defines the minimum IP address.
Count	The number of hosts in the set.
NAT	Enables Network Address Translation (NAT) for network.

The IPv6 External Host Parameters table provides descriptions of the parameters available in IPv6 External Host elements. You can use these parameters to define the network settings for each IPv6 External Host element in your test.

IPv6 External Host Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Tags	A list of logical names for the host or the UE.
Base IPv6 Address	Defines the minimum router IPv6 address.
Count	The number of hosts in the set.

IPv4 Static Host Parameters on the facing page provides descriptions of the parameters available in IPv4 Static Host elements. You can use these parameters to define the network settings for each IPv4 Static Host element in your test.

Note: The acronym "PSN", listed in the table below refers to a Protected Subnet address. Protected Subnet addresses are utilized in IPsec configurations that include IKEv1. The PSN address must match the mirrored Base IP host address. For example, in a scenario with two IPv4 Static Hosts network elements:

- ip_static_hosts_1 and ip_static_host_2 (these networks are behind 2 IPsec routers)
- ip_static_host_1 has the Base IP Address : 40.0.0.1
- -ip_static_host_2 has the Base IP Address : 70.0.0.1

The following PSN Addresses will be configured:

- ip_static_host_1 will have the PSN Address : 70.0.0.1 (which is the ip_static_host_2 Base IP Address)
- ip_static_host_2 will have the PSN Address : 40.0.0.1 (which is the ip_static_host_1 Base IP Address)

IPv4 Static Host Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Tags	A list of logical names for the host or the UE.
Base IP Address	Defines the minimum IP address.
Count	The number of hosts in the set.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
PSN Address	Defines the protected subnet. This IP must match the mirrored IP host address (not subnet address). his parameter is mandatory for IKEv1.
PSN Netmask	Network mask for the protected subnet in Port-to-DUT scenarios when we are in single protected subnet cases. This parameter is mandatory for IKEv1.
DNS Settings	A selection from the DNS Settings network element list.
Conditional DNS Settings	Select a Conditional DNS ID from the drop-down list to use a Conditional DNS.
LDAP Settings	Select a LDAP ID from the drop-down list to use a LDAP server.
MPLS Settings	Type in the name of one or more MPLS-settings. MPLS Settings ID's are separated by a semicolon.
Per-host Stats	Allows you to track per-host statistics. Enabling this parameter may affect throughput. This parameter may not work with every component types.
NAT	Enables Network Address Translation (NAT) for network.

The IPv6 Host Parameters table shown below provides descriptions of the parameters available in IPv6 Static Host elements. You can use these parameters to define the network settings for each IPv6 Static Host element in your test.

IPv6 Host Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Tags	A list of logical names for the host or the UE.
IPv6 Address Allocation Mode	Defines the allocation mode of the IPv6 address or addresses. If Static allocation mode is selected, the hosts will be configured with the Base IPv6 Address and incremented by 1 if the Count field is higher than 1. If SLAAC allocation mode is selected, an IPv6 SLAAC Client Configuration element must be referenced in the Alloc Config field. If DHCPv6 allocation mode is selected, a DHCPv6 Client Configuration element must be referenced in the Alloc Config field. (Exception: When IA Type is configured as IAPD.)
Alloc Config	Container for the IPv6 address allocation mode configuration.
Base IPv6 Address	Defines the minimum router IPv6 address.
Count	The number of hosts in the set.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Prefix Length	The length (in number of bits) of the service provider's 6rd IPv6 prefix set in the subnet mask.
DNS Settings	A selection from the DNS Settings network element list.
Conditional DNS Settings	Select a Conditional DNS ID from the drop-down list to use a Conditional DNS
MPLS Settings	Type in the name of one or more MPLS-settings. MPLS-Settings ID's are separated by a semicolon.
Maximum Bandwidth	Specify the maximum bandwith (TX plus RX) allowed for each host. This value is applicable when a client sends traffic from a source IP or a server sends to a destination IP in this range which is in the "Client Tags" of a component.

Parameter	Description
Per-host Stats	Allows you to track per-host statistics. Enabling this parameter may affect throughput. This parameter may not work with every component types.
NAT	Indicates that the hosts are in a private NAT network.
IP Address Algorithm	Select to use a Random or Sequential algorithim to generate IP addresses for the hosts.

The IPv4 DHCP Host Parameters table shown belowprovides descriptions of the parameters available in IPv4 DHCP Host elements. You can use these parameters to define the network settings for each IPv4 DHCP Host element in your test.

IPv4 DHCP Host Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Tags	A list of logical names for the host or the UE.
Count	The number of hosts in the set.
Allocation Rate	The number of new UEs on the mobile network per second.
Accept Simulated Servers Only	Causes the DHCP Client to only accept offers from BreakingPoint DHCP Servers.
NAT	Enables Network Address Translation (NAT) for network.
Per-host Stats	Allows you to track per-host statistics. Enabling this parameter may affect throughput. This parameter may not work with every component types.

The 6RD Customer Edge Router Parameters table shown below provides descriptions of the parameters available in 6RD Customer Edge Router elements. You can use these parameters to define the network settings for each 6RD Customer Edge Router element in your test.

6RD Customer Edge Router Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Tags	A list of logical names for the host or the UE.

Parameter	Description
Base CE IP Address	Defines the minimum CE IP address.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
Count	The number of hosts in the set.
DNS Settings	A selection from the DNS Settings network element list.
Per-host Stats	Allows you to track per-host statistics. Enabling this parameter may affect throughput. This parameter may not work with every component types.
6RD Prefix	An IPv6 prefix selected by the service provider for use by a 6rd domain.
6RD Prefix Length	The length (in number of bits) of the service provider's 6rd IPv6 prefix set in the subnet mask.
IPv4 Mask Length	The number of high-order bits in the network address that are identical across all Customer Edge (CE) IPv4 addresses within a given 6rd domain.
Border Relay IPv4 Address	A 6rd-enabled router managed by the service provider on a 6rd domain.
Hosts per CE	The number of simulated hosts behind each CE router, where a CE is a 6rd NAT client device. Calculated by multiplying the result of the Count field by the result of the Hosts per CE field.

The User Equipment Parameters table shown below provides descriptions of the parameters available in User Equipment elements. You can use these parameters to define the network settings for each User Equipment element in your test.

User Equipment Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
Tags	A list of logical names for the host or the UE.
Allocation Rate	The number of new UEs on the mobile network per second.

Parameter	Description
Per-UE/Bearer Stats	Enables keeping per-UE/bearer statistics.
NAT	Enables Network Address Translation (NAT) for network.
Mobility Action	Allows you to select the action that is invoked after the configured interval has expired.
Mobility Interval	Allows you to specify the amount of time before the configured action is invoked. You can specify any amount of time between 1,000 and 10,000,000 milliseconds.
Use IPv6	When this feature is enabled (box is checked), the UE requests an IPv6 address. When this feature is not enabled (box is unchecked) the UE requests an IPv4 address by default.
Mobility with traffic flow	When this feature is enabled, the UEs attach only when the traffic flow starts and detach as soon as the traffic flow is completed. When this feature is not enabled, all the UEs attach at the start of the test and do not detach until the test is complete.
User Equipment Info Settings	Information required to properly represent User Equipment on an LTE mobile network.
DNS Settings	A selection from the DNS Settings network element list.

Note: When the Use IPv6 box is checked, the UE will request an IPv6 address. When this box is not checked, by default the UE will request an IPv4 address.

LTE Element Group

The elements in this group represent simulated devices that act as infrastructure in the LTE core network.

LTE Elements

- eNodeB/MME (GTPv2)
- eNodeB/MME/SGW (GTPv2)
- eNodeB (S1AP/GTPv1)
- SGW/PGW
- MME/SGW/PGW
- PGW

Note: The LTE Elements listed below also support IPv6 addresses.

The eNodeB/MME (GTPv2) Parameters table provides descriptions of the parameters available in eNodeB/MME (GTPv2) elements. You can use these parameters to define the network settings for each eNodeB/MME (GTPv2) element in your test.

eNodeB/MME (GTPv2) Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
MME IPv4 Address	The IPv4 address of the MME to which the eNodeB is connected.
eNodeB List	List of cell towers used in the network. **Note:** The ability to split eNB IP addresses and MME IP addresses in different IP networks is not supported on Storm/Firestom platforms. - Netmask : Defines the subnet mask for the eNodeB Network Address - Gateway IP Address : Defines the eNodeB default gateway - Container : The device or network that eNodeB element resides behind or within. [Note: eNodeBs containers in the eNodeB list must be on same interface as the MME container]
Remote SGW IPv4 Address	IP address of server that routes and forwards user data.
Remote PGW IPv4 Address	IP address of server that routes and forwards user data.
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
DNS Settings	A selection from the DNS Settings network element list.
UE IP Allocation Mode	Determines whether UEs will suggest static IP addresses or request dynamic addresses.
Base UE Static Pool Address	Defines the base address for static UE IP addresses.

The eNodeB/MME/SGW (GTPv2) Parameters table provides descriptions of the parameters available in eNodeB/MME/SGW (GTPv2) elements. You can use these parameters to define the network settings for each eNodeB/MME/SGW (GTPv2) element in your test.

eNodeB/MME/SGW (GTPv2) Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	The IP address for the virtual device that will be simulated.
Remote PGW IPv4 Address	IP address of server that routes and forwards user data.
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
DNS Settings	A selection from the DNS Settings network element list.
UE IP Allocation Mode	Determines whether UEs will suggest static IP addresses or request dynamic addresses.
Base UE Static Pool Address	Defines the base address for static UE IP addresses.

The eNodeB (S1AP/GTPv1) Parameters table provides descriptions of the parameters available in eNodeB (S1AP/GTPv1) elements. You can use these parameters to define the network settings for each eNodeB (S1AP/GTPv1) element in your test.

eNodeB (S1AP/GTPv1) Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within. This can be an Interface or an IPsec router.
eNodeB List	List of cell towers used in the network.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.

Parameter	Description
Netmask	Defines the subnet mask for the Network Address.
DNS Settings	A selection from the DNS Settings network element list.
SCTP Over UDP	Selects SCTP over UDP to connect the eNodeB to the MME. SCTP is the default transport protocol for the eNodeB.
SCTP Source Port	The source port used for opening SCTP associations.
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.

The SGW/PGW Parameters table provides descriptions of the parameters available in SGW/PGW elements. You can use these parameters to define the network settings for each SGW/PGW element in your test.

SGW/PGW Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	The IP address for the virtual device that will be simulated.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
Base IPv6 Pool Address	The base address of the dynamic IPv6 address range that is allocated to the UEs upon request.
Base IP Pool Address	Defines the minimum pool address. Used only when the UE IP Allocation Mode for the eNodeB/MME/SGW (GTPv2) is set to Static .
Advertised PGW Address	The simulated PGW will advertise that the MME should connect for GTP-U.
Advertised SGW Address	The simulated SGW will advertise that the MME should connect for GTP-C.
Maximum Sessions	Maximum concurrent UE sessions
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.

Parameter	Description
DNS Settings	A selection from the DNS Settings network element list.

The MME/SGW/PGW Parameters table provides descriptions of the parameters available in MME/SGW/PGW elements. You can use these parameters to define the network settings for each MME/SGW/PGW element in your test.

MME/SGW/PGW Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within. This can be an Interface or an IPsec router.
IP Address	The IP address for the virtual device that will be simulated.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
Base IP Pool Address	Defines the minimum IP pool address.
Base IPv6 Pool Address	The base address of the dynamic IPv6 address range that is allocated to the UEs upon request.
Advertised PGW Address	The simulated PGW will advertise that the MME should connect for GTP-U.
Advertised SGW Address	The simulated SGW will advertise that the MME should connect for GTP-C.
Maximum Sessions	Maximum concurrent UE sessions.
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.
User Equipment Info Settings	Information required to properly represent User Equipment on an LTE mobile network.
DNS Settings	A selection from the DNS Settings network element list.

The PGW Parameters table provides descriptions of the parameters available in PGW elements. You can use these parameters to define the network settings for each PGW element in your test.

PGW Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	The IP address for the virtual device that will be simulated.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
Base IP Pool Address	Defines the minimum IP pool address.
Base IPv6 Pool Address	The base address of the dynamic IPv6 address range that is allocated to the UEs upon request.
Maximum Sessions	Maximum concurrent UE sessions.
PLMN Settings	A reference to a row item in the Public Land Mobile Network network element list.
DNS Settings	A selection from the DNS Settings network element list.

Mobile Configuration Element Group

The elements in this group define common configurations that can be shared across multiple LTE or 3G elements.

Mobile Configuration Elements

- HSS/UE Database
- Public Land Mobile Network
- Mobility Session Information

The HSS/UE Database Parameters table provides descriptions of the parameters available in HSS/UE Database elements. You can use these parameters to define the network settings for each HSS/UE Database element in your test.

HSS/UE Database Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Count	The number of hosts in the set.

Parameter	Description
IMSI Base	Identifies the SIM card of each device.
IMEI Base	International Mobile Equipment Identity.
MSISDN Base	Mobile Subscriber Integrated Services Digital Network Number.
Secret Key	Value of a secret key that is generated for each UE.
Secret Key Step	The value the Secret Key is incremented by for each UE.
Operator Variant	Specifies a unique value originally assigned by the UE manufacturer. The operator variant is usually unique to each brand of UE.
Mobility Session Info	Reference to a shared Mobility Session Info element for sharing configuration between multiple UEs.

The Public Land Mobile Network Parameters table provides descriptions of the parameters available in Public Land Mobile Network elements. You can use these parameters to define the network settings for each Public Land Mobile Network element in your test.

Public Land Mobile Network Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Mobile Country Code	The Mobile Country Code of the device to be tested.
Mobile Network Code	The Mobile Network Code of the device to be tested.
Description	User-defined description of the network element. The contents of this field are included when searching, so it can be used as a way to enhance the ability to find specific Network Neighborhoods.

The Mobility Session Information table provides descriptions of the parameters available in Mobility Session Information elements. You can use these parameters to define the network settings for each Mobility Session Information element in your test.

Mobility Session Information

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Access Point Name	An identifier used by mobile devices when connecting to a GSM carrier.
Username	The username that will be encoded along with the APN in requests from the SGSN to the GGSN to setup new PDPs.
Password	The password that will be encoded along with the APN in requests from the SGSN to the GGSN to setup new PDPs.
Dedicated Bearers	The number of dedicated paths over which a UE sends and receives data via the PDN.
Bearer Configuration	Settings that apply on a per-bearer basis, including the QCI for the bearer.

3G Element Group

The elements in this group represent simulated devices that act as infrastructure in a 3G mobile network.

3G Elements

- GGSN
- SGSN

The GGSN Parameters table provides descriptions of the parameters available in GGSN elements. You can use these parameters to define the network settings for each GGSN element in your test.

GGSN Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	The IP address for the virtual device that will be simulated.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Advertised Control IP Address	Defines the address the SGSN should connect to for GTP Control communication.

Parameter	Description
Advertised User Data IP Address	Defines the address the SGSN should connect to for GTP User Data communication.
Netmask	Defines the subnet mask for the Network Address.
Base client lease IP Address	Defines the minimum IP lease address.
Base client lease IPv6 Address	Defines the minimum IPv6 lease address.
Count	The number of hosts in the set.
DNS Settings	A selection from the DNS Settings network element list.

The SGSN Parameters table provides descriptions of the parameters available in SGSN elements. You can use these parameters to define the network settings for each SGSN element in your test.

SGSN Parameters

Parameter	Description
ID	The ID of the logical interface that displays the network component traffic.
Container	The device or network that this element resides behind or within.
IP Address	The IP address for the virtual device that will be simulated.
Gateway IP Address	Defines the default gateway router address. The Gateway Address must use the same subnet as the Network Address.
Netmask	Defines the subnet mask for the Network Address.
GGSN IP Address	The address of the GGSN that the simulated SGSN will connect to.

Creating a Network Neighborhood

The Network Neighborhood contains the addressing rules available for each test interface. Each test interface has a set of subnets to define the addressing rules for test traffic originating from each test interface.

The following video link provides a tutorial on how to Configure a Network Neighborhood: https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

The Control Center offers two methods of creating a Network Neighborhood: by either by creating a new Network Neighborhood or cloning an existing Network Neighborhood.

The following section will provide instructions on creating a new Network Neighborhood.

To create a new Network Neighborhood:

1. Select **Control Center > New Neighborhood** from the Menu bar.
2. Select the type of device you are testing.
3. Click **Select**.
4. Enter a name for the Network Neighborhood in the New Network Neighborhood **Name** field.
5. Click **OK**.

Cloning a Network Neighborhood

Cloned Network Neighborhoods are duplicates of an existing Network Neighborhood, which means they inherit all the elements and parameters defined for the parent Network Neighborhood.

To clone a Network Neighborhood:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Find the Network Neighborhood you want to clone.
   a. If the Network Neighborhood you want to clone does not appear in the list, enter a portion of the name of the Network Neighborhood into the **Filtered Search** field and click the **Search** button.
3. Select the Network Neighborhood you want to clone from the **Network Neighborhood** list.
4. Click **Save As**. A popup window will display, allowing you to name the Network Neighborhood.
5. Enter a name for the Network Neighborhood in the **New Network Neighborhood Name** field.
6. Click the **OK** button.

Deleting a Network Neighborhood

When you delete a Network Neighborhood, all its parameters and definitions will be removed from the system. You will need to select a new Network Neighborhood for any test using the deleted Network Neighborhood.

Note: If you open a test that uses a deleted Network Neighborhood, the Control Center will alert you that the Network Neighborhood is missing. Click **OK** to close the message window.

To delete a Network Neighborhood:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select the Network Neighborhood you want to delete from the **Network Neighborhood** list.
3. Click the Delete button.
4. Click **OK** when the confirmation window is displayed.

Conditional DNS

The Conditional DNS feature provides a mechanism to transparently redirect traffic flows using user-configurable DNS resolution parameters.

Together with the existing DNS functionality, the Conditional DNS feature emulates a complete DNS service necessary for complex scenarios where a differentiated DNS response needs to be generated based on the source IP address of the DNS query.

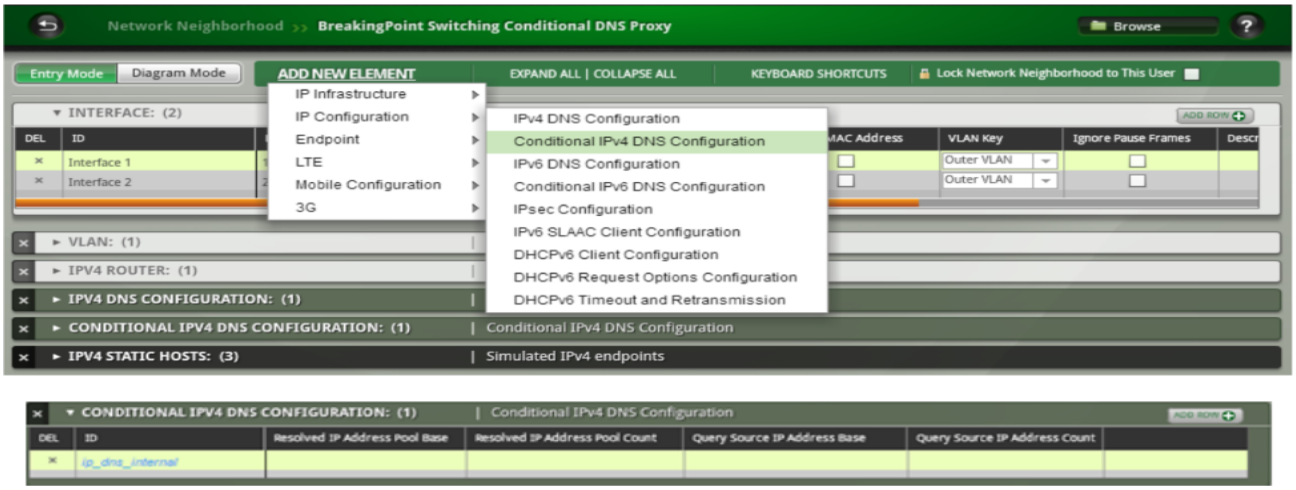
Such a DNS infrastructure is required when testing devices or systems that are transparently intercepting traffic based on DNS hostname resolution (e.g., certain transparent proxy devices).

Configure Conditional DNS and Clients

The Conditional DNS feature alone is used to resolve a DNS query that is initiated by known Query source IP Addresses using a user defined Resolved IP Address.

To configure Conditional DNS following the two elements are needed:

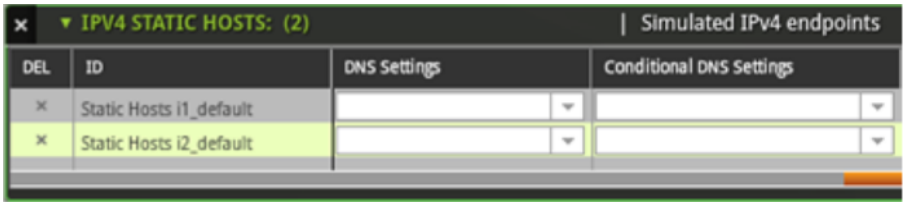
1. From the Network Neighborhood that you are using for your test, select **Add New Element > IP Configuration > Conditional IPv4 DNS Configuration** or Conditional IPv6 DNS **Configuration** (based on your test scenario).



- Enter the parameters as described in the table below.

ID	Description
Resolved IP Address Pool Base	This base IP address is the first IP address that can be returned when queries are received from IP addresses in the range defined by the **Query Source IP Address Base** and **Query Source IP Address Count**. If the Resolved IP Address Pool Count = 1, then this IP address will always be returned.
Resolved IP Address Pool Count	This value (along with the **Resolved IP Address Pool Base**) defines the IP addresses that are available in the Resolved IP Address Pool. For example, if the **Resolved IP Address Pool Base** is 40.25.101.251 and the **Resolved IP Address Pool Count** = 5, the pool includes IP address from 40.25.101.251 to 40.25.101.255. The actual IP address that is returned to a query is chosen randomly from within this range.
Query Source IP Address Base	When queries are made by source IP addresses within a specific range, an IP address defined within the Resolved IP Address Pool (described above) will be returned. This base IP address is the first source IP address within the range.
Query Source IP Address Count	This value (along with the **Query Source IP Address Base**) defines the query source IP addresses that will resolve to an IP address in the Resolved IP Address Pool. For example, if the **Query Source IP Address Base** is 192.168.10.1 and the Q**uery Source IP Address Count** = 5, the pool includes IP addresses from 192.168.10.1 to 192.168.10.5. Queries from IP addresses within this range will resolve to an IP address in the Resolved IP Address Pool.

2. Edit or add your **IPv4 Static Hosts**, **IPv4 DHCP Hosts** or **IPv6 Hosts** (this is the interface that will resolve DNS Queries based on the Conditional DNS Configuration described above).



For the **Conditional DNS Setting** field, use the drop-down arrow to select the Conditional DNS Configuration id that was created in step 1.

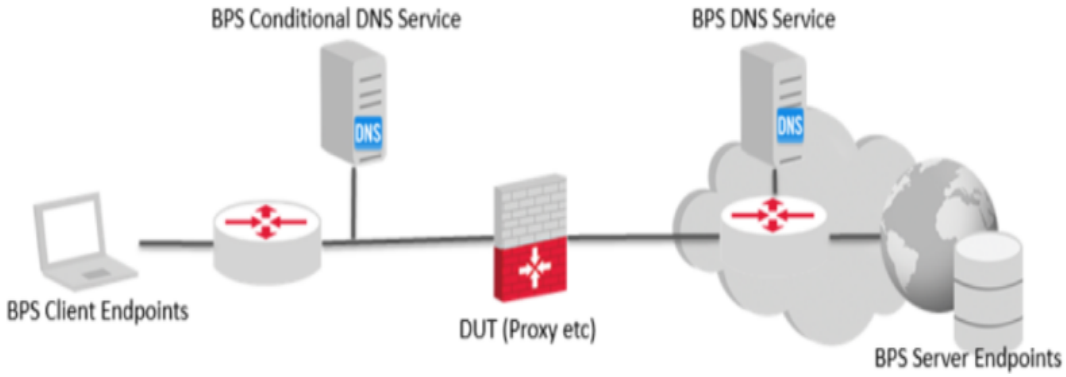
Explicit and Transparent Proxy Background

An explicit proxy is deployed in a scenario where a client is aware that their server request and resulting transaction is routed through a proxy (for example, when a client's HTML browser has a Proxy

IP address setting). A transparent proxy is deployed in a scenario where a client is NOT aware that their server request and resulting transaction is routed through a proxy.

Conditional DNS Example

In the scenario shown in the image below we will use the Conditional DNS feature to transparently route traffic through an external DUT. Using pre-existing DNS functionality as well as BreakingPoint`s unique capability to emulate client and server endpoints, a complete infrastructure service can be emulated to wrap-up the DUT (one BPS test interface can be used on each side of the network).



Configuration Details

The following information provides details about an example Conditional DNS configuration.

The BreakingPoint_Conditional_DNS.bpt canned test can be loaded and used as a reference for the Network Neighborhood elements described below.

1. The most important aspect of this configuration is the Network Neighborhood. It contains the following elements:
   a. **Three IPv4 static hosts:**
      i. Clients_Endpoints. – These endpoints act as clients behind a virtual router. The Virtual Router is VR-clients which is configured on the VLAN_VR_Clients VLAN defined on interface 1.
      ii. Internal_DNS - Acts as internal DNS servers and are configured on Interface 1. The Client sends a DNS Query and the Internal DNS Server responds with an IP address defined in the Conditional IPv4 DNS Configuration (which has the tag “Proxy”).
      iii. Web_DNS_Server - Acts as external DNS server as well as HTTP server. It is configured on Interface 2.
   b. **Conditional DNS:**
      i. Proxy - This element configures the Conditional DNS service so that if a request comes from 192.168.0.1 it will respond with the 40.25.101.254 as the resolved hostname IP.
2. **Test Components and Super Flows:**
   The Network Neighborhood elements described above will be mapped to 2 test components:

a. **Client Component:** Tags-Client Tags-Clients_Endpoints and Server Tags-Internal_DNS
   i. Clients_Endpoints will send DNS queries to the configured DNS server which is the host defined under Internal_DNS.
   ii. Host within Internal_DNS will reply with one of configured HTTP proxy IP addresses (based on the pool configured in the Conditional DNS Server).
   iii. Host in Clients_Endpoints will then send the HTTP GET request to the resolved IP address (HTTP proxy IP).
b. **Server Component:** Tags-Server Tags- Web_DNS_Server
   i. The HTTP proxy will initiate one DNS query to the DNS server in Web_DNS_Server.
   ii. DNS server in Web_DNS_Server will reply with the IP address of the external HTTP server.
   iii. The HTTP proxy then send the HTTP GET request to the resolved IP address

3. Each Test Component has the Following Application Profile:
   a. **BreakingPoint Conditional DNS Client** - There are 3 Super Flows within BreakingPoint Conditional DNS Client which are designed to perform the following.

      In Manage Host, a host needs to be added in order to send the hostname in the DNS Query sent by the Client for resolution. Apart from default hostnames like "Client" and "Server" under manage hosts, there is another hostname defined as "hostname" which contains the actual hostname that the DNS will carry for resolution.

      There needs to be two flows and the following should be the assignment for the client and server.

      - For the Flow DNS, "Client" & "Server" should be used.
      - For the Flow HTTP, "Client" & "hostname" should be used which is added within Manage Hosts.

      This super flow will first resolve the hostname (as in Resolve host as defined in hostname) and then use the resolved IP address as the destination for HTTP (this will happen if Use Response has been set to true in the Resolve Action) to send the HTTP GET request.
   b. **BreakingPoint Conditional DNS Server** - This Application Profile has a single superflow.

      This superflow should be run as a server component. Per the defined action it will wait for few specific HTTP GET requests. If the HTTP GET request path matches with the path specified in the conditional request then it responds with 200 OK. Otherwise it will close the connection after a timeout of 30 seconds.

In terms of the actual transaction sequence the following messages will be executed as follows:

a. Client is sending request to Internal DNS server
b. Server is resolving the query with the IP “40.25.101.254” (which is also the IP of the DUT)
c. Client is sending an HTTP request to the above IP “40.25.101.254”
d. DUT is performing DNS request for the domain present in the client HTTP request (“HOST” field)
e. DNS server is replying with an IP part of “Servers” IP range
f. DUT is sending the HTTP request to the HTTP server
g. DUT is forwarding the HTTP response to the client

Enabling and Viewing Per-UE/Bearer Statistics

Enabling this feature allows you to examine data for both mobility and non-mobility tests by tracking statistics for each UE (and IP address) in the test. When this feature is enabled, statistics are generated for each application the UE (and/or IP) sends or receives traffic on. The Per-UE/Bearer Stats feature is supported in Application Simulator, Client Simulator, Recreate, Session Sender, and Stack Scrambler test components; however, it is not supported in Routing Robot, Bit Blaster, or Security tests.

Note: Be aware that when this feature is enabled, it decreases performance, so tests may run slower. It also requires memory usage, possibly resulting in fewer simultaneous sessions.

This section will provide an overview of the tasks you must complete to enable the Per-UE/Bearer Stats feature and access the statistics in the downloaded .csv file using the Extended Stats feature.

The Enabling and Viewing Per-UE Statistics table lists the tasks you must complete to enable Per-UE/Bearer Stats and view the report.

Enabling and Viewing Per-UE Statistics

Task	Description
Task 1	Create a Network Neighborhood
Task 2	Create and Run a Test
Task 3	View the Report

Task 1: Create a Network Neighborhood

This section will describe how to create a Network Neighborhood.

To create a Network Neighborhood:

1. Select Control Center > Open Neighborhood from the Menu bar.
2. Search the list for a Network Neighborhood that supports mobile equipment (for example, BreakingPoint LTE eNodeB-MME-SGW-PGW).
3. Select the Network Neighborhood from the list and click Open.
4. Open the element labeled User Equipment.
5. Select the Per-UE/Bearer Stats check box and click Save As.
6. Enter a name for the Network Neighborhood in the Save As Network Neighborhood field and click OK.

Task 2: Create and Run a Test

This section will describe how to create and run a test.

To create and run a test:

1. Select Test > New Test from the Menu bar.
2. Select the Network Neighborhood created in the previous section and click Select.
3. Click ADD NEW in the Test Components section.
4. Select Application Simulator from the list and click Select.
5. Enter a name for the component and click Create.
6. Make adjustments to the Application Simulator component settings as necessary.
7. Click Save and Run.

Task 3: View the Report

This section will describe how to view the report.

To view the report:

1. Click the View the report icon at the bottom of the Real-Time Statistics screen.
2. Click the Download drop-down link in the upper right-hand corner of the report.
3. Select Extended Stats. The file will download as a .zip file.
4. Save the file.
5. Locate and open the zipped file.
6. Open the .csv file. The transactions for each UE in the test will be included in the file.

Enabling and Viewing Per-Host Statistics

The Per-host Stats feature provides visibility into statistics for each individual IP included in your tests. When this feature is enabled, statistics are generated for each application the IP sends or receives traffic on.

Note: Be aware that when this feature is enabled, it decreases performance, so tests may run slower. It also requires memory usage, possibly resulting in fewer simultaneous sessions.

This section will provide an overview of the tasks you must complete in order to access the statistics for each IP in the downloaded .csv file using the Extended Stats feature.

Enabling and Viewing Per-host Statistics below lists the tasks you must complete to enable the Per-host Stats feature and view the report.

Enabling and Viewing Per-host Statistics

Task	Description
Task 1	Create a Network Neighborhood
Task 2	Create and Run a Test
Task 3	View the Report

Task 1: Create a Network Neighborhood

This section will describe how to create a Network Neighborhood.

To create a Network Neighborhood:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Search the list for a Network Neighborhood that contain Endpoint elements.
3. **Note**: Endpoint elements are elements that represent the logical devices that will be transmitting or receiving traffic on the network.
4. Select the Network Neighborhood from the list and click **Open**.
5. Select **ADD NEW ELEMENT > Endpoint** and select an element from the list.
6. Select the **Per-host** check box and click **Save As**.
7. Enter a name for the Network Neighborhood in the **Save As Network Neighborhood** field and click **OK**.

Task 2: Create and Run a Test

This section will describe how to create and run a test.

To create and run a test:

1. Select **Test > New Test** from the Menu bar.
2. Select the Network Neighborhood created in the previous section and click **Select**.
3. Click **Add New** in the Test Components section.
4. Select **Application Simulator** from the list and click **Select**.
5. Enter a name for the component and click **Create**.
6. Make adjustments to the Application Simulator component settings as necessary.
7. Click **Save and Run**.

The following video link provides a tutorial on how to build and run an Application Simulator test:
https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

Task 3: View the Report

This section will describe how to view the report:

To view the report:

1. Click the **View the report** icon at the bottom of the Real-Time Statistics window.
2. Click the **Download** drop-down link in the upper right-hand corner of the report.
3. Select **Extended Stats**. The file will download as a .zip file.
4. **Save** the file.
5. Locate and open the zipped file.
6. Open the .csv file. The transactions for each host in the test will be included in the file.

Component Tags

Component Tags represent the type of network behavior that is associated with an element within a network. For instance, elements labeled with the Web Client tag can all be configured to behave as

similar web clients using a single configuration. Component Tags eliminate the need to assign specific configurations to devices each time they appear in the network. You can simply attach the Tag to the element you want to be associated with a particular configuration.

Component Tag Functionality

Component Tags are similar to domains in that they offer a way to assign a logical name to a group of IP addresses, or hosts, that are involved in a simulation. However Component Tags differ from domains in two important ways.

First, a Component Tag can be applied to multiple sets of IP addresses.

Previously, when a test component referred to a domain, it was always referring to one set of IP addresses on one particular interface. However, when a component refers to a Tag, it is referring to any Host Set (a particular set of IP addresses) that carries that Component Tag, regardless of which interface that Host Set appears on.

For example, you might use a Tag called Web Client in your network. Since many elements in your simulated network might act as web clients, the Web Client tag might appear on multiple sets of elements all over the network. When a test component is configured to transmit its traffic from the Web Client tag, the set of all hosts that carry the Web Client tag will be grouped and used as client addresses for that component.

Second, a particular set of IP addresses, or Host Set, can carry more than one tag.

Some of the hosts in your simulated network might act not only as web clients, but mail clients as well. For those Host Sets, you might apply both the Web Client tag and the Mail Client tag. In your test, the component that references the Web Client tag will include those addresses among its client addresses. Any other component referencing the Mail Client tag will include those same addresses.

Component Tag Compatibility

For tests created prior to Release 3.0, the Network Neighborhood definitions will automatically be migrated to the new Component Tag format. In that migration, every domain that existed will become a group of Host Sets in the new Network Neighborhood. That group of Host Sets will be given a Component Tag that maps to the old domain. The format of the Component Tag will be i<interface number>_<domain name>. For example, domain mydomain defined on Interface 3 would have the Component Tag i3_mydomain.

Tests that were configured to use a particular interface and domain will now have a reference to the corresponding Component Tag. So a test that had Interface 1 checked with domain default set as a client, it will now reference a client Component Tag of i1_default. This allows those preexisting tests and Network Neighborhoods to continue to function exactly as they did before.

Component Tags in Test Labs

Test labs prior to Release 3.0 were generally set up to run on Network Neighborhoods with the domain default on Interface 1 as the client, and the domain default on Interface 2 as the server. With Component Tags, those test labs will now be configured to run from the client tag i1_default to the server tag i2_default.

Component Tag Capability

It is possible to take advantage of the new flexibility provided by tags to use test labs that were previously restricted to Interface 1 and 2 so that they transmit on any interfaces desired. All that is necessary is to apply the i1_default Component Tag to Host Sets anywhere in the network that you want to act as clients, and apply the i2_default Component Tag to any Host Sets that you want to act as servers. Even if they are on interfaces other than 1 or 2, the test lab will group all of the hosts that have those tags and use them in the test.

Network Neighborhood Subnets

Each component tag must contain at least one subnet. The number of subnets that can be added depends on the type of subnet you are defining (i.e., VLAN or non-VLAN subnet). A component tag can contain one non-VLAN subnet; each subsequent subnet that you add to the component tag must have a VLAN ID assigned to it.

Dynamic Subnets

Dynamic subnets allow you to use Network Neighborhood parameters to customize the clients and servers for LTE traffic. These parameters allow you define the server and client port numbers and configure additional settings for certain LTE protocols. You can specify whether the devices in your test receive IP addresses from an internal DHCP server, GTP, or SGW. Since the parameters vary for each protocol, you will need to have an understanding of each LTE protocol in order to correctly configure the server and client.

When using dynamic subnets in your tests, it is important to remember that the number of sessions generated is directly proportional to the number of UEs (user equipment) per second configured for your test. For instance, if you configure a test to generate 100,000 maximum concurrent sessions and 100,000 sessions per second, you may expect the test to generate a great number of sessions.

However, if your test is configured for 1 new UE per second, the sessions per second will be limited to 1 UE per second, until the test reaches the maximum number of simultaneous UEs.

The following sections will describe how to add a subnet to a component tag. These sections will reference several network parameters. For more information on network parameters, see the section.

Defining a Subnet

To define a subnet:

1. Select **Control Center** > **Open Neighborhood** from the Menu bar.
2. Locate the Network Neighborhood from the Network Neighborhood list.

   Note: If the Network Neighborhood you want does not appear in the list, enter a portion of the name of the Network Neighborhood into the **Filtered Search** field and click the **Search** button.
3. Select a Network Neighborhood and click **Open**.

4. Click the **Add New Element** link to add the type of element (or elements) you want to use in your network configuration if it is not displayed. For example, select **IP Infrastructure** > **IPv4 Router**.
5. Click the arrow to expand the tab for the first element in the Network Neighborhood.
   If you are defining a subnet for the external interface (for endpoint testing), see External Interface Addressing on page 157.
6. Select Interface 1 in the Container field of the IPv4 Router interface.
7. Enter an IP address in the **IP Address** field of the IPv4 Router tab.
8. Enter a gateway IP address in the **Gateway IP Address** field of the IPv4 Router interface.
9. Enter a netmask in the **Netmask** field of the IPv4 Router tab.
10. Click the **Add Row** button in the IPv4 Router tab.
11. Repeat steps 4 - 10 for each additional interface you want to add to this element.
12. Click **Save** when you are done.

Defining a VLAN-Enabled Subnet

This section provides instructions for defining a VLAN-enabled subnet.

Note: Using a VLAN-enabled subnet will allow you to send and receive traffic on the same interface.

Note: The VLAN_Key setting should always be left at the default value, "Outer VLAN", when configuring single VLAN tags. Disregard that a single VLAN may indicate "Inner VLAN".

To define a VLAN-enabled subnet:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select a Network Neighborhood from the **Network Neighborhood** list.
3. If the Network Neighborhood you want does not appear in the list, enter a portion of the name of the Network Neighborhood into the **Filtered Search** field and click the **Search** button.
4. Select a Network Neighborhood and click Open.
5. Click the arrow to expand the IPv4 Static Hosts tab.

   Note: If you are defining a subnet for the external interface (for endpoint testing), see the External Interface Addressing on page 157
6. Click the **Add New Element** link.
7. Select **IP Infrastructure** > **VLAN**.
8. Select Interface 1 in the Container field of the VLAN tab.
9. Enter a VLAN ID in the Inner VLAN ID field of the VLAN tab.
   a. Enter a VLAN ID in the **Outer VLAN ID** field if necessary.
10. Select vlan_1 in the Container field in the IPv4 Static Hosts tab.
11. Click the **Add Row** button on the VLAN tab.
12. Select Interface 2 in the the Container field of the VLAN tab.

13. Enter a VLAN ID in the Inner VLAN ID field of the VLAN tab.
    a. Enter a VLAN ID in the **Outer VLAN ID** field if necessary.
14. Select vlan_2 in the Container field in the IPv4 Static Hosts tab.
15. Repeat steps 6 - 14 to add additional subnets.
16. Click **Save** when you are done.

Adding a Test Interface

By default, the system provides four transmitting and/or receiving interfaces and one external interface (for SSL testing). Therefore, if you have a two-blade chassis, you will need to add additional interfaces to your Network Neighborhood.

Each test interface in the Network Neighborhood corresponds to a data port on the chassis. When you add an interface to a Network Neighborhood, the system will automatically number the interface based on the order in which it was added.

When you go to the test editor to create your test, the Network Neighborhood that you select will show all the interfaces that are available for it.

Note: If you delete any of the interfaces, the system will automatically resequence the interfaces. The successive interfaces (following the deleted interface) will be renumbered to the preceding interface's value (e.g., '6' will become '5').

To add a test interface to a Network Neighborhood:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select a Network Neighborhood from the **Network Neighborhood** list.
3. Click **Open**.
4. Click the arrow to expand the element to which you want to add an interface.
5. Click the **Add Row** button.
6. Once you have added the interface to the Network Neighborhood, you can modify the subnet information as desired.

Setting the MTU per Port

You can manually configure the maximum transmission unit, or MTU, for any port that you have reserved. The MTU refers to the largest packet size (in bytes) that can be transmitted. Currently, BreakingPoint supports MTU sizes of 46-9198 bytes. By default, the system will set the MTU to 1500.

To set the MTU for a port:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select the Network Neighborhood from the list and click Open.
3. Locate the Interface element.
4. Click the MTU field of the port you want to set.
5. Enter the desired MTU in the **MTU** field.
6. This value must be between 46-9198.

7. Click **Apply**.
8. Repeat steps 3-7 for each port whose MTU you would like to configure.

Virtual Routers and Hosts

There are two modes that each subnet can operate on:

- Host Mode
- Virtual Router Mode

Host Mode

In **Host** mode, BreakingPoint simulates a number of hosts on a network. Each IP address configured as a host will respond to ARP requests, and in some cases, ICMP Echo requests. The MAC address is derived from the IP address, where the host address is concatenated with the specified host Ethernet address.

Note: The host range must fit within the range of the network and netmask.

Virtual Router Mode

In **Virtual Router** mode, BreakingPoint acts as a virtual router existing on a particular network. You can specify the IP address and MAC address of the virtual router, and it will respond to ARP and ICMP Echo requests. The hosts range can be any range of your choice.

All host traffic will appear to come from the virtual router's MAC address at the Ethernet level. When operating in virtual router mode, you need to setup static routes on your DUT so that it knows to use the virtual router as the gateway to the configured host addresses.

Setting the Mode for Layer 2 Devices

The virtual router mode offers the best performance for layer 2 devices (e.g., switches). Virtual router mode ensures that only one Ethernet address will be used per port, which avoids problems with overflowing the MAC cache per port.

Since many devices can only track a limited number of MAC addresses per-port (i.e., 1 to 16), the DUT will go into broadcast mode for all ports in the broadcast domain if you configure a port in the Host mode and have more hosts configured than the size of the per-port MAC address cache of the DUT.

Although you may not want to overload the device’s per port MAC cache, this is a good way to expose defects in the device, such as packet leaking between broadcast domains (VLANs), decreased performance or even crashes.

Setting the Mode for Layer 3 Devices

In order to test Layer 3 devices, you should choose different modes per port. Generally, for these devices, you should be aware of the ARP cache. Since most devices have an ARP cache between a few hundred and a few thousand entries, you can easily overflow the ARP cache with a netmask of less than 24 in Host mode.

Setting the Mode for Edge Routers

In order to test edge routers, you should configure all interfaces on the BreakingPoint system using the Virtual Router mode.

Setting the Mode for Servers

Most likely, if you are testing a server, it is on a network with some other hosts and is reachable through a router for other hosts. Therefore, when you want to simulate locally connected hosts, you should use the Host mode for the interface. To simulate a default gateway with remotely connected hosts, configure a virtual router instead.

Setting the Mode for NAT, Firewalls, and Other Gateway Router Devices

To test a NAT, firewall or other gateway router devices, you should configure a private hosts network on the private interface, and a virtual router on the public interface.

Network Address Translation

The Network Address Translation (NAT) feature seamlessly addresses NAT scenarios using 2-Arm mode configurations. The primary function of the NAT feature in BPS is to identify which 2-arm flow a packet belongs to when it is received.

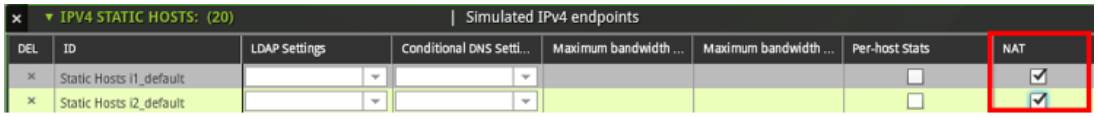
- Without NAT, incoming packets are matched to flows based on Source/Destination IP/Port tuples
- With NAT, the following can occur:
  - The DUT changes the IPs and/or ports in the packet headers (including the type v4<->v6)
  - The DUT mangles the L4 payload. For example, by changing IPs and/or ports embedded in the payload (FTP, SIP, H.323, RTSP, etc.)
  - The DUT terminates connections itself, and possibly initiates some connections on the behalf of the other side. We call this "Proxy" and we ignore it here because it is **not supported** with the NAT feature.

Starting with BPS 8.0.1, the NAT feature triggers a specific internal mechanism to enable interoperability with a DUT/SUT performing NAT.

Enabling NAT for a Subnet

To enable NAT for a subnet, select the **NAT** option as shown below.

Note: Enabling the NAT option will result in some impact in performance and should not be used in non-NAT scenarios.



Note: NAT support is available for the following test components: Application Simulator, Client Simulator and Session Sender.

Note: Because the **Behind NAT** option had no functional impact (starting with BPS release 3.4 and onwards) it has been removed (starting with BPS release 8.0.1). BPS 8.01 introduces a NAT feature which triggers a specific internal mechanism to enable interoperability with a DUT/SUT performing NAT. The **NAT** checkbox should be enabled on all Network Neighborhood elements for this type of test scenario, including: client side, server side, external host (for DNAT), etc. All configurations that were exported from versions prior to BPS 8.0.1 will have the **NAT** checkbox disabled (including those that had the **Behind NAT** option enabled) when the configuration is imported into BPS 8.0.1 or any following BPS version.

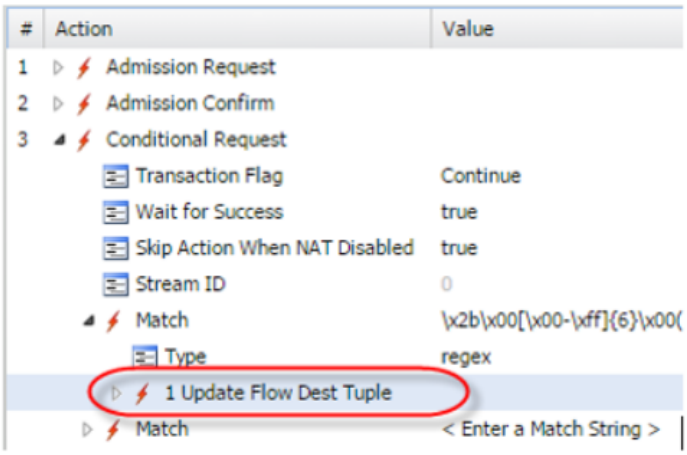
Note: See NAT Environment Options for information on creating Application Profiles and Super Flows that will be used in a test environment that includes NAT.

BPS NAT Advanced Configuration: Application Layer Gateway (ALG):

Specific protocols that embed the IP or ports information within the payload (and use them for initiating new connections) require the support of an Application Layer Gateway (ALG).

BPS NAT ALG support pre-existing requirements:

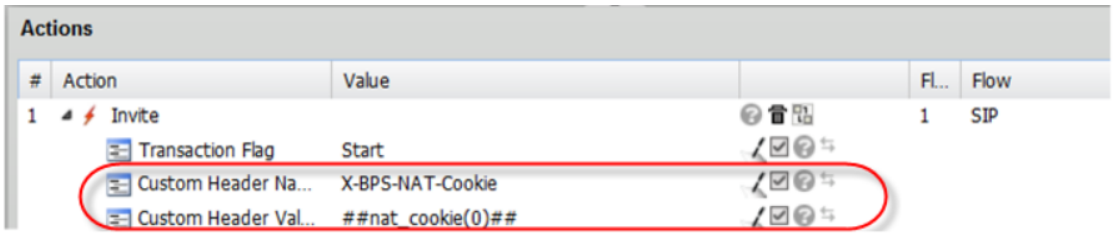
- TCP-based ALGs (FTP, SIP over TCP, RTSP, H.323 etc.)
  - **Conditional Requests** actions to parse and save the external IP and Port used by the device to accept public incoming connections
  - **Update Flow Dest Tuple/Port**action for subsequent connection initiated from the public side



Use the corresponding pre-canned "NAT" tagged Super Flows.

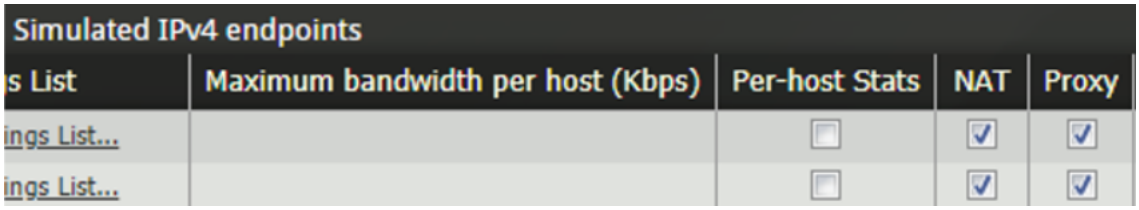
UDP-based ALGs (FTP, SIP over TCP, RTSP, H.323 etc.), pre-existing requirements:

- Previous pre-existing TCP-based ALGs requirements, plus:
- Custom token in the packet: **##nat_cookie(0)##**:
  - Used in the first UDP session packet in any L7 header
  - Required for Super Flows where the DUT is modifying the L7 data



Use the corresponding pre-canned "NAT" tagged superflows.

NAT/Proxy Interoperability



When both **NAT** and **Proxy** are selected for the same subnet, the following occurs:

- The Super Flows having "Proxy" support will be passed to the Proxy internal mechanism
- The other superflows will be treated as going through a DUT/SUT performing **NAT**

External Interface Addressing

All addressing for end-point testing (i.e., one-arm security testing or SSL server testing) can be configured through the External element in the Network Neighborhood. The addressing information defined here allows the Session Sender, Security, and Stack Scrambler test components to address a specific destination address (or range of addresses) through the test port of the DUT. The DUT, in this case, would act as the server, and the BreakingPoint system would act as the client.

BreakingPoint allows you to set up multiple address ranges per subnet, which provides you with greater flexibility over the IP addresses that are used.

Note: External device addressing does not support VLAN tagging on the Security Component.

When to Use the External Interface

You should use an external interface only if you want BreakingPoint to target a host that is not being simulated by the BreakingPoint system itself. Two widely used use cases are load-testing a standalone machine and targeting a virtual device.

If you want to test a particular device as a default gateway on the network, you do not need to use an external interface. You will need to simply configure the gateway in the subnet for the interfaces on which it is connected.

The only test components for which you should define an external interface are:

- Application Simulator
- Client Simulator
- Session Sender
- Recreate
- Security

These are the only components that can do one-arm TCP/IP and produce a tangible pass/fail result. Potentially, any test could target a standalone host, namely Stack Scrambler or Routing Robot; however, these tests rely on receiving their own packets to determine if the test passes or fails, so they will always fail if they target an external host.

You can always connect the client interface to your target device if you want to target an external device, but the default test criteria may not provide useful data.

Additionally, if the destination IP address is not specified as external explicitly, BreakingPoint will not ARP for its MAC address, instead it will use the internally generated one. In order to work around this, you will need to enter the DUT's MAC address in the provided field for the subnet and use virtual router mode. This is equivalent to hard coding a MAC address in the ARP table.

Note: BreakingPoint does not check for overlaps between IP addresses in an external interface and IP addresses configured as virtual hosts on other ports. This can lead to some unexpected results, as there will be multiple devices bound to the same IP address if the external and the affected subnet are used in the same test.

Setting up the External Interface

The following provides instructions on setting up the external interface.

To set up the External interface:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Locate the Network Neighborhood you want to use from the **Network Neighborhood** list.
3. If the Network Neighborhood you want does not appear in the list, enter a portion of the name of the Network Neighborhood into the **Filtered Search** field and click the **Search** button.
4. Select a Network Neighborhood and click Open.
5. Click the arrow to expand the **External** Host tab.
6. Enter a minimum IP address in the **Base IP Address** field.

   Note: The minimum address is the lowest value in the address range for the subnet. Use the format: x.x.x.x, where x can be a value between 0-255.
7. Enter the number of IP addresses to be used in the Count field.
8. Click the **Add Row** button to add an additional IP address range.

9. Repeat steps 4 - 8 to add additional address ranges.
10. Click **Save**.

IPsec Feature Specifications

The IPsec Feature Specifications table shown below provides details about the IPsec features that are supported on the BreakingPoint system.

Supported IPsec Features

Feature	Details	Notes and Exceptions
Keying Methods	IKEv1 IKEv2	
IPsec parameters IKE phase1/AUTH_SA	IKEv1 Main / Aggressive IKE mode Hash Algorithms: o HMAC-MD5 o HMAC-SHA1 o AES-XCBC-MAC-96 (RFC3566) o HMAC-SHA256 o HMAC-SHA384 o HMAC-SHA512 Encryption algorithms: o DES o 3DES o AES-128-CBC o AES-192-CBC o AES-256-CBC o AES-128-GCM ICV 8/12/16 o AES-192-GCM ICV 8/12/16 o AES-256-GCM ICV 8/12/16 IKEv1 XAUTH user authentication ModeCFG address assignment Pseudo-random functions (PRF) o HMAC-MD5 o HMAC-SHA1 o AES-XCBC o HMAC-SHA256 o HMAC-SHA384 o HMAC-SHA512	GCM ICV is only supported for IKEv2

Feature	Details	Notes and Exceptions
IPsec parameters Phase 2/CHILD_ SA	ESP Tunnel Mode Hash algorithms: o HMAC-MD5-96 o HMAC-SHA1-96	Only supported for IKEv2
	o HMAC-SHA256-128 o HMAC-SHA384-192 o HMAC-SHA512-256	Only supported on the PerfectStorm platform
	Encryption algorithms: o NULL o DES and 3DES o AES-128-CBC o AES-192-CBC o AES-256-CBC	
	o AES-128-GCM ICV 8/12/16 o AES-192-GCM ICV 8/12/16 o AES-256-GCM ICV 8/12/16	Only supported on the PerfectStorm platform
	Perfect Forward Secrecy (PFS) Lifetime negotiation and re-keying	
Authentication Method	Pre-shared key	

Feature	Details	Notes and Exceptions
DH Groups	DH-01 (MODP-768) DH-02 (MODP-1024) DH-05 (MODP-1536) DH-14 (MODP-2048) DH-15 (MODP-3072) DH-16 (MODP-4096) DH-17 (MODP-6144) DH-18 (MODP-8192) DH-19 (ECP-256) DH-20 (ECP-384) DH-21 (ECP-512) DH-22 (MODP-1024-S160) DH-23 (MODP-2048-S224) DH-24 (MODP-2048-S256) DH-25 (ECP-192) DH-26 (ECP-224)	
IPsec Features	Site to Site and Remote Access test scenarios IPsec Initiator Initial Contact Payload NAT Traversal (NAT-T) (IKEv2 only) VLAN support	
Tunnel Control	Tunnel setup and tear down IPsec tunnel flapping (dynamic sessions) Re-keying support IKE message retry timers Lifetime negotiation Re-keying Dead Peer Detection (DPD) (IKEv2 Only) NAT Traversal (NAT-T) (IKEv2 only)	
Addressing	IPv4/IPv4 Single host per emulated gateway	

Currently, the IPsec features described in the table below are NOT supported by the BreakingPoint system.

Unsupported IPsec Features

Feature	Details
Keying Methods	Manual keying
IPsec parameters Phase 2/CHILD_SA	Transport mode Encryption algorithms: o AES-128-GMAC o AES-192-GMAC o AES-256-GMAC IKEv1: Multiple Phase2 SAs over a single Phase1 SA IKEv2: Multiple ChildSAs over a single IKE SA
Authentication Method	RSA and ECDSA Certificates EAP (MD5, SIM, TLS, AKA) EAP vs. PreSharedKeys EAP vs. Certificates
Certificate Management	SCEP (Simple Certificate Enrollment Protocol) CMPv2 (Certificate Management Protocol) CRL (Certificate Revocation List) OSCP (Online Status Certificate Protocol)
IPsec Features	IPsec responder modes IPsec pre-fragmentation IPsec post-fragmentation
Addressing	IPv6/IPv6 IPv4/IPv6 IPv6/IPv4 Multiple hosts per emulated gateway Unique VLANs per emulated gateway Multicast traffic over IPsec tunnel

IPsec Overview

This section provides a general overview of the IPsec protocol. The purpose of IPsec is to provide privacy, encryption, and data integrity for network traffic traveling over an insecure network, such as the Internet. For details on the IPsec features supported on the BreakingPoint system, IPsec Feature Specifications on page 159

IPsec is used in two basic topologies:

- Site To Site
- Remote Access

Site To Site

A site-to-site IPsec topology contains two sites connected through a pair of IPsec Secure Gateways. The LANs at each location are presumed to be secure and the insecure segment between the Secure Gateways is secured through the use of the tunnel. IPsec tunnel mode is used in site-to-site topologies.

Remote Access

In a Remote Access IPsec topology, the client is operating as its own Secure Gateway.

Note: A security gateway is an intermediate system that implements IPsec protocols. For example, a router implementing IPsec is a security gateway.

IPsec Protocols: AH and ESP

IPsec uses two protocols to provide traffic security:

- Authentication Header (note that BreakingPoint does not support AH)
- Encapsulating Security Payload

Authentication Header

Authentication Header (AH) provides connectionless integrity, data origin authentication, and an optional anti-replay service.The AH header uses a cryptographic checksum over a portion of the packet to ensure that the packet has not been modified during transit.

Encapsulating Security Payload

Encapsulating Security Payload (ESP) may provide confidentiality (encryption), and limited traffic flow confidentiality. It also may provide connectionless integrity, data origin authentication, and an anti-replay service. (One or the other set of these security services must be applied whenever ESP is invoked.) The ESP header introduces a portion of the original packet that has been encrypted.

Either header may be used separately or both may be used at the same time. Each protocol supports two modes of use:

- Transport Mode, in which the protocols provide protection primarily for upper layer protocols.
- Tunnel Mode, in which protocols are applied to tunneled IP packets.

Transport Mode

In IPsec transport mode, two peers authenticate each other in phase one, and establish the traffic signing and encryption parameters in phase two. Transport mode leaves the original header intact and adds an AH or ESP header.

The AH header includes a cryptographic checksum over the entire packet. The receiving end can verify that the entire packet was received without error or modification.

The ESP header also includes a full packet cryptographic checksum, and in addition, the packet's payload section is encrypted.

Transport mode is only used in Remote Access connections, where the source of the packets is also the security gateway.

IPsec Tunnel Mode

IPsec tunnel mode is used for secure site-to-site communications over an untrusted network. Each site has an IPsec gateway configured to route traffic to the other site. In tunnel mode, the original packet is encapsulated into a new packet which includes a new header and AH and/or ESP headers.

The AH header is used to authenticate the entire packet. The ESP header is used to encrypt and authenticate the original packet. When used in combination, the original packet is encrypted and the entire packet is authenticated.

Tunnel mode is used when a security gateway is used to perform IPsec operation on behalf of a client computer, as is the case in LAN-to-LAN IPsec networks. The use of both AH and ESP headers provides maximum protection.

Cryptographic Key Distribution

Because security services use shared secret values (cryptographic keys), IPsec relies on a separate set of mechanisms for putting these keys in place. IPsec mandates support for both manual and automatic distribution of keys:

- **Manual Key Management:**

  The simplest form of key management is manual management, in which a person manually configures each system with keying information and security association management data, to support secure communication with other systems. Manual keying techniques are practical in small, static environments but they do not scale well.
- **Automated SA and Key Management:**

  Widespread deployment and use of IPsec requires an Internet-standard, scalable, automated, SA management protocol. Such support is required to facilitate use of the anti-replay features of AH and ESP, and to accommodate on-demand creation of SAs (for user- and session-oriented keying, for example).

  The default automated key management protocol selected for use with IPsec is Internet Key Exchange (IKE). There are two versions of IKE in widespread use. Refer to IKEv1 Overview and IKEv2 Overview for more information.

  These keys are used for authentication/integrity and encryption services.

IPsec Security Associations

A Security Association (SA) is a relationship between two or more entities that describes how the entities will utilize security services to communicate securely. It is a construct that provides unidirectional security services to inbound or outbound IP traffic. An SA associates IP packets with specific security services. The IPsec security services are made available to an SA by the use of AH or ESP, but not both. If both AH and ESP protection is applied to a traffic stream, then two (or more) SAs

are required (one for AH and one for ESP). To secure bi-directional communication between two hosts, or between two security gateways, at least two SAs (one in each direction) are required.

A SA is uniquely identified by:

- A Security Parameter Index (SPI): a unique number that identifies the session
- The destination IP address
- A security protocol (AH or ESP) identifier.

Both AH and ESP use SAs, and a major function of IKE is the establishment and maintenance of SAs.

IKE 1 Overview

The purpose of the IKE protocol is to set up the parameters that allow two IPsec endpoints to communicate securely with each other. The set of parameters is called a Security Association (SA). SAs can be uni-directional or bi-directional.

The negotiation process between IPsec endpoints involves one party acting as an initiator and the other acting as a responder. Where parameters are being negotiated, the initiator offers the set of authentication, encryption, and other techniques that it is ready to use with the other endpoint. The responder tries to match this list against its own list of supported techniques. In response to the initiator's offer, the responder sends a single set of parameters; these are the parameters that the responder accepts from the offer. The initiator and responder then proceed with the negotiated settings.

IKE negotiation is broken down into two phases:

- **Phase 1** - Allows two gateways (one of which may be a client acting as its own gateway) to authenticate each other and establish communications parameters for phase 2 communications.
- **Phase 2** - Allows two gateways to agree on IPsec communications parameters on behalf of sets of hosts on either side of the gateway.

The IKEv2 protocol is described in RFC 4306. Authentication using certificates is described in RFC 2409 (for IKEv1) and RFC 4306 (for IKEv2).

IKEv1 Negotiation - Phase 1

During IKEv1 phase 1 negotiation, two endpoints authenticate each other. That is, based on policies enforced at each end they decide that the other party is to be trusted. The means by which two parties trust each other can be based on one or more of the following:

Authentication methods:

- Pre-shared key
- Public key cryptography, using X.509 certificate

Identification methods:

- ID_IP_ADDR (IP address)
- ID_IP_ADDR_SUBNET (IP address and subnet mask)
- ID_FQDN (fully-qualified domain name)

- ID_USER_FQDN (fully-qualified username)
- ID_DER_ASN1_DN

The endpoints also agree on the particular authentication and encryption algorithms to use when exchanging their later stage phase 1 and all phase 2 messages. Two endpoints use a single bi-directional SA at the end of their phase 1 negotiation. There are two phase 1 negotiation modes:

- **Aggressive Mode** - Three messages are exchanged to create a bi-directional ISAKMP SA. The identity of the peers is sent in clear text.
- **Main Mode** - Six messages are exchanged to create a bi-directional ISAKMP SA. The last two messages are encrypted, thereby hiding the identify of the peers. Main Mode is also referred to as Identity Protection Mode.

A single phase 1 ISAKMP SA may be used to establish any number of phase 2 ISAKMP SAs. During the negotiation process the two endpoints generate a shared secret that is used to encrypt their communications. This shared secret is generated using public-private key cryptography in which two parties can generate a common data string without explicitly transmitting that data.

IKE Negotiation - Phase 2

The objective of phase 2 negotiation is to establish unidirectional SAs used by the two gateways. These SAs are used to transmit IPsec packets to each other on behalf of hosts attached to their local LANs. Phase 2 messages operate under the protection of a phase 1 SA, using the negotiated shared secret between the gateways. In addition to negotiating authentication and encryption parameters, they also contribute random data to be used in generating the keys for the encryption algorithms that encrypt the payload data.

Phase 2 SAs are unidirectional; that is, each gateway initiates a negotiation for an SA that it will use to send data to its partners. The set of parameters

IKEv2 Overview

IKE Version 2 simplifies the IKE protocol. IKEv2 is designed for IPsec tunnel negotiation and brings increased security, lower overhead, and increased flexibility versus the initial IKEv1 negotiation method.

The main differences between IKEv1 and IKEv2 are:

- Simplified initial exchange – In IKEv2, the initial contact between peers is accomplished using a single exchange of four messages. IKEv1 provides a choice of eight separate exchange mechanisms.
- Reduced setup latency – the initial exchange of two round trips (four messages), coupled with the ability to simultaneously set up a child Security Association (SA) on the back of that exchange, reduces setup latency for most common setup scenarios.
- Fewer header fields and bits – The Domain of Interpretation (DOI), Situation (SIT), and Labeled Domain Identifier fields have been removed in IKEv2, as have the Commit and Authentication Only bits.
- Fewer cryptographic mechanisms – IKEv2 protects its own packets with an ESP-based mechanism very similar to the one it uses to protect IP payloads, simplifying implementation and security analysis.

- Increased reliability – In IKEv2, all messages must be acknowledged and sequenced (in IKEv1, message IDs are random), which reduces the number of possible error states.
- Resistance to attacks – To better resist attacks, an IKEv2 host does not do much processing until it has satisfied itself that a potential peer is authentic. IKEv1 is vulnerable to DoS attacks (attack by causing excessive processing) and spoofing (access using a forged address).

For detailed information, refer to RFC 5996, Internet Key Exchange (IKEv2) Protocol.

IKEv2 Initial Exchanges

Communication between IKEv2 peers begins with exchanges of IKE_SA_INIT and IKE_AUTH messages (in IKEv1, this is known as Phase 1). These initial exchanges normally consist of four messages, although there may be more for some scenarios. All IKEv2 message exchanges consist of request/response pairs.

- IKEv2 Server A (Initiator) -> IKE_SA_INIT Request -> IKEv2 Server B (Responder)
- IKEv2 Server A (Initiator) <- IKE_SA_INIT Response <- IKEv2 Server B (Responder)
- IKEv2 Server A (Initiator) -> IKE_AUTH Request -> IKEv2 Server B (Responder)
- IKEv2 Server A (Initiator) <- IKE_AUTH Response <- IKEv2 Server B (Responder)

The first pair of messages (IKE_SA_INIT) negotiates the cryptographic algorithms to be used, exchange nonces, and exchange Diffie-Hellman values.

The second pair of messages (IKE_AUTH) authenticates the previous messages, exchange identities and certificates, and establish the first Child SA. Parts of these messages are encrypted and have their integrity protected using keys established through the IKE_SA_INIT exchange, to hide the peers' identities from eavesdroppers. Furthermore, all fields in all messages are authenticated.

Initiator to Responder

The initial exchange (IKE_SA_INIT Request) begins with the initiator sending the following to the responder:

- An IKE header that contains the Security Parameter Indexes (SPIs), version numbers, and has various flags set or unset;
- A payload listing the cryptographic algorithms that the initiator supports for the IKE SA;
- A payload containing the initiator's Diffie-Hellman shared secret;
- A payload containing the initiator's nonce (a random or pseudo-random number that is used only once in a session).

Responder to Initiator

The responder replies to the initiator with an IKE_SA_INIT Response:

- A payload naming the cryptographic suite selected by the responder from those offered by the initiator;
- A payload containing the responder's Diffie-Hellman shared secret;
- A payload containing the responder's nonce value;
- Optionally, the responder may send a certificate request as well.

At this point in the negotiation, each peer uses the nonces and Diffie-Hellman values to generate the seed values to be used in turn to generate all the keys derived for the IKE SA. Keys are generated for encryption and integrity protection (authentication); separate keys are generated for each function in each direction.

An additional value is derived from the Diffie-Hellman values, to be used to generate keys for child SAs.

Beyond this point, all parts of the messages exchanged between the peers are encrypted and authenticated, except for the headers.

Initiator to Responder

In the next series of exchanges (IKE_AUTH exchanges), the initiator asserts its identity, proves that it knows the secret corresponding to identity and integrity protects the contents of the first message using the AUTH payload. If a certificate was requested, it may return the certificate and a list of its trust anchors. If it does send a certificate, the first certificate provided contains the public key used to verify the AUTH field. At this stage, if the responder hosts multiple identities at the same IP address, the initiator can specify with which of the identities it wants to communicate. The initiator next begins negotiating a child SA.

Responder to Initiator

The responder replies by asserting its own identity, optionally sending one or more certificates (again with the certificate containing the public key used to verify AUTH listed first), authenticates its identity and protects the integrity of the second message with the AUTH payload, and completes negotiation of a Child SA.

IKEv2 Child SAs

A Child SA in IKEv2 is the equivalent of a a Phase 2 exchange in IKEv1. Activation of the first Child SA under an IKE_SA is handled slightly differently than activation of subsequent Child SAs under that same IKE SA. (Because either endpoint can initiate a Child SA, the term initiator in the context of a Child SA exchange refers to the endpoint that initiates the Child SA.)

First Child SA

The IKEv2 protocol was designed such that the first Child SA is activated during processing of the IKE_AUTH request and response message exchange. The first Child SA establishes the parameters for using ESP, AH, and IPComp.

The IKE_AUTH request contains the initiator's list of SA proposals and the traffic selectors that describe the traffic to be protected by the Child SA. However, the IKE_AUTH request does not contain keying information or a nonce that is specific to the Child SA. The nonces and keying information from the IKE_SA_INIT exchange are used in computing the keys for the first Child SA.

Subsequent Child SAs can be initiated to create a new IPsec SA or to perform rekeying of the IKE SA in two messages.

Additional Child SAs

Each additional Child SA is established using a single CREATE_CHILD_SA request/response exchange. The initiator sends a CREATE_CHILD_SA request, containing a list of proposals for the Child SA. Each

proposal defines an acceptable combination of attributes for the Child SA that is being negotiated (AH or ESP SA). The responder picks a proposal that is acceptable and returns the choice to the initiator in the CREATE_CHILD_SA response. The attributes that can be negotiated include the following:

- Protocol (AH or ESP)
- Authentication algorithm
- Encapsulation mode (tunnel or transport)
- Encryption algorithm
- Diffie-Hellman group information

The portion of the Child SA message after the header is encrypted, and the entire message (including the header) is integrity protected (authenticated) using the cryptographic algorithms negotiated for the IKE SA.

Requesting Internal Addresses on Remote Networks

IKEv2 includes a mechanism for external hosts to obtain a temporary IP address for a host on a network protected by a security gateway. This mechanism, described in section 2.19 of RFC 4306, involves adding a Configuration Payload (CP) request to Child SA request.

When a security gateway receives a CP request for an address, it can either obtain an address from an internal pool or it may query external servers (such as DHCP or BOOTP servers) to obtain the address. To return the address, the gateway returns a CP reply.

This mechanism provides IKEv2 with functionality similar to XAUTH and MODE-CFG in IKEv1.

Deleting a Child SA

Deleting an IKE SA automatically deletes all Child SAs based on it; deleting a Child SA deletes only that Child SA.

Initial Contact Notification Message

An IKE SA can use an INITIAL_CONTACT notification message to assert that it is the only SA currently active between the authenticated identities. It may be sent when an SA is established following a crash, thereby allowing the recipient to delete any other IKE SAs it has to the same authenticated identity, rather than waiting for a timeout.

IKEv2 Authentication Methods

IKEv2 supports the following authentication methods:

- Public key signatures
- Shared secrets
- Extensible Authentication protocol (EAP)

IKEv2 Support for SA Rekeying

Rekeying refers to the re-establishment of SAs to replace SAs that have expired or are about to expire. If attempts to rekey an SA fail, the SA and its Child SAs are terminated. The peers can then negotiate new SAs.

To improve performance and reduce the potential number of lost packets, most IKE v2 implementations allow SAs to be rekeyed before they expire (in-place rekeying).

To rekey a Child SA within an existing SA, a new, equivalent Child SA is created and the old one is deleted.

To rekey an SA, a new equivalent SA is created with the peer. The new SA inherits all of the original SA's Child SAs, and the old SA is deleted by sending a message containing a "Delete" payload over it. The Delete payload is always the last request sent over an SA that terminates normally.

In IKEv1, peers negotiated SA lifetimes with each other. In IKEv2, each peer selects its own lifetime for an SA, and is responsible for rekeying the SA when necessary. If the two peers select different lifetimes, the peer that selects the shorter lifetime initiates rekeying.

If an SA and its child SAs have carried no traffic for a long time and if its endpoint would not have initiated the SA without any traffic for it, the endpoint may close the SA when its lifetime expires, instead of rekeying it.

Some IKE peers may impose a random delay before initiating rekeying. This is done to prevent a collision-like situation in which both peers select identical lifetimes for an SA, and then simultaneously attempt to rekey it, resulting in duplicate SAs.

IKEv2 does not prohibit duplicate SAs. RFC 4306 states that endpoints can establish multiple SAs between them that have the same traffic selectors in order to apply different traffic quality of service (QoS) attributes to the SAs.

BreakingPoint Support for RFC 5685

The BreakingPoint IPsec plug-in supports these features of the redirect mechanism:

- Using the redirect mechanism during the IKEv2 initial exchange (the IKE_SA_INIT exchange)
- Using anycast addresses with the redirect mechanism
- Using the redirect mechanism during an active session (the VPN gateway to redirects the client to another VPN gateway in the middle of a session)
- Using the redirect mechanism during the IKE_AUTH exchange
- Prevention of redirection loops, using MAX_REDIRECTS) within a designated time period (REDIRECT_LOOP_DETECT_PERIOD) for a particular IKEv2 SA setup
- The following redirect payloads: REDIRECT_SUPPORTED, REDIRECT, REDIRECTED_FROM
- IKEv2 Cookies Support

IKEv1 supported cookies, and IKEv2 continues that support.

Internet Security Association and Key Management Protocol (ISAKMP) fixed message header includes two eight-octet fields titled "cookies", and that syntax is used by both IKEv1 and IKEv2, though in IKEv2, they are referred to as the IKE SPI and there is a new separate field in a Notify payload holding the cookie.

IP Compression (IPcomp)

IP Payload Compression Protocol (IPComp) is designed to improve the performance of communications between hosts by reducing the size of the IP datagrams sent between them. IPComp supports a number

of compression algorithms. BreakingPoint supports DEFLATE, an LZ77-based compression method.

IPsec peers can negotiate the use of IPComp as part of the setup of a Child SA. A peer requesting a Child SA can advertise that it supports one or more IPComp compression algorithms. The other peer indicates its agreement to use IPComp by selecting one of the offered compression algorithms.

IPsec NAT-T

NAT-T (Network Address Translation Traversal) was developed to address the problem of using IPsec over NAT devices. Because NAT devices modify addresses in the IP header of packets, these packets fail the checksum validation when IPsec is in use. To IPsec, the packets appear to have been modified in transit, something IPsec is intended to prevent.

NAT-T detects the presence of NAT devices between two hosts, switches the IPsec function to a non-IPsec port, and encapsulates the IPsec traffic within UDP packets. To preserve the original source and destination port numbers, NAT-T inserts an additional header containing the port numbers between the IP header and the ESP header. For example, after IKE peers initiate negotiation on port 500, detect support for NAT-T, and detect a NAT device along the path, they can negotiate to switch the IKE and UDP-encapsulated traffic to another port, such as port 4500 (the BreakingPoint IPsec plug-in listens on port 4500 to establish a connection for IKEv2.).

You configure NAT-T on a per-range basis, using a checkbox in the IKE Phase 2 tab.

- The main RFCs for NAT-T implementations are:
- RFC 3715, IPsec-Network Address Translation (NAT) Compatibility Requirements
- RFC 3947, Negotiation of NAT-Traversal in the IKE
- RFC 3948, UDP Encapsulation of IPsec ESP Packets

IPsec MODECFG

MODECFG (Mode-configuration) is an IPsec feature that functions like DHCP for IPsec clients. It enables the client to obtain address information (such as a private IP address, a netmask, a DNS server IP address, and so forth) from an initiator. There are two modes:

- **Push**: A responder sends (pushes) address information to an initiator.
- **Pull**: A client retrieves (pulls) address information from a server.

MODECFG is typically used in remote-access scenarios, where addresses may be part of a pool, with different privileges given to different addresses, or groups of addresses. The responder (the device supplying addresses) sends the addresses during the IKE key exchange.

The MODECFG options are negotiated between phase 1 and phase 2. For this reason, MODECFG is also referred to as Phase 1.5.

The MODECFG exchanges are done in IKE transaction packets. IPsec XAUTH IKE Extended Authentication (XAUTH) is an enhancement to the existing Internet Key Exchange (IKE) Protocol. XAUTH was developed to leverage legacy authentication schemes (such as RADIUS) with IKE. Whereas IKE performs device authentication, XAUTH performs user authentication. XAUTH user authentication occurs after IKE authentication phase 1, but before IKE IPsec SA negotiation phase 2. With XAUTH, once a device has been authenticated during normal IKE authentication, IKE can then also authenticate the user of that device.

When XAUTH is active, a user accessing the network must provide a username and a password for authentication. Additionally the user can be identified as belonging to a group specified by the group name.

The XAUTH exchanges are done in IKE transaction packets.

Major IPsec RFCs

The major RFCs that define the IPsec protocols are as follows:

- RFC 2407, The Internet IP Security Domain of Interpretation for ISAKMP
- RFC 2408, Internet Security Association and Key Management Protocol (ISAKMP)
- RFC 2409, The Internet Key Exchange (IKE)
- RFC 3715, IPsec-Network Address Translation (NAT) Compatibility Requirements
- RFC 3947, Negotiation of NAT-Traversal in the IKE
- RFC 3948, UDP Encapsulation of IPsec ESP Packets
- RFC 4301, Security Architecture for the Internet Protocol
- RFC 4302, IP Authentication Header
- RFC 4303, IP Encapsulating Security Payload (ESP)
- RFC 4308, Cryptographic Suites for IPsec
- RFC 5985, Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2)
- RFC 5996, Internet Key Exchange (IKEv2) Protocol

Test Paths

Typically, when you create a test, you have to specify the client and server interfaces that each component will use to transmit and receive traffic. BreakingPoint will automatically map these interfaces for you by creating different client and server pairings.

For example, if you have selected Interface 1 and 3 as the client interfaces and Interface 2 and 4 as the server interfaces, then you will have the following possible client/server pairings:

- 1 > 2
- 1 > 4
- 3 > 2
- 3 > 4

However, there may be cases in which you will want to manually define the client and server pairings. This is where Test Paths are useful.

Test Paths enable you to explicitly specify the interfaces with which other interfaces can communicate. This feature becomes extremely useful in cases where you want to ensure that all your connections succeed.

Let’s reuse the test configuration mentioned in the previous example. Now, let’s say that you have loopback cables connected from Interface 1 to 2 and from Interface 3 to 4.

In this particular case, the connections between Interfaces 1 and 4 and Interfaces 2 and 3 will fail because there are not any physical connections between those interfaces. Only the connections between Interface 1 and 2 and Interface 3 and 4 will work because they are physically connected.

Therefore, to ensure a 100% success rate between all your connections, you will want to explicitly define the valid connections between your interfaces.

To define Test Paths:

1. Select **Control Center > Open Neighborhood** from the menu bar.
2. Select an existing Network Neighborhood from the **Network Neighborhoods** list or create a new Network Neighborhood.
3. Click **Open**.
4. Click the **Test Paths** button.
5. Select the **Specifically defined test paths** option.
6. Select the interfaces that are connected by choosing an interface from each drop-down menu.
7. Click the **Add** button.
8. Repeat steps 5 - 8 for each additional Test Path you would like to add.
9. Click **Apply** then click **Close**.

Asymmetrical Test Paths

With Test Paths, you can create test cases for asymmetrical network configurations. Asymmetrical network configurations refers to any system in which the data speed or quantity differs in one direction as compared with the other direction, averaged over time. Asymmetrical data flow can, in some instances, make more efficient use of the available infrastructure than symmetrical data flow, in which the speed or quantity of data is the same in both directions, averaged over time.

When a client on Interface 1 sends a packet to a server on Interface 2, the server should reply with Interface 4 to the client. The reply should be accepted by that client on Interface 3.

Proxy Support

BPS supports 2 Proxy Implementations:

- Transparent Proxy –This proxy feature (introduced in BPS 8.20), addresses Transparent Proxy scenarios. It utilizes a 2-arm BPS configuration. Network Elements have a check box that can be selected to enable internal mechanisms to properly identify flows that are being expected to pass through a transparent proxy environment. See Transparent Proxy Feature for details.
- Explicit Proxy/Load Balancers – This proxy configuration addresses Explicit Proxy/Load Balancer scenarios. It utilizes a dual 1-arm BPS configuration. See Explicit Proxy/Load Balancer Configuration for details.

Transparent Proxy Feature

The Proxy feature enables seamless traversal for HTTP and SSL flows through transparent proxy devices. Transparent Proxy support is also available for the following flows: IMAPv4-Advanced, SMTP and POP3-Advanced. For IMAPv4-Advanced, SMTP and POP3-Advanced, this functionality is achieved

by determining the end of a protocol specific message (delimiter), in order to keep the client and server synchronized (see the Flow action support tables shown below).

Note: This feature is only supported on Ixia CloudStorm and PerfectStorm.

The Proxy feature applies to a subset of Super Flows (described below) that have been tagged with "Proxy". The Proxy check box should be enabled for all proxy scenarios. The option enables specific internal mechanisms to properly identify flows that are being expected to pass through a transparent Proxy environment. Enabling the Proxy option will result in a some impact in performance and should not be used in non-Proxy scenarios.

The following table describes the Network Neighborhood Element support for the Proxy feature.

Network Neighborhood Element	Proxy Support
IPv4/IPv6 Static Hosts	Yes
IPv4/IPv6 External Hosts	Yes
VLAN	Yes
IPv4/IPv6 Router	Yes
DHCPv4/v6 (client/server)	No
IPv4 DNS	Yes
IPv6 DNS	Yes
GTP/LTE	No
6rd/DSlite	No
IPsec	No

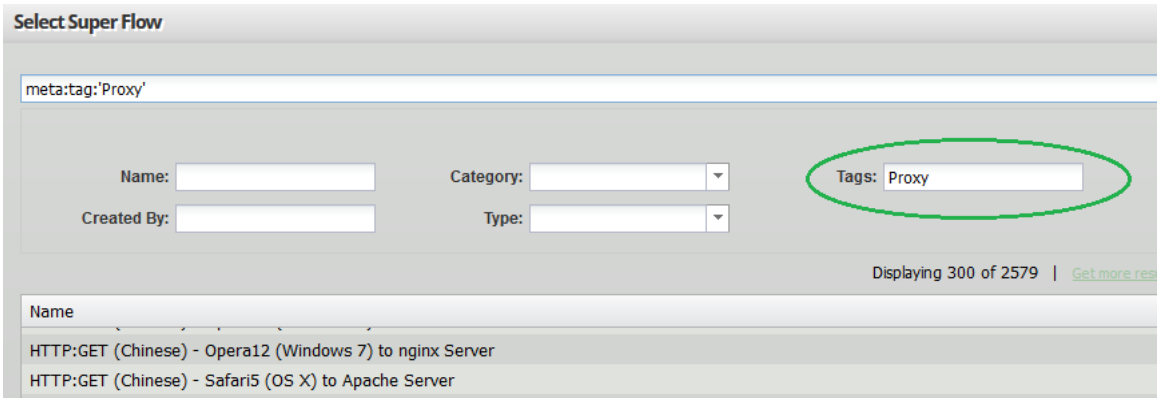
The following table describes the Test Component support for the Proxy feature.

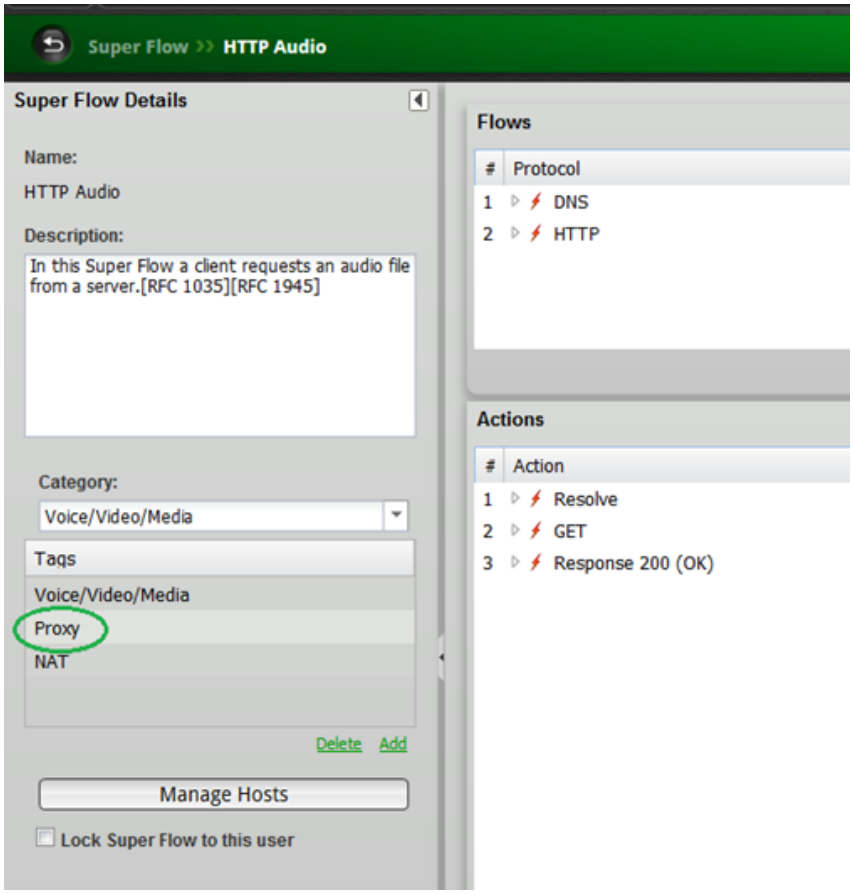
Test Component	Proxy Support
Application Simulator	Yes
Bit Blaster	No
Client Simulator	Yes (2-arm mode)
Recreate	No
Routing Robot	No
Security	No

Test Component	Proxy Support
Malware	No
Session Sender	No
Stack Scrambler	No

Proxy Tagged Super Flows

All pre-canned Super Flows that have been validated in transparent proxy scenarios are “Proxy” tagged as shown in the examples below.





General Guidelines for Proxy-compatible Super Flows

Manual intervention in order to sync client/server actions is not compatible with Proxy. For example, Conditional Requests, Match Content, Multi-match, etc.

Sequentially placed data nodes on the same side (client/server) is not compatible with Proxy. For example, HTTP Pipelining (multiple HTTP requests are sent without waiting for the corresponding responses).

Creating Custom Super Flows with Proxy Support

Each Super Flow contains the protocols that can be used to set up flows, server and client configurations, and the sequence of actions that will occur between the server and the client.

Note: When creating custom Super Flows based on HTTP, a ##proxy_cookie(0)## gets automatically embedded in the HTTP header (no need for manual intervention). Note that this support is not available for HTTP Raw Messages.

Note: When creating custom Super Flows based on IMAPv4-Advanced, SMTP or POP3-Advanced flows, the following constraints exist for Proxy support:

- Not compatible with additional syncing mechanisms. For example, Conditional Requests and Match Conditions.
- Proxy support was not extended to the following actions:
  - Raw Message, Custom Command
  - SSL: Start TLS, Accept TLS actions

IMAP

The following table describes the canned Super Flows that have support for the Proxy feature:

#	Super Flow
1	Bandwidth IMAPv4
2	IMAPv4-Advanced
3	IMAPv4-Advanced (Lawful Intercept)
4	IMAPv4-Advanced-Chinese
5	IMAPv4-Advanced-French
6	IMAPv4-Advanced-French
7	IMAPv4-Advanced-German
8	IMAPv4-Advanced-Italian
9	IMAPv4-Advanced-Japanese
10	IMAPv4-Advanced-Persian
11	IMAPv4-Advanced-Spanish
12	Financial IMAPv4-Advanced
13	IMAP Cisco EMIX
14	IMAP LTE Mix
15	IMAP RMIX 4K
16	IMAP RMIX 16K
17	IMAP RMIX 90K

The following table describes the flow action support for the Proxy feature.

#	Flow	Action	Delimiter
1.	IMAPv4-Advanced	Banner	\r\n
2.	IMAPv4-Advanced	Capabilities Request	\r\n

#	Flow	Action	Delimiter
3.	IMAPv4-Advanced	Capabilities Response	OK CAPABILITY completed\r\n
4.	IMAPv4-Advanced	Authenticate (Command)	\r\n
5.	IMAPv4-Advanced	Login OK	OK LOGIN completed\r\n
6.	IMAPv4-Advanced	Login NO	\r\n
7.	IMAPv4-Advanced	Authenticate: Login	\r\n
8.	IMAPv4-Advanced	Prompt for Username	\r\n
9.	IMAPv4-Advanced	Username	\r\n
10.	IMAPv4-Advanced	Prompt for Password	\r\n
11.	IMAPv4-Advanced	Password	\r\n
12.	IMAPv4-Advanced	Authentication OK	\r\n
13.	IMAPv4-Advanced	Authentication NO	\r\n
14.	IMAPv4-Advanced	Select Request	\r\n
15.	IMAPv4-Advanced	Select Response	OK [READ-WRITE] SELECT completed\r\n
16.	IMAPv4-Advanced	Create Request	\r\n
17.	IMAPv4-Advanced	Create Response	OK CREATE completed\r\n
18.	IMAPv4-Advanced	Rename Request	\r\n

#	Flow	Action	Delimiter
19.	IMAPv4-Advanced	Rename Response	OK RENAME completed\r\n
20.	IMAPv4-Advanced	Delete Request	\r\n
21.	IMAPv4-Advanced	Delete Response	OK DELETE completed\r\n
22.	IMAPv4-Advanced	Fetch	\r\n
23.	IMAPv4-Advanced	Fetch Server	FETCH completed\r\n
24.	IMAPv4-Advanced	IDLE	IDLE\r\n
25.	IMAPv4-Advanced	IDLE Response	\r\n
26.	IMAPv4-Advanced	Done	DONE\r\n
27.	IMAPv4-Advanced	Done Response	\r\n
28.	IMAPv4-Advanced	Logout	\r\n
29.	IMAPv4-Advanced	Bye	OK LOGOUT completed\r\n

#	Flow	Action	Delimiter
30.	IMAPv4-Advanced	Login	1. Server: \r\n 2. Client: \r\n 3. Server: OK CAPABILITY completed\r\n 4. Client: \r\n 5. Server: \r\n 6. Client: \r\n 7. Server: \r\n 8. Client: \r\n 9. Server: authenticated\r\n 10. Client: \r\n 11. Server: OK NAMESPACE completed\r\n 12. Client: \r\n 13. Server: OK LSUB completed\r\n 14. Client: \r\n 15. Server: OK LSUB completed\r\n 16. Client: \r\n 17. Server: OK LSUB completed\r\n 18. Client: \r\n 19. Server: OK LSUB completed\r\n 20. Client: \r\n 21. Server: OK LSUB completed\r\n 22. Client: \r\n 23. Server: OK LSUB completed\r\n 24. Client: \r\n 25. Server: OK LSUB completed\r\n 26. Client: \r\n 27. Server: OK LSUB completed\r\n 28. Client: \r\n 29. Server: OK LIST completed\r\n 30. Client: \r\n 31. Server: OK [READ-WRITE] SELECT completed\r\n 32. Client: \r\n 33. Server: \r\n

#	Flow	Action	Delimiter
31.	IMAPv4-Advanced	Retreive Mail	1. Client: \r\n 2. Server: OK IDLE completed\r\n 3. Client: \r\n 4. Server: OK NOOP completed\r\n 5. Client: \r\n 6. Server: FETCH completed\r\n 7. Client: \r\n 8. Server: FETCH completed\r\n 9. Client: \r\n 10. Server: OK UID FETCH completed\r\n 11. Client: \r\n 12. Server: \r\n 13. Client: \r\n 14. Server: OK IDLE completed\r\n 15. Client: \r\n 16. Server: FETCH completed\r\n 17. Client: \r\n 18. Server: STORE completed\r\n 19. Client: \r\n 20. Server: \r\n 21. Client: \r\n 22. Server: OK IDLE completed\r\n
32.	IMAPv4-Advanced	Quit	1. Client: \r\n 2. Server: OK CLOSE completed\r\n 3. Client: \r\n 4. Server: OK LOGOUT completed\r\n

POP3

The following table describes the canned Super Flows that have support for the Proxy feature:

#	Super flow
1	Bandwidth POP3 RETR Only
2	Bandwidth POP3
3	POP3 512

#	Super flow
4	POP3-Advanced
5	POP3-Chinese
6	POP3-French
7	POP3-German
8	POP3-Italian
9	POP3-Japanese
10	POP3-Persian
11	POP3-Spanish
12	Financial POP3-Advanced
13	POP3 LTE Mix
14	POP3 RMIX 16K
15	POP3 RMIX 4K
16	POP3 RMIX 90K

The following table describes the flow action support for the Proxy feature:

#	Flow	Action	Delimiter
1	POP3-Advanced	STAT(Commad)	1. Client: STAT\r\n 2. Server: \r\n
2.	POP3-Advanced	LIST (Command)	1. Client: \r\n 2. Server: \r\n.\r\n
3.	POP3-Advanced	RETR(Command)	1. Client: \r\n 2. Server: \r\n.\r\n
4.	POP3-Advanced	DELE(Command)	1. Client: \r\n 2. Server: +OK\r\n
5.	POP3-Advanced	RETR(Command Loop)	1. Client: \r\n 2. Server: \r\n.\r\n
6.	POP3-Advanced	Banner	\r\n

7.	POP3-Advanced	AUTH(Command)	AUTH\r\n
8.	POP3-Advanced	QUIT(Command)	QUIT\r\n
9.	POP3-Advanced	+OK(Response)	+OK\r\n
10.	POP3-Advanced	-ERR(Response)	\r\n
11.	POP3-Advanced	CAPA(Command)	CAPA\r\n
12.	POP3-Advanced	CAPA(Command)	\r\n.\r\n
13.	POP3-Advanced	USER(Command)	\r\n
14.	POP3-Advanced	USER(Response)	+OK Password required.\r\n
15.	POP3-Advanced	PASS(Command)	\r\n
16.	POP3-Advanced	PASS(Response)	+OK logged in.\r\n
17.	POP3-Advanced	PASS Err (Response)	\r\n
18.	POP3-Advanced	Login	1. Server: \r\n 2. Client: AUTH\r\n 3. Server:\r\n 4. Client: \r\n\r\n 5. Server: +OK\r\n 6. Client: \r\n\r\n 7. Server: +OK\r\n
19.	POP3-Advanced	Retrieve Mail	1. Client: STAT\r\n 2. Server: \r\n 3. Client: LIST\r\n 4. Server: \r\n.\r\n 5. Client: UIDL\r\n 6. Server: \r\n.\r\n 7. Client: RETR 1\r\n 8. Server: r\n.\r\n
20.	POP3-Advanced	QUIT	1. Client: QUIT\r\n 2. +OK\r\n

SMTP

The following table describes the canned Super Flows that have support for the Proxy feature:

#	Super flow
1	SMTP 100k
2	SMTP 17k
3	SMTP Authentication Failure
4	SMTP Email
5	SMTP Email (Lawful Intercept)
6	SMTP Email-Chinese
7	SMTP Email-French
8	SMTP Email-German
9	SMTP Email-Italian
10	SMTP Email-Japanese
11	SMTP Email-Persian
12	SMTP Email-Spanish
13	Financial SMTP Email
14	SMTP Cisco EMIX
15	SMTP Enterprise
16	SMTP RMIX 16K
17	SMTP RMIX 4K
18	SMTP RMIX 90K

The following table describes the flow action support for the Proxy feature:

#	Flow	Action	Delimiter
1.	SMTP	Send HELO	1. Server: \r\n 2. Client: \r\n 3. Server: \r\n

2.	SMTP	Send EHLO	\r\n
3.	SMTP	Server 250 HELLO	customized For Example: ["250-#{servername}", "250-SIZE 0", "250-PIPELINING", "250 8BITMIME"], delimiter = 8BITMIME\r\n
4.	SMTP	AUTH Login	1. Client: \r\n 2. Server: \r\n 3. Client: \r\n 4. Server: \r\n 5. Client: \r\n
5.	SMTP	AUTH OK	Authentication successful\r\n
6.	SMTP	AUTH Failed	Authentication failed\r\n
7.	SMTP	Send VRFY	1. Client: \r\n 2. Server: \r\n
8.	SMTP	Send email	1. Client: \r\n 2. Server: Sender OK\r\n 3. Client: \r\n 4. Server: \r\n 5. Client: DATA\r\n 6. Server: <CRLF>.<CRLF>\r\n 7. Client: r\n.\r\n 8. Server: 250 2.6.0 <FESFSEF> Queued mail for delivery\r\n
9.	SMTP	Send FROM	\r\n
10.	SMTP	Send RCPT	\r\n
11.	SMTP	Send DATA	DATA\r\n

12.	SMTP	Send Email (Raw)	1. Client: \r\n 2. Server: Sender OK\r\n 3. Client: \r\n 4. Server: \r\n 5. Client: DATA\r\n 6. Server: <CRLF>.<CRLF>\r\n 7. Client: r\n.\r\n 8. Server: 250 2.6.0 <FESFSEF> Queued mail for delivery\r\n
13.	SMTP	250 Queued	delivery\r\n
14.	SMTP	Send QUIT	QUIT\r\n
15.	SMTP	221 Closing	Service closing transmission channel\r\n
16.	SMTP	Send RSET	RSET\r\n
17.	SMTP	250 OK	250 OK\r\n
18.	SMTP	Sender OK	Sender OK\r\n
19.	SMTP	Message DATA	\r\n.\r\n
20.	SMTP	354 Start	<CRLF>.<CRLF>\r\n
21.	SMTP	Server Connected	\r\n

Explicit Proxy/Load Balancer Configuration

Test Paths are particularly useful for setting up a test environment for proxies, or more specifically, for testing load balancers. However, keep in mind that Test Paths are not used solely for testing proxy support; this is only one instance in which you may want to use Test Paths.

In order to configure a load balancer to work with BreakingPoint, you will need to simulate clients connecting to the virtual server that is represented by the proxy, and you will need to be able to simulate the servers that are in the private pool.

To do this, you will need to set up your test configuration so that BreakingPoint's clients are talking to the load balancer. For example, you may want to connect the public interface to the BreakingPoint system's Interface 1 and the private interface to the BreakingPoint system's Interface 2. In this case, you will want the BreakingPoint system's clients to only communicate with the load balancer, since the private servers are hidden behind the load balancer.

For this particular example, you will want to set up a network where your clients are specified on Interface 1, your load balancer's public IP address is set up on the External interface, and your private servers are specified on Interface 2. After you have done this, you will need to configure the Test Paths so that there is only one connection from the Client to the External interface.

Once the network has been properly configured, you will need to create your test as normal. On the Interfaces tab of your test, you will need to select Interface 1 as the client, Interface 2 as the server, and External as the server, and you will need to select the domain you configured for proxy support for each interface.

This enables the component to make connections from Interface 1 to the External interface, and allow it to still listen for new connections on interface 2, enabling it to act as a one-arm server.

To set up proxy support:

1. Create a new Network Neighborhood or modify an existing Network Neighborhood so that it emulates the Network Neighborhood described below.
2. To create a new Network Neighborhood, select **Control Center > New Neighborhood** from the menu.
3. Click **NAT/Proxy** and then click **Select**.
4. Enter a name in the New Network Neighborhood Name field and then Click **OK**. Interfaces and elements are displayed.
5. Delete the IPv4 Router element.
6. Click the IPv4 Static Hosts element in order to view and modify its configuration options (note the Tags field).

   Note: This example uses logical interfaces 1 and 2, which could be mapped to any of the ports that have been reserved.

   a. For Lab Client, i1_default, click the Container field dropdown selector and select Interface 1.
   b. Configure the Base IP Address, Count and Netmask.
   c. For Lab Server,i2_default, click the Container field dropdown selector and select Interface 2.
   d. Configure the Base IP Address, Count and Netmask.
7. Click the IPv4 External Hosts element in order to view and modify its configuration options.
8. Configure the Base IP Address and Count.
9. In preparation for the next step, create a new Super Flow or be prepared to use a previously created Super Flow that you want to use for the test.

   Note: To create a new Super Flow, select **Managers** > **Super Flow**s from the menu.
10. From the menu select, **Managers** > **Application Profiles**.
11. Click **Create New**.
12. Name the Application Profile.
13. Click **Add Super Flow**.
14. Click the plus symbol at the right side of the Super Flow's name to add it the Associated Super Flows list. Type in the name of your Super Flow in the S**earch Criteria** field. Click the **Search** button.
15. Click **OK**. Click the **Save** button.
16. From the menu select **Test** > **New Test**.
17. Click the **Add New** button.
18. Select the Application Simulator. Click the **Select** button. Type "Server" in the field to name the Application Simulator.

19. Click the **Create** button.
20. Click the "writing pen" icon to edit the Server component that was just added.
21. Click the **Load Profile** button at the top right of the screen.
22. Configure the Load Profile as shown in the table below.

Ramp Up	Steady State	Ramp Down
Ramp Up Behavior= Full Open	Steady State Behavior=Hold Sessions Open	Ramp Down Behavior=Full Close
Ramp Up Duration= 00:00:00	Steady State Duration= 00:01:00	Ramp Down Behavior=00:00:15

23. Click **Return to Component Settings**.
24. In the Data Rate section, select the **Date Rate Unlimited** check box.
25. In the Session/Super Flow Configuration section, select the **Unlimited Super Flow Close Rate** check box.
26. In the App Configuration section, enter "1" in the Streams Per Super Flow field.
27. In the App Configuration section, configure the Application Profile setting by browsing, searching for and selecting the Application Profile that you created.
28. In the Component Tags section under Client Tags, delete i1_Default Client, click the "x" at the left side of the tag name.
29. Click **Return to Workspace**.
30. Click **Add New**.
31. Select **Client Simulation**. Click the **Select** button.
32. Click **Create**.
33. Click the "writing pen" icon to edit the ClientSim_1 component that was just added.
34. Click the **Load Profile button.**
35. Configure the ClientSim_1 Load Profile as described in the table below.

Ramp Up	Steady State	Ramp Down
Ramp Up Behavior= Full Open	Steady State Behavior=Open and Close Sessions	Ramp Down Behavior=Full Close
Ramp Up Duration= 00:00:00	Steady State Duration= 00:00:30	Ramp Down Behavior=00:00:15

36. Click **Return to Component Settings**.
37. Configure the Data Rate and Session/Super Flow Configuration sections as shown in the table below.

Data Rate	Session/Super Flow Configuration
Data Rate Unlimited (unchecked)	Maximum Simultaneous Super Flows=1
Data Rate Scope=Limit Per-Interface Throughput	Maximum Simultaneous Active Flows=0 See Maximum Simultaneous Active Flows Details on page 811.
Data Rate Unit=Megabits / Second	Maximum Super Flows Per Second=.0001
Data Rate Type=Constant	Unlimited Super Flow Open Rate (unchecked)
Minimum Data Rate=100	
Minimum Data Rate=1	

38. Configure the App Configuration section as shown in the table below.

Data Rate	Value
Streams Per Super Flow	1
Content Fidelity	Normal
Delay Start	00:00:10
Super Flow	(Browse and select the Super Flow that will be used for the test.)

39. Click **Return to Workspace**.
40. Click **Save As**, enter a name for the Proxy Test, and then click **Save**.

Packet Filtering

The Packet Filter allows the most efficient usage of the interface card's capture history. Packet Filters are set on a per-port basis, and will process packets as they are received and only capture the packets that you have chosen to capture.

This type of filtering can be used to increase the effective depth of the capture memory by only capturing the packets of interest.

Editing Packet Filters allows you to select which packets you wish to capture.

Note: The Packet Filter feature only captures traffic that is to be received. No transmitted traffic will be captured while the Packet Filter feature is in use.

To edit Packet Filters:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select a Network Neighborhood from the **Network Neighborhood** list.
3. Click the Packet Filter link that corresponds to the interface to which you want to apply filters.
4. Select the parameters you want to include.
5. Click Accept.

The Packet Filter Parameters table provides descriptions of each Packet Filter parameter.

Packet Filter Parameters

Parameter	Description
Vlan	Keep any packets matching the given Vlan ID
Src Port	Keep any packets matching the given source port
Dest Port	Keep any packets matching the given destination port
Src IP	Keep anything matching the given source IP address
Dest IP	Keep anything matching the given destination IP address
Combine Filters	Logically combines each filter value. For example, if the Combine Filters section is configured for And, and the Source IP and Destination IP are configured, the resulting filter will match on packets only when a packet contains both the configured source IP and the configured destination IP.

Note: Leave the Not check box in the Combine Filters section unchecked to include the corresponding parameter. Select the Not check box to exclude the corresponding parameter.

Impairments

Use the Impairments feature to introduce impairments such as dropped packets, corrupt IP checksum, and corrupted packets in various ranges to your tests. You can also specify what percentage of packets is to be impaired.

Note: The component sections related to frames transmitted (Tx) represent the statistics before impairments are introduced. The aggregate statistics (aggstats) section represents the statistics after impairments have been introduced.

Any combination of the following list of impairments can be selectively included on a per-port basis:

- Drop packet
- Frack packet *
- Corrupt packet in bytes 1-64
- Corrupt packet in bytes 65-256
- Corrupt packet in bytes 257-end

- Randomly corrupt packet
- Corrupt IP checksum

Note:
- In this context, the term Frack refers to the process of separating a packet into 8-byte portions and removing random portions of that packet.
- The term Corrupted Packets refers to packets corrupted at Layer 3 or above. When using the Corrupt packets in bytes 257-end impairment , it is important to note that packets corrupted at Layer 2 will have a bad frame check sequence (FCS) and will be dropped at the physical layer and counted as dropped packets instead of as Corrupted Packets.

To add impairments:

1. Select **Control Center > Open Neighborhood** from the Menu bar.
2. Select a Network Neighborhood from the **Network Neighborhood** list.
3. Click the Impairments link that corresponds to the interface to which you want to introduce impairments.
4. Select the Impairments parameters you want to include.
5. Enter the percentage of packets you want to be impaired in the Rate field. For example, to drop 1 packet out of every 10 packets, enter **10** in the Rate field. To drop 1 packet out of every 1000 packets, enter `0.1` in the Rate field.
6. Click Accept.

Impairments Parameters table provides descriptions of each Impairments parameter.

Impairments Parameters

Parameter	Description
Drop Packet	Drops packets at the rate specified
Frack Packet	Separates the packet into 8-byte portions and randomly removes portions from the packet
Corrupt Packet in Bytes 1-64	Corrupts packets only within the first 64 bytes of the packet
Corrupt Packet in Bytes 65-256	Corrupts packets only between the 65th and the 256th byte of the packet
Corrupt Packet in Bytes 257-end	Corrupts packets only between the 65th and the 256th byte of the packet
Randomly Corrupt Packet	Corrupts packets in a random location within the packet
Corrupt IP Checksum	Creates an invalid checksum

Note:
- You may receive unexpected results from the MAC when you run tests with Impairments turned on. MAC errors will often be higher than the number of impairments.
- The term Corrupted Packets refers to packets corrupted at Layer 3 or above. When using the Corrupt packets in bytes 257-end impairment , it is important to note that packets corrupted at Layer 2 will have a bad frame check sequence (FCS) and will be dropped at the physical layer and counted as dropped packets instead of as Corrupted Packets.

SCTP Tunneling Over UDP

BreakingPoint supports tunneling of SCTP over UDP. This allows SCTP to function in any network that supports UDP.

One disadvantage of tunneling SCTP over UDP is that the source and destination ports must be the same. This requirement limits the number of possible flow tuples that can be simultaneously used in a test. This may cause a test to reach a maximum number of concurrent sessions that is less than the amount configured for the test. The number of tuples can be increased by modifying the Network Neighborhood to have a larger range of IP addresses for each client and server interface used in each test.

Another disadvantage of tunneling SCTP over UDP is that more protocol header overhead is required, leaving less room available for application payload.

Note: When using SCTP over UDP, both SCTP and UDP flow counts will be displayed on the Real-Time Statistics page. This information will also be contained with the resulting report.

SCTP Shared Connection

When the SCTP shared connection is enabled within a Super Flow, only one SCTP connection will be opened and all sessions for that Super Flow will share the same SCTP connection. However, you can configure the component to open more than one shared SCTP connection by changing the Streams Per Super Flow component setting. For example, if you configure the number of Streams Per Super Flow to 2, the component will create two independent SCTP sessions.

User Simulation for Next Generation Firewalls

Next generation firewalls (NGFWs) have the ability to set policies based on the identification of the user on the enterprise. BreakingPoint allows you to test the user based policies of these next generation firewalls.

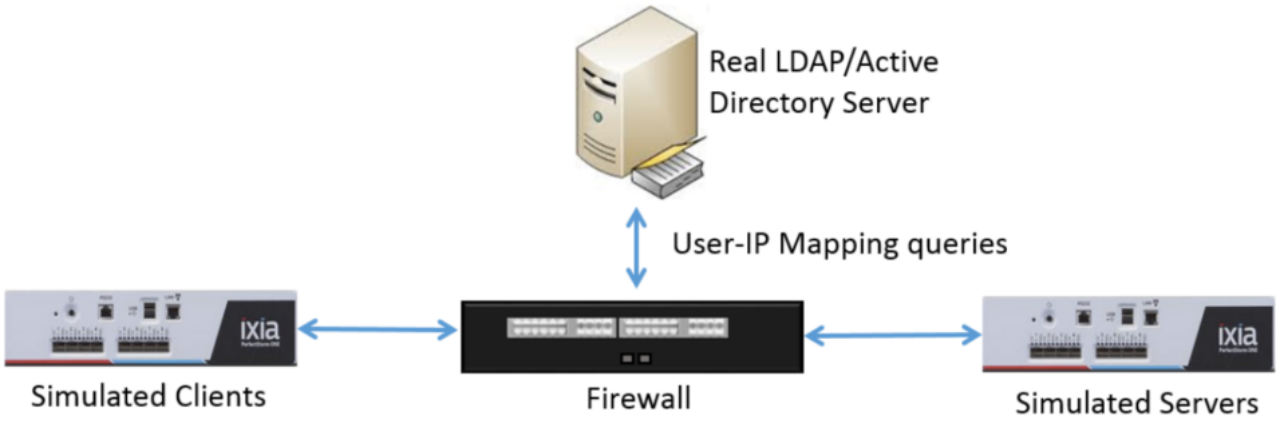
BPS User Simulation Notes:

- The Application Simulator, Client Simulator and Session Sender components support this feature.
- The users must be present in the Active Directory Server before running a test.
- The IP addresses used in a test will be selected in a round robin fashion during LDAP authentication.
- Clear text authentication will be used.

The User Authentication Sequence is as follows:

1. The BP Simulated Client attempts to log in/authenticate.
2. The firewall queries the LDAP/Active Directory Server for user information. The authentication process will establish the IP address/username correlation in LDAP.
3. If authentication is successful, the BPS simulated client sends traffic to the BP simulated Server.
4. The firewall applies access rules to the authenticated users. The user can be restricted to specific Apps, ports, etc., based on the access rules.

Example Test Diagram



Configure Test Parameters

1. From your Network Neighborhood, select **Add New Element > Endpoint > IPv4LDAP Server** and configure the following parameters for the LDAP server.

LDAP Server Parameters

Parameter	Description
ID	LDAP server settings ID for internal usage.
Username Start Range	The username will be generated with the defined text as an alphanumeric combination. For example, if the username is usr, the user names will be generated as usr1, usr2, etc.
Password Tag	Passwords will be generated in the same manner as described for username above. For example, pass1, pass2, etc.
DN Fixed Part	LDAP servers require a login with a complete distinguished name such as cn=usr1,ou=ixia-group ,cn=admin,dc=test,dc=com. Enter the fixed part of the distinguished name.
User Sequence Min	Enter the User and Password start value. See LDAP Server User Sequence Logic for a detailed description.

Parameter	Description
User Sequence Max	Enter the User and Password end value. See LDAP Server User Sequence Logic for a detailed description.
User Count	Displays the number of users that will be simulated for the test based on the User Sequence Min and User Sequence Max settings.
LDAP Server IP Address	Enter the LDAP server IP address.
Authentication Rate	Enter the user authentication rate of the BPS client. Note that flooded authentication will generate errors from server side.
Authentication Timeout	Define the retry timeout for authentication.

2. Edit the IPv4 Static Hosts. For each IPv4 Static Host that you want to use in your simulation, configure the LDAP Server setting to match the ID value defined in step 1. This step will create the IPv4 Static hosts that will be used for the simulation.

LDAP Server User Sequence Logic

The section provides examples of how the **User Sequence Min** and **User Sequence Max** parameters are used to create the users and passwords for Next Generation Firewall Simulation.

Note:

- Authentication requires the combination of an IP address and user. In the examples below, note that the IP addresses used in a test will be associated with users in a round robin fashion during LDAP authentication. During a test, a single user can be associated with multiple IP addresses, but multiple users cannot be associated with a single IP address.
- BPS clients do not use their assigned IP addresses (described below) in sequential order when sending traffic.

Example 1 Configuration

IPv4 LDAP Config					IP Static Host	
Username Start Range	Password Tag	User Sequence Min	User Sequence Max	User Count	Base IP Address	IP Count
ixia	pass	5	7	4	10.10.0.1	4

Example 1 Results

User-ID	Username	Password	IP Address
1	ixia5	pass5	10.10.0.1

User-ID	Username	Password	IP Address
2	ixia6	pass6	10.10.0.2
3	ixia7	pass7	10.10.0.3
1	ixia5	pass5	10.10.0.4

Example 2 Configuration

IPv4 LDAP Config					IP Static Host	
Username Start Range	Password Tag	User Sequence Min	User Sequence Max	User Count	Base IP Address	IP Count
ixia	pass	5	7	2	10.10.0.1	4

Example 2 Results

User-ID	Username	Password	IP Address
1	ixia5	pass5	10.10.0.1
2	ixia6	pass6	10.10.0.2
1	ixia5	pass5	10.10.0.3
2	ixia6	pass6	10.10.0.4

Example 3 Configuration

IPv4 LDAP Config					IP Static Host	
Username Start Range	Password Tag	User Sequence Min	User Sequence Max	User Count	Base IP Address	IP Count
ixia	pass	0	0	2	10.10.0.1	4

Example 3 Results

User-ID	Username	Password	IP Address
1	ixia0	pass0	10.10.0.1
1	ixia0	pass0	10.10.0.2

User-ID	Username	Password	IP Address
1	ixia0	pass0	10.10.0.3
1	ixia0	pass0	10.10.0.4

Example 4 Configuration

IPv4 LDAP Config					IP Static Host	
Username Start Range	Password Tag	User Sequence Min	User Sequence Max	User Count	Base IP Address	IP Count
ixia	pass	5	7	3	10.10.0.1	2

Example 4 Results

User-ID	Username	Password	IP Address
1	ixia5	pass5	10.10.0.1
2	ixia6	pass6	10.10.0.2

This page intentionally left blank.

CHAPTER 9 Port Reservations

This section covers:

Device Status

The Device Status screen provides a graphical representation of the BreakingPoint system. It displays the slots (or blades) on the chassis and their ports. This is an interactive screen that enables you to reserve ports, assign Active Groups, export packet buffers, and remap ports. The figures in this section show the Device Status screen and provide callouts for each feature on this screen.

The Device Status screen is accessible from any area in the Control Center using the BreakingPoint icon located in the upper-right corner of the Control Center. Clicking on this icon will open a pop-up window displaying the Device Status screen.

Depending on which Ixia BreakingPoint product you have purchased, the Device Status screen in one of the following figures will be displayed when you click on the BreakingPoint icon. Each table below the figures provides callouts for each feature on the Device Status screen.

BreakingPoint Storm Device Status Screen

The following table lists the elements on the BreakingPoint Storm Device Status screen.

BreakingPoint Storm Device Status Screen

Callout	Element	Description
1	Locked Port Reservation Icon	The icon with a key indicates that the port is reserved by you. The number on the icon indicates the Active Group to which the port belongs.
2	Port Reservation Icon	The padlock icon indicates that another user has the port reserved.
3	Active Group	Use this menu to assign an Active Group to a slot.
4	Port Mapping Options	Use the port mapping panel to remap locked ports to different interfaces. The panel will only show the selected Active Group's ports.
5	Port Configuration	Use the Port Configuration panel to set the port speed for each port in your test.
6	Packet Export	Use the Packet Buffer Export feature to export PCAPs from the latest test run. For more information on exporting packet buffers, see the Exporting a Packet Buffer on page 650 section.

The following figure illustrates the Device Status screen of the BreakingPoint FireStorm.

BreakingPoint FireStorm Device Status Screen

The following table lists the elements on the BreakingPoint FireStorm Device Status screen.

BreakingPoint FireStorm Device Status Screen Elements

Callout	Element	Description
1	Active Group	Use this menu to assign an Active Group to a slot.
2	Port Mapping Options	Use the port mapping panel to remap locked ports to different interfaces. The panel will only show the selected Active Group's ports.
3	Port Configuration	Use the Port Configuration panel to set the port speed and configure the MTU for each port in your test.
4	Packet Export	Use the Packet Buffer Export feature to export PCAPs from the latest test run. For more information on exporting packet buffers, Export Packet Buffer on page 212.
5	Port Reservation Icon	The padlock icon indicates that another user has the port reserved.
6	Locked Port Reservation Icon	The icon with a key indicates that the port is reserved by you. The number on the icon indicates the Active Group to which the port belongs.

The following figure illustrates the Device Status screen of the Ixia BreakingPoint PerfectStorm.

Ixia BreakingPoint PerfectStorm Device Status Screen

The following table lists the elements on the Ixia BreakingPoint PerfectStorm Device Status screen.

Ixia BreakingPoint PerfectStorm Device Status Screen Elements

Callout	Element	Description
1	Active Group	Use this menu to assign an Active Group to a slot.
2	Port Mapping Options	Use the port mapping panel to remap locked ports to different interfaces. The panel will only show the selected Active Group's ports.
3	Port Configuration	Use the Port Configuration panel to set the port speed and configure the MTU for each port in your test.
4	Packet Export	Use the Packet Buffer Export feature to export PCAPs from the latest test run. For more information on exporting packet buffers, Export Packet Buffer on page 212
5	Port Reservation Icon	The padlock icon indicates that another user has the port reserved.
6	Locked Port Reservation Icon	The icon with a key indicates that the port is reserved by you. The number on the icon indicates the Active Group to which the port belongs.

Port Reservations

In order to run tests on the BreakingPoint system, you must make port reservations. To reserve a single port, click on the port you want to reserve. To reserve all of the ports on a blade, right-click on one of the ports and select Reserve all ports on this slot.

When you lock a port reservation, the system will automatically map the port to the next available test interface. Each test interface references a set of domains in a Network Neighborhood.

For more information on Network Neighborhoods, see the What is a Network Neighborhood? on page 103 section. For more information on port mapping, see the Port Mapping on page 12 section.

The number of tests that you can run concurrently depends on the number of available ports that the BreakingPoint Storm has. For example, a single-blade BreakingPoint Storm with four available ports can only run four tests at a time. A two-blade chassis with sixteen total available ports can run sixteen tests simultaneously. However, in order to run all sixteen tests concurrently, you will need to assign each available port to a different Active Group.

The following video link provides a BreakingPoint tutorial on how to Reserve Ports: https://www.youtube.com/watch?v=GIEJyPHIDgs

Active Groups

All reserved ports belong to an Active Group. The basic function of an Active Group is to enable you to run multiple tests concurrently. In order to run multiple tests concurrently, each test must be run under a different Active Group.

For example, if Slot 1/Ports 0-3 can be assigned to Group 1, and Slot 2/Ports 0-3 can be assigned to Group 2, then you can run two tests simultaneously. However, if all ports across both blades share the same Active Group, then only one test can run at a time.

The number of available Active Groups depends on the number of ports you have reserved at the time. For example, if you have no ports reserved, then the **Active Group** menu will only list Group 1. If you have one port reserved, then you will see Group 1 and Group 2. If you have two ports reserved under two different groups, then you will see Group 1, Group 2, and Group 3.

Note: The maximum number of groups available in the Active Group drop-down menu is determined by the number of ports on our chassis. BreakingPoint recommends that you limit the number of Active Groups that you create to the number of ports on your chassis. Creating more Active Groups than the number of ports on your chassis will require you to unreserve and re-reserve ports in the user interface.

To change an Active Group:

1. Select **Control Center > Device Status** from the Menu bar.
2. Click the **Active Group** drop-down menu.
3. Select an Active Group from the drop-down menu.
4. The system will always list one more group than number of groups you are currently using.
5. Click on a port on the slot you would like to reserve.

All ports on the slot will be tagged with an icon and a number denoting the port's assigned group.

Port Reservation Methods

- Reserving an unreserved port
- Force reserving a reserved port

Reserving an Unreserved Port

Unreserved ports may be reserved simply by right-clicking on one of the ports on a slot and selecting Reserve all ports on this slot. This will reserve all the ports on the slot under your account.

Note: For ports that you have reserved, an icon with a key and the number of the assigned group will be displayed on the ports. For ports that another user has reserved, a padlock will be displayed on the ports without a key or a group number.

An important thing to remember when reserving your ports is the order in which you reserve them. Whenever you reserve a port, the system will automatically map that port to an interface on the chassis.

For example, if you reserve ports 0 and 1, then port 0 will map to interface 1 and port 1 will map to interface 2. You can use these interfaces to run tests. If an interface is not mapped to a port, then you cannot use that interface to run tests.

If you want to remap the ports to different interfaces, you can click on the **Port Mapping** options, located on the **Device Options** screen, and manually remap the ports. For more information on Port Mapping, see the Port Mapping on the facing page section.

Note: Only reserved ports will be mapped to interfaces.

To reserve an unreserved port:

1. Select **Control Center** > **Device Status** from the Menu bar.
2. Click the **Active Group** drop-down menu.
3. Select the Active Group to which you would like to assign the ports.
4. Click on the port(s) you would like to reserve.

An icon containing a key and the port's Active Group number will be displayed over the port you have reserved.

Force Reserving a Port

A force reserve will remove another user's reservations from the ports and reserve the ports under your account. During a force reserve, the system will alert you that the ports are reserved by another user and ask if you want to force reserve the port(s).

Note: If another user unreserves a port that you are using, or if a port you are using becomes disrupted, the user interface may not alert you that you are no longer connected to that port. Therefore, it is possible for the user interface to display stale, or out-of-date test information while you are testing. For instance, the user interface may display a Valid status for a test, even though the test was not completed. The current status of your tests will be displayed whenever you make a change to your test (such as clicking Save, Update, or Apply).

You should check the port notes before you force reserve the port(s) because other system users may not want you to remove their port reservations. If available, the port notes will appear as a yellow note icon located below the port.

As a best practice recommendation, you should add a port note to your reserved ports. For example, you may want to note that port 0 and port 1 are connected to ports 5 and 6 on your switch. This lets you know the physical layout of the lab without having to enter it.

To force reserve ports:

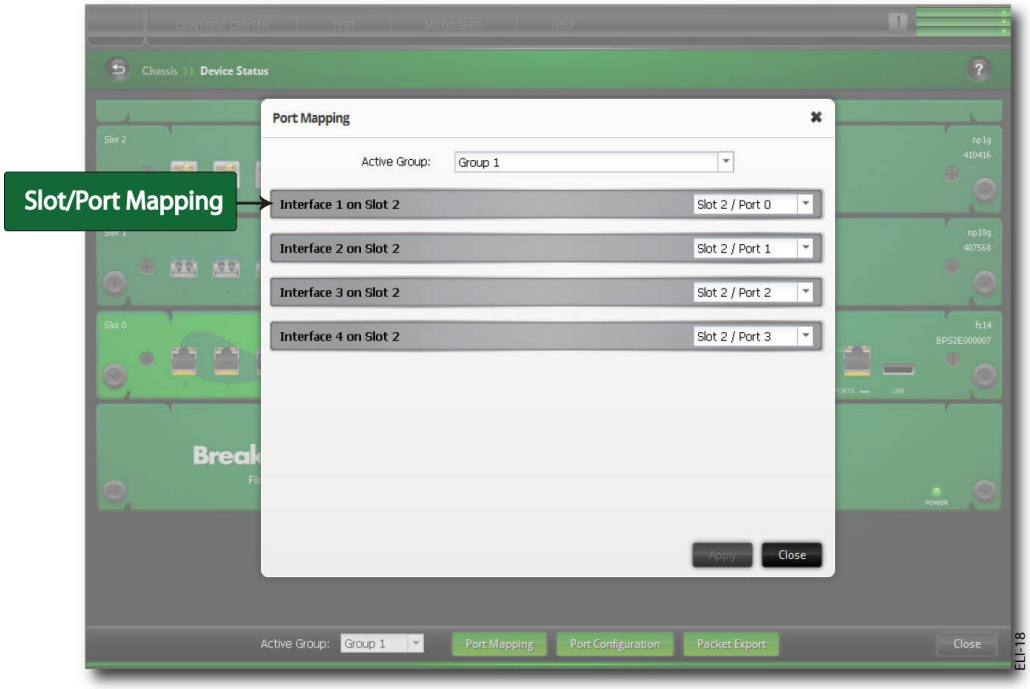
1. Select Control **Center > Device Status** from the Menu bar.
2. Click on the port(s) you would like to reserve.
3. You can only force reserve ports that do not have tests or system processes running on them. When you attempt to force reserve ports that have tests or system processes running on them, the system will alert you that there is a process running on that slot.
4. Click **Yes** when the dialog window displays, asking if you would like to force reserve all the ports in the slot.
5. The port(s) that you clicked on will display an icon, denoting that this port has been reserved by you.

You can also force reserve ports by right-clicking on one of the ports on a slot and selecting Reserve all ports on this slot. The system will alert you that the ports are reserved by another user and ask if you want to force reserve the port(s).

Port Mapping

The BreakingPoint system automatically maps ports to interfaces when you make your port reservations. Port mappings are important because they link a port on the BreakingPoint system to an interface in the Network Neighborhood. Each interface references a specific domain in the Network Neighborhood. See the following figure.

Port Mapping Panel



For example, if you click on the ports in the following order: Port 0, 1, 2, and 3 on Slot 1 and Port 0, 1, 2, and 3 on Slot 2, then the system will automatically map the ports in the following interfaces:

- Slot1/Port 0 to Interface 1
- Slot1/Port 1 to Interface 2
- Slot1/Port 2 to Interface 3
- Slot1/Port 3 to Interface 4
- Slot2/Port 0 to Interface 5
- Slot2/Port 1 to Interface 6
- Slot2/Port 2 to Interface 7
- Slot2/Port 3 to Interface 8

Unreserving a reserved port will automatically resequence the ports to the preceding interface.

For example, if you have all slots and ports mapped according to the previous example, and you unreserve Slot1/Port 0, then the system will automatically resequence the port mappings to the following interfaces:

- Slot1/Port 0 to Interface 1
- Slot1/Port 1 to Interface 2
- Slot1/Port 2 to Interface 3
- Slot2/Port 0 to Interface 4
- Slot2/Port 1 to Interface 5
- Slot2/Port 2 to Interface 6
- Slot2/Port 3 to Interface 7

You will notice that Slot1/Port1 have been removed from the port mappings; only 7 interfaces are in use; and the port mappings have resequenced to the preceding interface.

Manual Port Mappings

The BreakingPoint Storm automatically maps ports to interfaces when you make port reservations; however, there may be instances when you want to remap your ports to different interfaces. In these cases, you should use the Port Mapping feature.

If you need to remap ports to different interfaces, click on the **Port Mapping**button on the Device Status screen. This will open a pop-up window that lists all the ports that have port reservations for the current Active Group you have selected.

Note: In order to manually map ports, the Active Group whose ports you want to map must be selected.

The interfaces will be labeled using the following format: **Slot X:N**, where **Slot X** represents the slot number to which the port belongs, and **N** represents the port number.

To manually map a port:

1. Select **Control Center > Device Status** from the Menu bar.
2. Verify that the Active Group whose ports you would like to remap is selected.
3. Verify that you have ports reserved under this Active Group.
4. Click the **Port Mapping** button.
5. You will see all the ports that have locked reservations under the selected Active Group.
6. Click the drop-down button located under each interface.
7. A list of ports with reservations will display.
8. Select a the desired port from the list.
9. Repeat steps 4 - 8 for each port mapping.
10. Click the **Apply** button when you are done.

Port Notes

Ports Notes are used to add a note, or comment, to a specific port. When posted, the Port Note will appear as a small yellow note under the port. See the following figure.

Port Notes

Port Notes are visible to all users who are logged into the system, so all users can quickly assess a port's availability and/or physical layout. For example, Port Notes can provide descriptions of the test lab's layout (e.g., BreakingPoint Storm Slot 1/Port 1 is connected to Port 8 on the Cisco switch), or the Port Note can alert other users that the port will be in use on a specific date and time (e.g., 24 hour test running on 12/3 ending on 12/4).

This is extremely useful in cases where you are running a test remotely and do not want to go to the lab to figure out the test setup or when you are running tests over a period of time and do not want other users to reserve the ports.

Viewing Port Notes

Port Notes can be viewed by any user logged into the system. To access the information within a Port Note, simply click on the yellow note to open up the note. When viewing the note, you have the option of adding information to it or removing the Port Note entirely.

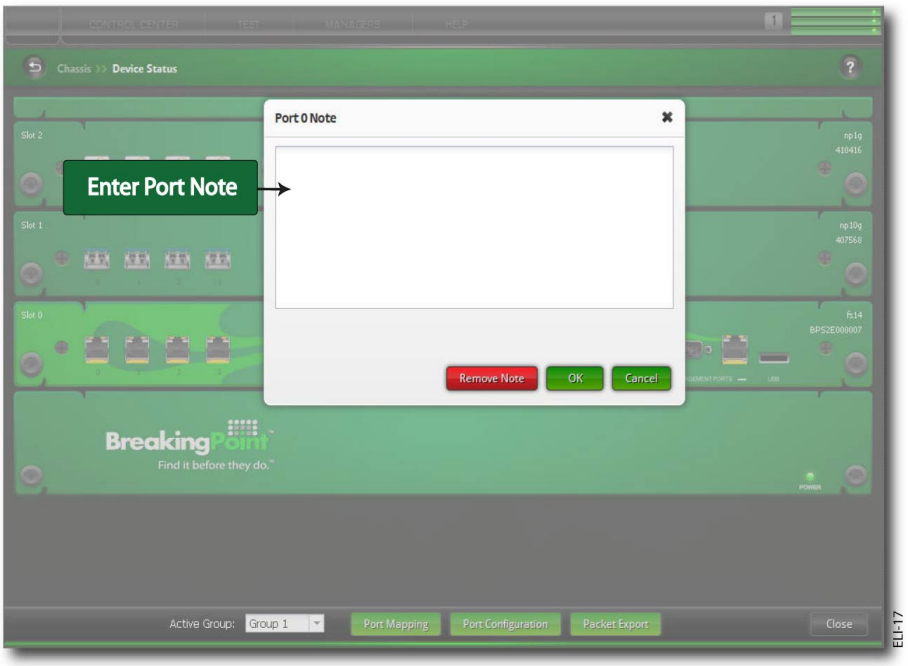
Note: The BreakingPoint Storm does not denote the user who has posted or modified the Port Note.

Adding Port Notes

Port Notes can be used to append information to a port. They are visible to all users who are logged into the system.

Users can add a Port Note at any time, regardless of whether they have the slot reserved or not. See the following figure.

Adding Port Notes



To add a Port Note:

1. Select **Control Center > Device Status** from the Menu bar.
2. Right-click on the port to which you would like to add a Port Note.
3. Select **Port Note** from the menu.
4. The Note Pad will appear for the Slot/Port you have selected.
5. Enter your note in the space provided.
6. There is a 700 character limit.
7. Click the **OK** button.

Modifying Port Notes

Any user can add information to or modify an existing Port Note. Once the change has been applied, the system will automatically update the Port Note so all users who are logged into the system will be able to see the most current Port Note.

To modify a Port Note:

1. Select **Control Center > Device Status** from the Menu bar.
2. Right-click on the Port Note you would like to edit.
3. Select **Port Note** from the menu.
4. Enter your note in the space provided.
5. Click the **OK** button.

Removing Port Notes

Port Notes can be removed by any user at any time.

To remove a Port Note:

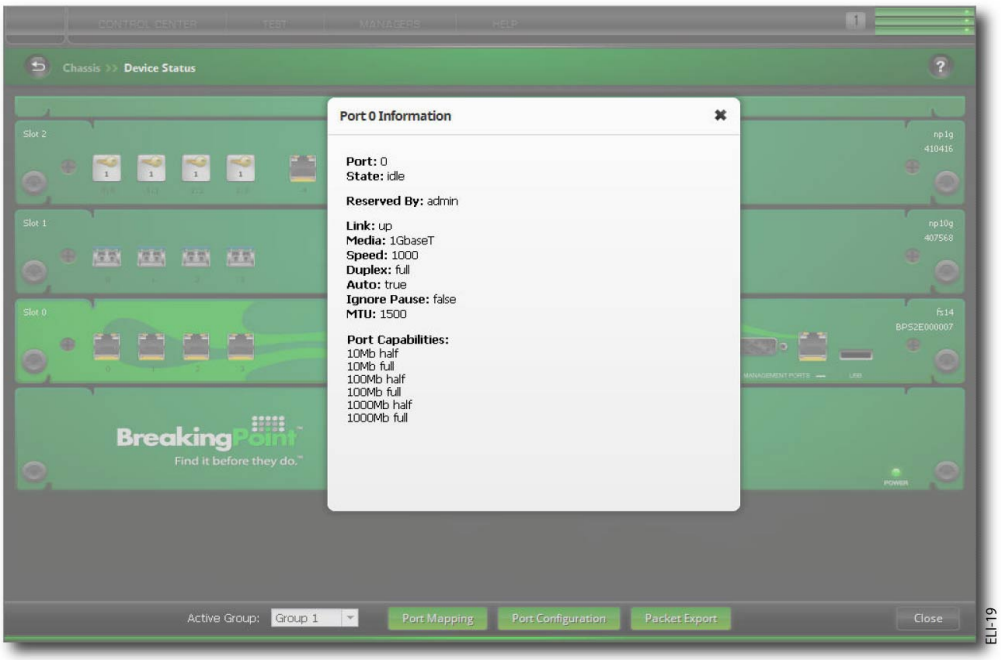
1. Select **Control Center > Device Status** from the Menu bar.
2. Click on the Port Note you would like to remove.
3. The Port Note will open.
4. Click the **Remove** button.

Port Information

If you right-click on any port on the Device Status screen and select **Port Information**, the system will provide you with the following information about the port:

- The slot and port number
- The port state (OK = port is functioning normally)
- The port's current reservation status (true = reserved; false = unreserved)
- The user account under which the port is reserved
- The link status (up or down)
- The connection media (fiber or copper)
- The port speed
- The auto-negotiation setting (true = auto-negotiation is on; false = auto-negotiation is off)
- Ignore Pause (true/false)
- The maximum transmission unit (MTU)
- Port Capabilities
- 10000 Mb full (10 Gbase-SR Short reach fiber XFP)
- 10 Mb half (1 Gbase-T Standard copper SFP)
- 10 Mb full (1 Gbase-T Standard copper SFP)
- 100 Mb half (1 Gbase-T Standard copper SFP)
- 100 Mb full (1 Gbase-T Standard copper SFP)
- 1000 Mb half (1 Gbase-T Standard copper SFP)
- 1000 Mb full (1 Gbase-T Standard copper SFP)
- The following figure displays this information.

Port Information



Resource Allocation

BreakingPoint devices do not employ an aggregation mode, whereby users would be required to manually allocate CPU resources between components. Instead, BreakingPoint devices utilize a load balancing algorithm to automatically distribute CPU resources to each component being used.

The resources of each blade are allocated in proportion to the number of ports reserved on the blade. In some cases, it may be necessary to reserve additional ports in order to secure enough resources for the test being performed. The sessions and bandwidth available to a test may be insufficient to adequately perform the test if too few ports have been reserved.

Note: A component can use the resources of only one CPU. BreakingPoint recommends that you define at least one component per CPU in order to fully utilize the available resources.

For example, if you reserve one port on a blade that has four total ports, you will have access to 25% of that blade's total resources. If you reserve three ports on that same blade, you will then have access to 75% of that blade's total resources.

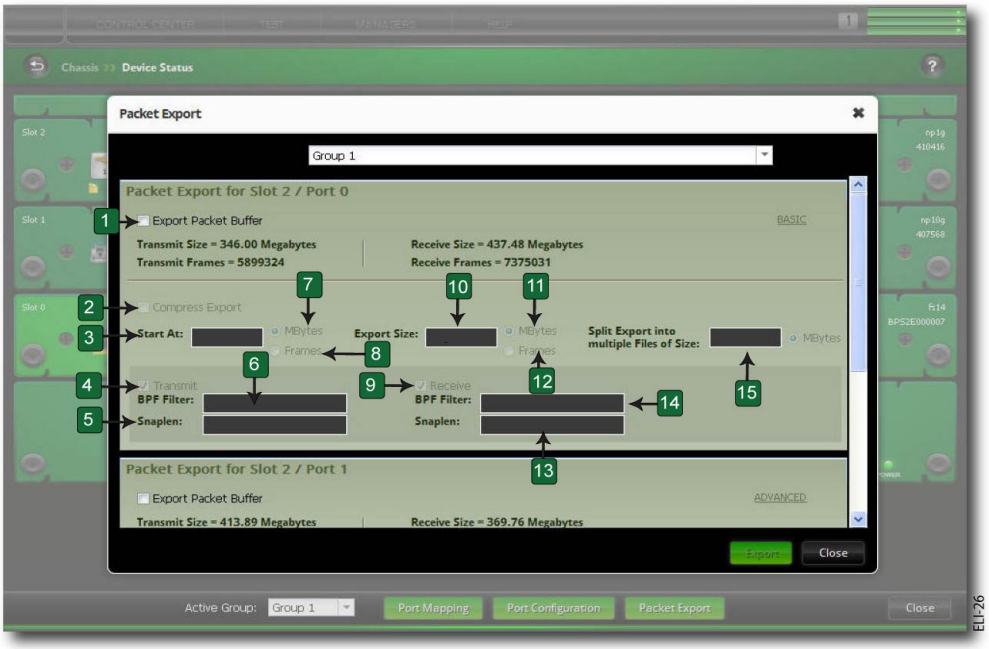
To see the resources available to you, BreakingPoint provides you with a resource allocation chart at the startup screen. This chart will provide you with information on which resources are in use and which resources are available for you to access. This chart will be displayed when you first click on a port if you have not logged in for the past 30 days. You can also view the resource allocation percentage of each port by placing your mouse over the port.

Note: Resource allocation can also be impacted when multiple tests are running concurrently.

Packet Export

From the Device Status screen, there is a **Packet Export** button that enables you to select the packet buffers you would like to export. Each port has its own packet buffer, so you will need to select the slot (s) and port(s) from which you would like to export content.

Packet Export Screen.



Export Packet Buffer

Callout	Parameter	Description
1	Export Packet Buffer	Select packet buffer to export
2	Transmit Size	Transmit size of the packet capture in megabytes
3	Transmit Frames	Number of frames transmitted
4	Receive Frames	Number of frames received
5	Receive Size	Size of the packet capture received in megabytes

For more information on the Packet Export button, see the section on Exporting a Packet Buffer.

CHAPTER 10 Configuring CloudStorm Speed Options

CloudStorm load modules (cards) have a fan-out feature that provides flexibility in configuring your test ports.

The two physical ports of a CloudStorm load module can function in any of the following modes:

- Two 100GbE CXP ports
- Two 40G QSPF+ ports (40G mode)

Note: CloudStorm cards may be shipped with functionality that supports only 40G mode or only 100G mode. The card can be upgraded to support both speeds by using an activation code that is specific to each card. Activation codes can be purchased from Ixia Sales Support.

To activate the code:

1. After logging in to the BPS web interface, click the **Ixia Chassis** icon.
2. In the Chassis Manager that opens, click the gear icon at the top right area of the page.
3. Click **Advanced Settings**.
4. Enter the activation code provided by Ixia Sales Support.
5. Click **ACTIVATE**.

Note: For the activation code to be successfully applied, the IxServer service needs to be fully up and the specific card that needs activation needs to be plugged in and visible in the Chassis Manager Overview tab.

Configuring CloudStorm Speed

To set the load module in fan-out mode:

1. In the BPS web interface, access the Device Status window.
2. Click the gear icon on at the top of the CloudStorm card.
3. Select a fan-out slot option.
   a. 40Gbe: Fan-out to 2 x 40
   b. 100Gbe: Fan-out to 2 x 100
4. Select the **Fan Out** check box.
5. Click **Apply**.

The load module will reboot. After the reboot has completed, the port options displayed on the load module will reflect the new configuration.

CHAPTER 11 CloudStorm 40GE 2-port Fusion Load Module

This card functions identically to a CS100G Load Module in 40G speed down mode. It operates in 40G mode by default and will support 100G mode/speed with the proper license activation code - please contact Customer Support or Sales for details.

Note that the text displayed at the top of the slot in the image below is "Slot 1/40G". This text indicates the current slot/speed of the load module. The text will change to "Slot 1/100G" if you install a 100G license and switch the card to 100G mode.

This page intentionally left blank.

CHAPTER 12 Configuring PerfectStorm Speed Options

PerfectStorm 40Gbe and 100Gbe load modules (cards) have a fan-out feature that provides flexibility in configuring your test ports.

The single physical port of a PerfectStorm 100GbE load module can function in any of the following modes:

- One 100GbE CXP port
- Two 40G QSPF+ ports (40G mode)
- Eight 10G SFP+ ports (10G mode)

The single physical port of a PerfectStorm 40GbE load module can function in any of the following modes:

- One 40G QSPF+ ports (40G mode)
- Four 10G SFP+ ports (10G mode)

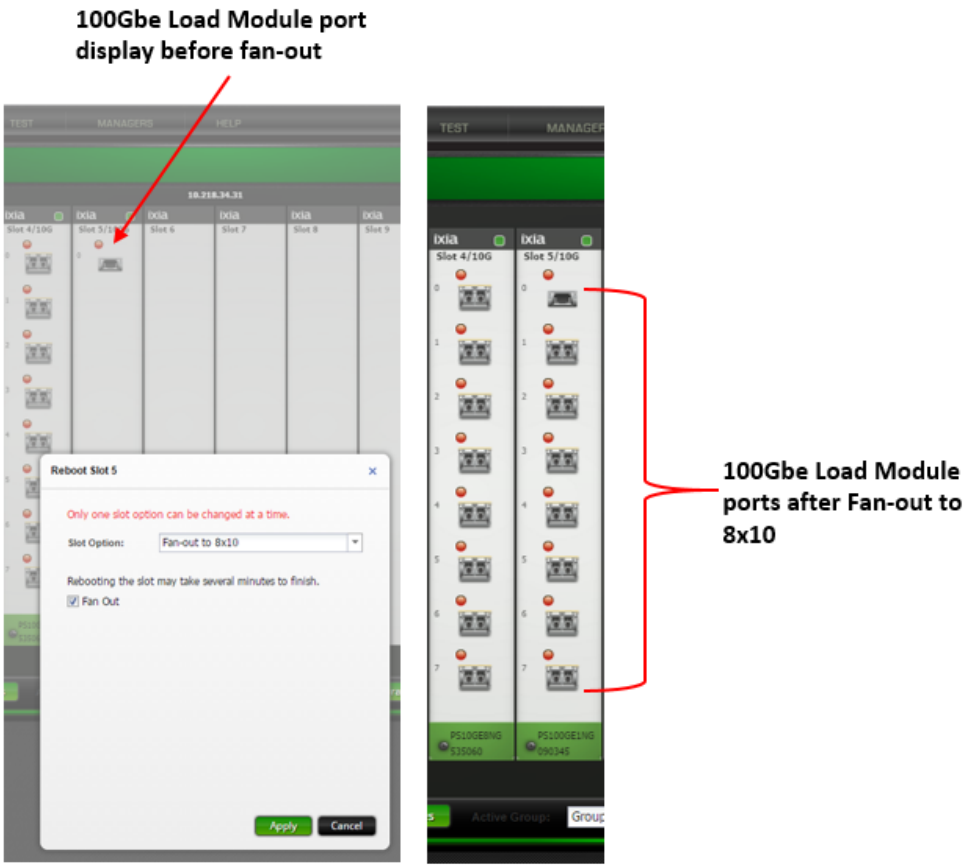
Configuring PerfectStorm Speed

To set the load module in fan-out mode:

1. In the UI, click on the gear at the top of a 100G or 40G card.
2. Select a fan-out slot option.
3. 40Gbe: Fan-out to 1 x 40 or 40Gbe or Fan-out to 4 x 10,
4. 100Gbe: Fan-out to 1 x 100 or Fan-out to 2 x 40 or Fan-out to 8 x 10
5. Select the **Fan Out** check box.
6. Click Apply.

The load module will reboot. After the reboot has completed, the port options displayed on the load module will reflect the new configuration.

The image below displays how the GUI is updated after a fan-out option has been applied.



Note:

- Fan-out options can also be configured in IxExplorer by right-clicking on the card and selecting, **Properties > Enable Ports >**, then select the desired configuration.
- If a user reserves two 100GbE ports on adjacent (odd slot/even slot) load modules, the load modules will automatically be configured as partners for testing purposes. Modules are partnered in an odd slot/even slot manner, for example, slots 1&2 and slots 3&4 would be partnered. Modules in slots 2&3 would not be partnered.

PerfectStorm Fan-out Physical/Logical Connections

The image below provides an example of how fan-out cables are connected to a load module.

100GbE Standard Fan-out Cable

The following table describes your ability to use the standard cable that ships with a 100Gbe Load Module to support several connection scenarios based on the fan-out mode selected in the GUI.

100Gbe Fan-Out Cable – DUT Side

	Cable 1	Cable 3	Cable 3	Unlabeled Cable
DUT Interfaces	1 x 40Gbe	Not used	1 x 40Gbe	1 x 100Gbe or 8 x 10Gbe
Fan-out mode selected in GUI	Fan-out to 2x40	n/a	Fan-out to 2x40	Fan-out to 1 x 100 or Fan-out to 8 x 10
Ports numbers onload module in GUI	0 – 3	n/a	4 – 7	1 x 100 = 1 8 x 10 = 1 - 8
Notes				Maximum traffic throughput at 100Gbe = 80Gb

40GbE Standard Fan-out Cable

The following table describes your ability to use the standard cable that ships with a 40Gbe Load Module to support four 10Gbe connections when the Fan-out to 4 x 10 slot mode is selected in the GUI.

40Gbe Fan-Out Cable – DUT Side

	Cable 1	Cable 3	Cable 3	Cable 4
DUT Interfaces	1 x 10Gbe	1 x 10Gbe	1 x 10Gbe	1 x 10Gbe
Fan-out mode selected in GUI	Fan-out to 4 x 10	Fan-out to 4 x 10	Fan-out to 4 x 10	Fan-out to 4 x 10
Ports numbers on load module in GUI	0	1	2	3

CHAPTER 13 Strike List

This section covers:

Strike List Overview

The Strike List is the central location for customizing attack traffic. From the Strike List, you can customize attacks by grouping Strikes together. The Evasion Profile settings establish the evasion techniques for a single group of Strikes.

Note: The StrikeVariants feature requires release 3.4 firmware and the installation of ATI Update ATI-2015-01 (or later). If ATI-2015-01 (or later) is not installed to the BreakingPoint system, Variations will appear as an empty group in the list of Evasion Profiles. ATI updates can be downloaded from the Ixia website at: https://support.ixiacom.com > **Software Downloads** > **BreakingPoint Software**. Note that you will need to log in to access this section of the support website.

Several terms that are specific to the BreakingPoint system's security suite: Strike, Smart Strike List, Evasion Profile settings, and Strike Lists. For a better understanding of these terms, see the list below.

Strike List Terminology

Term	Definition
Strike	An attack that exploits or exposes vulnerabilities
Smart Strike List	A list of strikes that is automatically updated to include new ATI Update items that relate to the list

Term	Definition
Evasion Profile settings	Evasion techniques for an Evasion Profile
Strike List	A collection of Strikes

Strike List Security Group

The Strike List is the top-level security group. It contains all of the attacks and evasion options that will be used in a Security test. You can use any default Strike List to exploit vulnerabilities in various hosts and applications; however, if you need more granular control over the attack traffic, you can customize your own attacks through the Strike List.

When you create a new Strike List, it will have its own set of options that determine which evasion techniques to use in the attack traffic. You can create as many strike lists as you want. Remember that each strike list will have its own set of Strike Options, so you should create a strike list for each unique set of evasion options that you need.

Default Strike List

By default, BreakingPoint provides you with a set of default strike lists. These strike lists have been custom designed by the BreakingPoint Security team to target specific types of security testing.

The list below provides a sample of the most commonly used strike lists that come packaged with the system.

Sample of BreakingPoint Strike Lists

Strike List	Description
All Strikes	Contains all Strikes on the system. This Strike List can take over a day to complete.
Backdoor Strikes	Contains all Strikes that can simulate trojans and backdoor network activity.
Clientside Strikes	Contains all Strikes that simulate exploiting of vulnerabilities in client applications, usually involving file transfers.
Denial of Service Strikes	Contains all Strikes that can trigger denial of service flaws.
Important Strikes	All Strikes with CVSS score > 7.0.
Microsoft Strikes	All Strikes for Microsoft-related vulnerabilities.
Mobile Malware Strikes	All Strikes sending mobile malware.

Strike List	Description
Protocol Fuzzers	Contains all Strikes related to protocol fuzzing.
Reconnaissance Strikes	Contains all Strikes that can simulate attacks that gather information.
SCADA Strikes	All Strikes targeting SCADA applications.
Strike Level 1, 2014	All Strikes for vulnerabilities with CVSS score of 10.0, disclosed in 2014.
Strike Level 3, 2013	All Strikes for vulnerabilities disclosed in 2013.

Strike List Manager

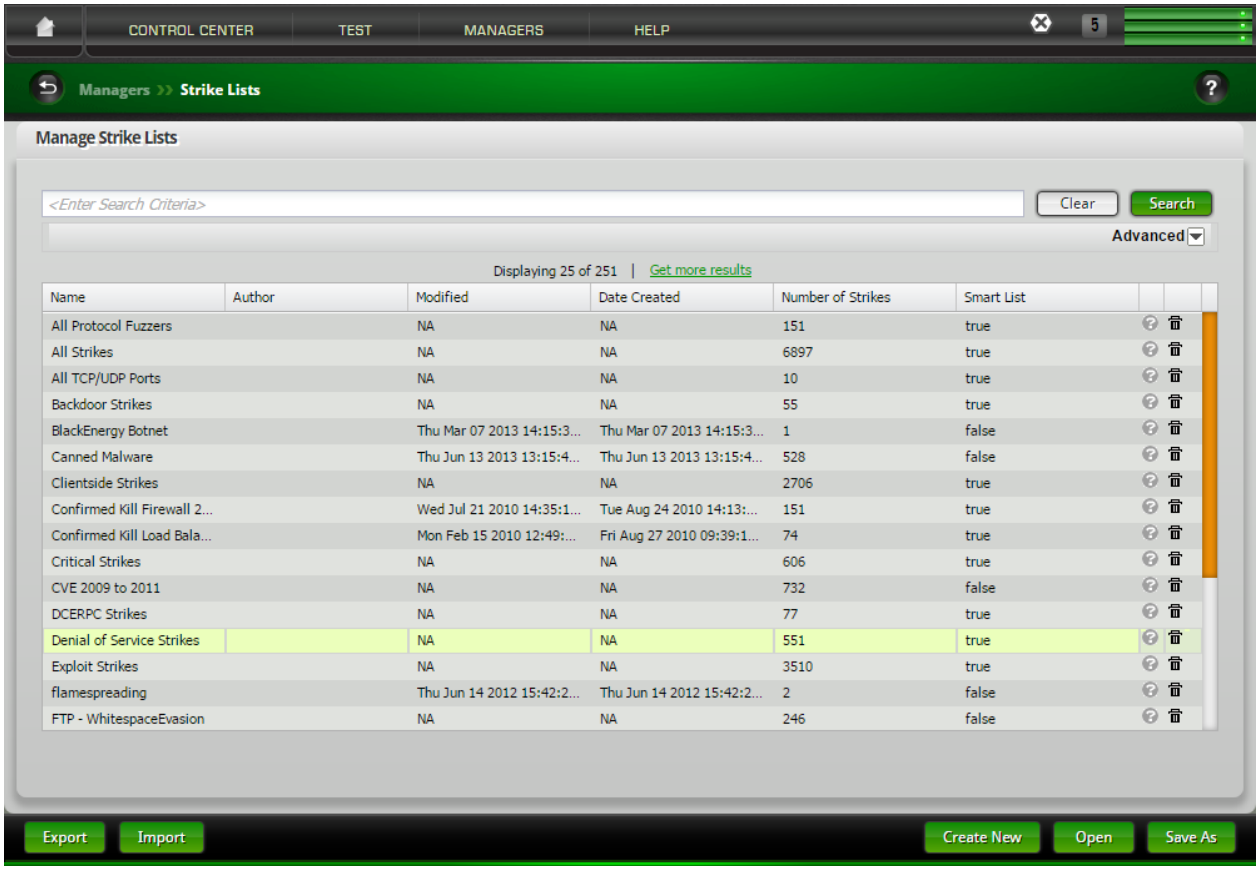
The Strike List Manager allows you to search for individual strikes and Strike Lists. You also have the ability to create your own customized list of strikes as well as Smart Strike Lists. Smart Strike Lists are automatically updated to include new ATI Update items that relate to the list.

To open the Strike List Manager select:

Managers > Strike Lists

The Strike List Manager displays and allows Strike Lists to be:

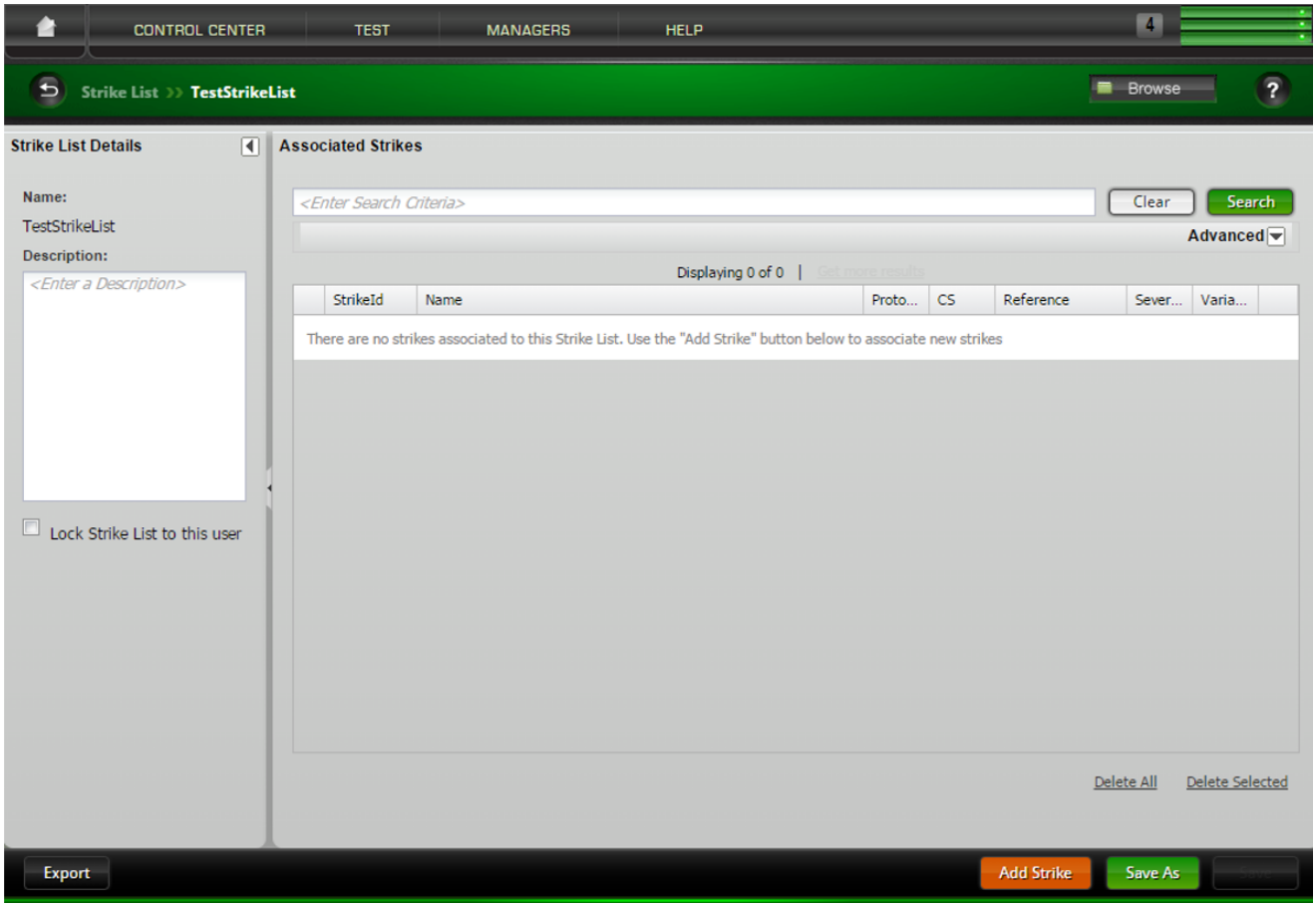
- Viewed
- Edited (Including adding and removing strikes. Double-click the Strike List or select the Strike list and click **Open**.)
- Exported
- Imported
- Cloned (Saved under a different name using the **Save As** button.)
- Created



To search for specific Strike Lists, enter a search criteria and click the **Search** button. Detailed information about the Search feature is provided later in this section.

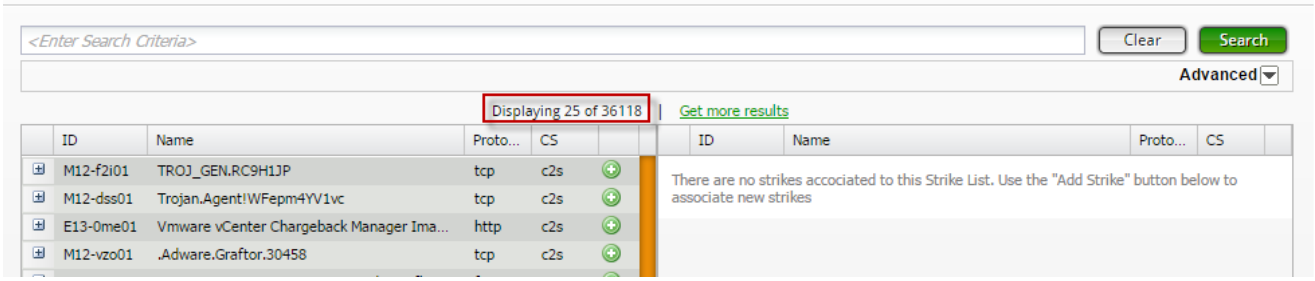
Click the **Create New** button to begin the process of creating a new strike list. A Strike List Name dialog will display.

Enter a name for the Strike List. Click **OK**. The Strike List Configuration window displays.

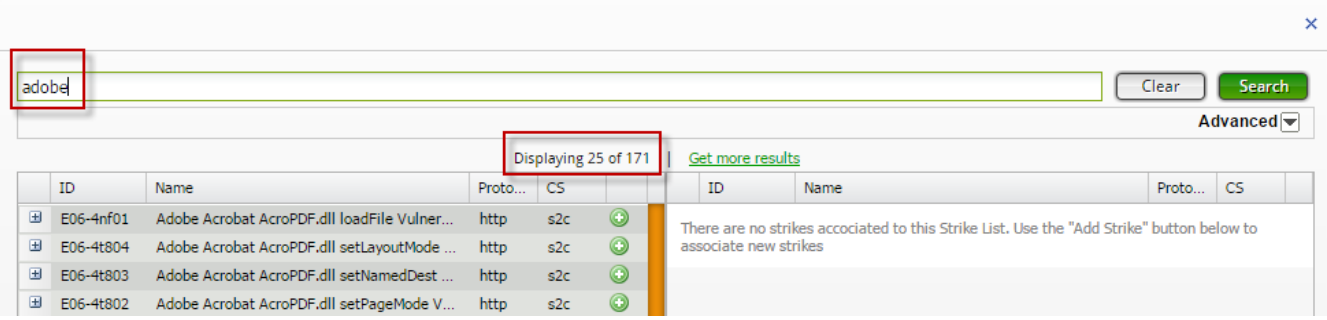


Optionally, you can enter a Description for the Strike List.

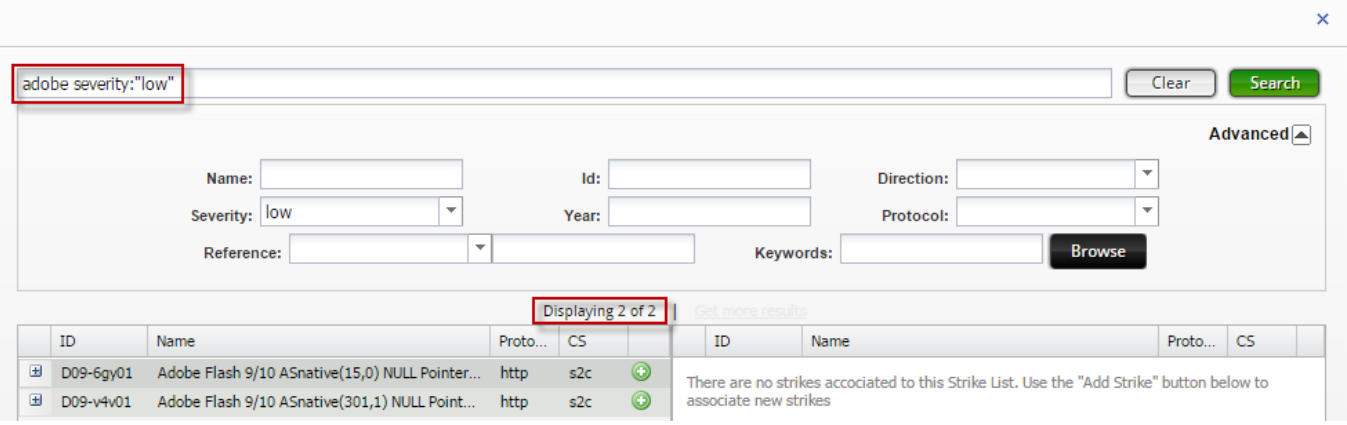
Click the **Add Strike** button to add strikes to the Strike List. The Strike Selector window displays.



Initially, all of the strikes in the strike library are available and can be displayed in the left-side panel.



A search for a specific strike, for example “adobe”, will refine the list of candidates for the Strike List down to 171 strikes.



Adding “low” severity to the criteria in the **Advanced** option section further reduces the list of candidates to 2 strikes. Also note how the text, “severity:“low” has been added to “adobe” in the search criteria window. For additional information on manually entering this type of advanced search criteria, See Advanced Strike Searches.

Move a strike from the left-side panel to the right-side panel to add the strike to the strike list.

Clicking the **Add All** button (or Add All link) will move all of the strikes displayed in the right-side panel to the strike list. To add individual strikes to the strike list, click the green encircled "+" (plus symbol) that is located at the right side of a strike's name (or click the Add Selected link). To add several specific strikes to the strike list simultaneously, use standard Windows multi-select functionality (Shift key - drag or Ctrl key - select) to select the strikes and then click the **Add Selected** button (or click the Add Selected link).

Note: New searches can be implemented at any time by entering a criteria and clicking the Search button. New search results will be displayed in the left-side panel while items that have been moved to the strike list (right-side panel) based on earlier searches will be maintained.

Optionally, select the **Smart Strike List** option to store only the defined search criteria in a Smart Strike List.

A Smart Strike List is a strike list that is automatically updated to maintain the latest strikes. To stay current, a Smart Strike List queries the system strike library. When new strikes are found based on the

saved search criteria, those strikes are added to the Smart Strike List. The system strike library is updated when a user installs an ATI update (updates are released by on a bi-weekly basis).

Note: Clicking the Smart Strike List check box will remove any strikes that a user has manually added to a Strike List.

Click **OK**.

Click **Save**.

Searching for Strikes

With the Strike List page, you can search for individual strikes based on details such as protocol, strike, direction, run ID, model name, keyword, path ID, or a category ID. To perform a search, enter one of the items listed into the Search field on the Strike List page. To narrow your search, you can enter more than one item into the search field.

Note: To view the details of a strike, such as its path, category, available keywords, or model name, right-click on the strike and select View Strike Details.

The default search capability (no keywords) will search for a string anywhere in the description. For example, if you search for HTTP, you will receive results for strikes against other protocols if HTTP is anywhere in the description. If you only want strikes against the HTTP protocol, use the protocol:http search operation.

Query Strings For Strikes below contains some of the query strings that can be used to search for specific types of strikes contained in your tests. Enter these query strings into the search field to narrow your search.

Query Strings For Strikes

Query Type	Description	Query String	Example
category	Lists strikes that belong to the category that you specify.	category:category	category:Exploits: Web Application Cookie
categoryid	Lists strikes that contain the details that you specify.	categoryid:categoryid	categoryid:/strikes/exploits/ftp/ categoryid: Exploits
deprecated	Lists strikes that are denoted as deprecated.	deprecated	deprecated

Query Type	Description	Query String	Example
direction	Lists strikes that contain the directionality (c2s – meaning client to server, s2c – meaning server to client, etc.) that you specify.	direction:direction	direction:c2s
id	Lists strikes that contain the ID that you specify.	id:id	id:d10
keyword	Lists strikes that contain the keyword you specify.	keyword:keyword	keyword:ms_2010-07
modelname	Lists strikes associated with the test name that you specify.	modelname:test name	modelname:0-sc
name	Lists strikes that contain the details that you specify.	name:name	name:ActiveX
pathid	Lists strikes included in the path that you specify.	pathid:path	pathid:/strikes/denial/browser/aol_activex_cookie.xml

Query Type	Description	Query String	Example
protocol	Lists specified strikes contained in the test that include the specifed protocol.	protocol:protocol	protocol:http
reflabel	Lists strikes in the test that contain the specifed reference.	reflabel:reference label	reflabel:xyz
reftype	Lists strikes that contain the reference id number that you specify.	reftype:reference id number	reftype:BPS 2010-0001
runid	Lists strikes that were blocked, errored, or allowed in the specified test.	runid:Blocked:internal ID runid:Errored:internal ID runid:Allowed:internal ID*	runid:Blocked:684
strikelist	Lists strikes that contain a portion of the details that you specify in the name of the strike.	strikelist:strikelist	strikelist:Apache
strikelistname	Lists strikes that contain the exact details that you specify in the name of the strike.	strikelistname:strikelistname	strikelistname:Apache File Access:.http

Query Type	Description	Query String	Example
year	Lists strikes that were created in the year that you specify.	year:year	year:2009
* The internal ID can be found at the end of the test report URL.			

Example 1

To search for all HTTP attacks, enter the following search operation:

```
protocol:http
```

Example 2

To search for all clientside HTTP attacks, enter the following search operation:

```
protocol:http
direction:c2s
```

Example 3

To search for all clientside http and ftp attacks:

1. Enter `protocol:http direction:c2s` in the Search field.
2. Right click Select All.
3. Replace search criteria with protocol:ftp direction:c2s
4. Right click Select All.

Runid Query

You must always include the runid query in conjunction with the other queries in order to use the other queries to conduct a search on a specific test. For example, if you want to know which strikes were http-based for test 653, enter the following query string into the search field:

```
runid:653
protocol:http
```

Search Operators

contains the search operators that are currently available for the Strike List. These operators help to enhance the search capabilities of the Strike List page by giving you the ability to narrow your search. You can use one or more of the available search operators in a single search.

Search Operators

Available Operator	Meaning	Example
OR	Include any of these items in the search.	T1 OR T2 would mean search for items including T1 or T2.
–	Exclude these items from search. Do not place a space after the " – " operator. Doing so will return no results.	cve –c2s would mean search for items that are cve but not c2s.
"term"	Search for this exact term.	"abc" "def" would mean search for items that contain both abc and def.

Example

To return all strikes that are not malware, enter the following query string into the search field:

```
$bps searchStrikes -limit 5000 " -
strikes/malware"
```

To search for Strikes:

Select Managers > Strike List > from the Menu bar. Click the **Create New** button and then click the **Add Strike** button.

Enter your search criteria into the Search field. Your search criteria can consists of a protocol, strike, Strike List, run ID, model name, keyword, path ID, or a category ID. You can base your search on one, or a combination of any of these items.

Note: To view the details of a strike, such as its path, category, available keywords, or model name, click the "+" (plus symbol) at the left of the strike's name.

Note: When using multiple search criteria in a single search, be sure to use one or more of the available search operators.

Press the **Enter** key.

Searching for Strike Lists

You can also perform advanced searches for Strike Lists.. The Strike List Manager allows you to search for Strike Lists based on details such as author, title, and class.

The table below contains some of the query strings that can be used to search for specific Strike Lists. Enter these query strings into the Search Strike Lists field of the Browse Strike Lists page to narrow your search.

Query Strings Used To Search for Strike Lists

Query Type	Description	Query String	Example
author	Lists Strike Lists according to the name of the user that last modified the strike list.	author:name of last editor	author:BreakingPoint
createdBy	Lists Strike Lists according to the name of the user that created the strike list.	createdBy:name of creator	createdBy:BreakingPoint
class	Lists Strike Lists according to the class that the Strike List belongs to.	class: class type	class: exploit
title	Lists Strike Lists that contain portions of the title specified.	title:title	title:Strike Level

To search for Strike Lists:

1. Select **Managers > Strike List** > from the Menu bar. The Strike Lists Manager is displayed.
2. Enter your search criteria into the Search Criteria field.
3. Press the **Enter** key.

The Order of Strikes

The order in which strikes are sent depends on whether individual strikes or Smart Strike Lists have been added to the Strike List. If strikes have been individually added, then they will be sent out in the order they are listed in the Strike List. If Smart Strike Lists are included in the Strike List, then all strikes will be sent in a random order.

Evasion Profile Settings

Evasion Profile settings establish the evasion techniques that are available for a Strike List. The values defined for Evasion Profile settings will override the default values defined in a Strike List. Tables through list all the Evasion Profile settings that are available.

Upper-layer protocols will inherit the option settings for lower layer protocols. For example, HTTP option types will inherit TCP and IP option settings because they are part of the same TCP/IP stack.

COMMAND Settings

COMMAND Settings below lists the Evasion Profile settings for COMMAND.

COMMAND Settings

Option	Description	Valid Values
PadCommandWhitespace	Pads the whitespace between commands and arguments using space and tab characters	true or false

Option	Description	Valid Values
Malicious	Sends commands that are intended to do nefarious things, such as delete files Ethernet	true or false
PadPathSlashes	Pads UNIX path names using space and tab characters	true or false

DCE/RPC Settings

DCE/RPC Settings below lists the Evasion Profile settings for DCE/RPC.

DCE/RPC Settings

Option	Description	Valid Values
MaxFragmentSize	Maximum fragment size for DCE/RPC requests	1 – 65,535
MultiContextBind	Hides the real bind request between fake UUIDs	true or false
MultiContextBindHead	Number of fake UUIDs that occur before the real UUID	0 – 65,535
MultiContextBindTail	Number of fake UUIDs that occur after the real UUID	0 – 65,535
UseObjectID	Specifies a fake object ID on all call requests	true or false

EMAIL Settings

EMAIL Settings below lists the Evasion Profile settings for EMAIL. All settings defined for EMAIL will affect the following protocols: SMTP, POP3, and IMAP.

EMAIL Settings

Option	Description	Valid Values
EnvelopeType	Determines whether the To and From fields in the header are system generated or user-defined	**User-specified** – User defines the To/From fields for email headers **System-generated** – System generates random To/From fields for the email header
From	Defines the From header in email messages if **EnvelopeType** is **User-specified**	String value (0 – 128 character length)
To	Defines the To header in email messages if **EnvelopeType** is **User-specified**	String value (0 – 128 character length)

Option	Description	Valid Values
ShuffleHeaders	Randomizes the order of headers in this protocol.	true or false

Ethernet Settings

Ethernet Settings below lists the Evasion Profile settings for Ethernet.

Ethernet Settings

Option	Description	Valid Values
MTU	Specifies the Maximum Transmission Unit used to send frames	64 – 9216

FILETRANSFER Settings

FILETRANSFER Settings below lists the Evasion Profile settings for FILETRANSFER.

FILETRANSFER Settings

Option	Description	Valid Values
Pop3Encoding	Encoding for files transported via POP3.	base64 or quoted-printable
TransportProtocol	Different transport protocols to use when sending files.	FTP, HTTP, IMAP4, POP3, or SMTP
Imap4Protocol	Encoding for files transported via IMAP4.	base64 or quoted-printable
FtpTransferMethod	The FTP method to be used to transfer the file.	PASV_RETR, RETR, or STOR
CompressionMethod	Different compression methods to use when transferring files.	Gzip, None, Tar, Tgz, or Zip
SmtpEncoding	Encoding for files transported via SMTP.	base64, quoted-printable, or uuencode

FTP Settings

FTP Settings on the next page lists the Evasion Profile settings for FTP.

FTP Settings

Option	Description	Valid Values
AuthentificationType	Determines whether the FTP server authentication is user-defined or system generated.	**System Generated** – System generates the password and username **User-specified** – User defines the password and username
FTPEvasionLevel	Alters the FTP commands with the selected telnet control character option	**0** – No telnet opcode evasion **1** – Single telnet opcode placed at the beginning of the command **2** – Single telnet opcode that is randomly placed in the command **3** – Multiple telnet opcodes placed at the beginning of the command **4** – Multiple telnet opcodes that are randomly placed in the command **5** – One telnet opcode per word **6** – One telnet opcode per character
PadCommandWhitespace	Pads the whitespace between commands and arguments using space and tab characters	**1** – Always use whitespace evasion **2** – For Evasion Levels higher than zero, use whitespace evasion. Otherwise, do not. **3** – Never use whitespace evasion
Password	Defines the password used for FTP connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)

Option	Description	Valid Values
Username	Defines the username used for FTP connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)

Global Settings

Global Settings below lists the Evasion Profile settings for Global.

Global Settings

Option	Description	Valid Values
AllowDeprecated	Allow deprecated strikes to run	true or false
FalsePositives	Run strikes in false positive mode	true or false
Allow Override		

HTML Settings

HTML Settings below lists the Evasion Profile settings for HTML.

HTML Settings

Option	Description	Valid Values
HTMLUnicodeEncoding	Uses Unicode encoding for HTML content	None UTF-7 UTF-8 UTF-16BE (big-endian) UTF-16LE (little-endian) UTF-32BE (big-endian) UTF-32LE (little-endian)
HTMLUnicodeUTF7EncodingMode	Uses the Unicode UTF-7 character encoding mode for HTML content	**Standard** – Do not encode alphanumeric characters in accordance with UTF-7 encoding **All** – Encode all characters with UTF-7 character encoding

Option	Description	Valid Values
HTMLUnicodeUTF8EncodingMode	Uses the Unicode UTF-7 character encoding mode for HTML content	**Overlong** – Encode characters using alternate UTF-8 encoding. **Invalid** – Encode invalid characters with alternate UTF-8 invalid overlong encoding.
HTMLUnicodeUTF8EncodingSize	Defines the number of alternate whitespace characters to prepend	2 – 7

HTTP Settings

HTTP Settings below lists the Evasion Profile settings for HTTP.

HTTP Settings

Option	Description	Valid Values
AuthenticationType	Determines whether the HTTP server authentication is user-defined or system generated	**System-generated authentication** – System generates the password and username **User-specified authentication** – User defines the password and username
Base64EncodePOSTData	Encode POST data using Base64 encoding	true or false
ClientChunkedTransfer	Uses chunked transfer-encoding to separate the client requests	true or false
ClientChunkedTransferSize	Defines the maximum chunk size for **ClientChunkedTransfer**	1 – 4,294,967,295
DirectoryFakeRelative	Inserts fake relative directories between path elements	true or false
DirectorySelfReference	Converts all directories to self-referenced relative directories	true or false
EncodeDoubleNibbleHex	Encode each hex nibble of URL characters separately	true or false

Option	Description	Valid Values
EncodeDoublePercentHex	Double encode URL characters	true or false
EncodeFirstNibbleHex	Double encode the first hex nibble of URL characters	true or false
EncodeHexAll	Encodes the entire URL in Hex	true or false
EncodeHexRandom	Encodes random parts of the URL in Hex	true or false
EncodeSecondNibbleHex	Double encode the second hex nibble of URL characters	true or false
EncodeUnicodeAll	Encodes the entire URL in Unicode	true or false
EncodeUnicodeBareByte	Encodes the entire URL in Unicode	true or false
EncodeUnicodeInvalid	Encodes the URL with invalid Unicode	true or false
EncodeUnicodePercentU	Encode the request with 16-bit percent-U unicode	true or false
EncodeUnicodeRandom	Encodes random parts of the URL in hex	true or false
EndRequestFakeHTTPHeader	Encode an HTTP header in the URL	true or false
ForwardToBackSlashes	Converts all forward slashes to back slashes (Windows only)	true or false
GetParameterRandomPrepend	Prepends random values to the query string	true or false
HTTPServerProfile	Configures evasion options based on HTP server compatibility	none, iis, or apache
MethodRandomInvalid	Uses a random invalid method	true or false
MethodRandomValid	Uses a random valid method	true or false
MethodRandomizeCase	Randomizes the case of the request method	true or false
MethodURINull	Inserts a null character between the method and URL in HTTP requests	true or false

Option	Description	Valid Values
MethodURISpaces	Insert multiple spaces between the method and URL in HTTP requests	true or false
MethodURITabs	Inserts tab characters between the method and URL in HTTP requests	true or false
Password	Defines the password used for HTTP connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)
PostParameterRandomPrepend	Prepends random values to the POST data	true or false
RequestFullURL	Uses the full URL in the request URL	true or false
ServerChunkedTransfer	Uses chunked transfer-encoding to break up the server response	true or false
ServerChunkedTransferSize	Defines the chunk size for **ServerChunkedTransfer**	1 – 4,294,967,295
ServerCompression	Uses compression to encode the server response	none, delfate, gzip
URIAppendAltSpaces	Appends whitespace characters to the URL	true or false
URIAppendAltSpacesSize	Defines the number of whitespace characters to append to the URL if **URIAppendAltSpaces** is true	0 – 65,535
URIPrependAltSpaces	Prepends random whitespace characters to the URL	true or false
URIPrependAltSpacesSize	Defines the number of whitespace characters to prepend to the URL if **URIPreprendAltSpaces** is **true**	0 – 65,535
URIRandomizeCase	Randomizes the case of the request URL	true or false
Username	Defines the username used for HTTP connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)
VersionRandomInvalid	Uses a random string for the HTTP version	true or false

Option	Description	Valid Values
VersionRandomizeCase	Randomizes the case of the HTTP version	true or false
VersionUse0_9	Uses HTTP version 0.9 instead of 1.0 or 1.1	true or false
VirtualHostname	Modifies the type of HTTP host header sent	String value (0 – 32 character length)
ShuffleHeaders	Randomizes the order of headers in this protocol.	true or false
VirtualHostnameType	Defines the HTTP header used if **VirtualHostname** is set to **Custom**	**1** – System-generated HTTP host header **2** – User-specified HTTP host header

IMAP4 Settings

IMAP Settings below lists the Evasion Profile settings for IMAP.

IMAP Settings

Option	Description	Valid Values
AuthenticationType	Determines whether the IMAP server authentication is user-defined or system generated	**System-generated authentication** – System generates the password and username **User-specified authentication** – User defines the password and username
Password	Defines the password used for IMAP connections if **AuthenticationType** is **User-specified authentication**	String value (0 – 32 character limit)
Username	Defines the username used for IMAP connections if **AuthenticationType** is **User-specified authentication**	String value (0 – 32 character limit)

IP Settings

IMAP Settings on the previous page lists the Evasion Profile settings for IP. All settings defined for IP will affect the following protocols: TCP and UDP.

IP Settings

Option	Description	Valid Values
FragEvasion	Enables IP evasion using fragmentation	Disabled **Overlap-Last-New** – Overlap end fragments, favoring new data (Linux/IOS) **Overlap-Last-Old** – Overlap end fragments, favoring old data (Windows/Solaris/BSD) **Overlap-All-New** – Overlap all fragments, favoring new data (IOS) **Overlap-All-Old** – Overlap all data, favoring old data (Windows/Solaris/BSD)
FragOrder	Changes the order in which fragments are sent	default, random, or reverse

Option	Description	Valid Values
FragPolicy	Determines how IP fragments are reassembled	**Last** – Newer fragments always replace older fragments (Cisco IOS) **First** – Older fragments are never replaced by new fragments (SunOS 5.5-5.8, HP-UX 11i) **Linux** – Reassemble the fragments according to Linux/OpenBSD IP stack behavior **Bsd** – Reassemble the fragments according to BSD IP stack behavior (AIX, BSD, or Irix) **Bsd-Right** – Reassemble the fragments according to HP JetDirect IP stack behavior **Windows** – Reassemble the fragments according to Windows IP stack behavior **Solaris** – Reassemble the fragments according to Solaris 9/10 IP stack behavior
MaxFragSize	Defines the maximum packet size for all transactions	8 – 65,535
IPEvasionsOnBothSides	Allows the use of IP evasions on both sides, regardless of Strike direction	true or false
MaxReadSize	Defines maximum IP packet size for server-to-client transactions	8 – 65,535
MaxWriteSize	Defines the maximum IP packet size for client-to-server transactions	8 – 65,535
RFC3514	Enables RFC3514 compatibility	true or false

Option	Description	Valid Values
TOS	Defines the TOS field for all packets	0 – 255
TTL	Defines the TTL field for all packets	0 – 255

JAVASCRIPT

JAVASCRIPT Settings below lists the Evasion Profile settings for JAVASCRIPT.

JAVASCRIPT Settings

Option	Description	Valid Values
Encoding	Specify a JAVASCRIPT encoding method	HUFFMAN – Huffman encode the JAVASCRIPT (compress it) XOR – XOR encode the JAVASCRIPT none – Do not encode the JAVASCRIPT
Obfuscate	Obfuscate JAVASCRIPT code in order to make signature-based detection more difficult	true or false

Variations

VARIATIONS Settings below lists the Evasion Profile settings for VARIATIONS.

VARIATIONS Settings

Option	Description	Valid Values
TestType	Select how variants will be replayed during the test	**AllVariants** - Replay all variants for all Strikes in a Strikelist **VariantLimit** –When selected, the **Limit** value is used to set the maximum number of variants to generate for each Strike **VariantSubset** - When selected, the **Subset** field value is used to specify the Strike variant number(s) to replay
Limit	Specify the maximum number of variants to generate for each strike	0-50000

Option	Description	Valid Values
Subset	Specify a subset of strike variations by number using a comma separated list	Example: 1,3,5-8,14,440
VariantShuffle	Randomizes the order of the variants	n/a

StrikeVariant Testing

StrikeVariant testing is implemented using an evasion profile that is designed to enumerate all possible variants of a Strike in order to allow individual testing of each specific variant. This feature is designed to assist users who want to ensure identification and/or coverage for all possible exploit variations for a vulnerability.

Within a given strike there are functions that represent a selection of a code path from one or more possible code paths that are used to define and format the data that is placed on the wire. Those functions are:

- rand_select
- false_positive

Note:

- Only Strikes that are written using the above mentioned functions will have variations enumerated using the StrikeVariants testing method. Strikes written without these functions will have a VariantCount of 1.
- When a seed value is used with this feature, the exact same payload contents can be regenerated across systems.
- Because StrikeVariants are generated at runtime, test initialization times can increase substantially. The more variants a strike has, the more time initialization will take. Some Strikes have more than 300,000 variants - and test initialization may take multiple hours or possibly days.
- The StrikeVariants method is used to test code paths; it does not enumerate all possible values for the functions used. For example, if a strike selects between two HTTP parameters, each of the parameters represents a code path. But the values for those parameters will be determined by the functions used and do not represent individual code paths.

StrikeVariant Test Example

The StrikeVariants feature is implemented as an Evasion Profile. In the following example, a test will be configured to replay specific Variants.

1. Create a new test (**Test** > **New Test**).
2. Add a new Security Test Component (Add New > Security).
3. In the **Add a Security** window, add a Strike List then click **Create.**

4. Edit the new Security test component that you created (double-click the name or right-click > Edit).
5. Edit the Evasion Profile (at the Evasion Profile field, click the **Edit** button).
6. Under the **Variations** section, select the **Allow Override** and **VariantTesting** check boxes.
7. At the right side of **TestType** field, select **Allow Override.**
8. From the **TestType** dropdown, select **VariantSubSet.**
9. At the right side of the **Subset** field, select the **Allow Override** checkbox.
10. In the **Subset** field, enter a comma-delimited list of Variant numbers, (e.g. 1,3,5-8,14,440).
11. **Save** the Evasion Profile (or **Save As** if you are using a BP provided Strike List).
12. Click **Return to the Workspace.**
13. **Save** the test.
14. Run the test.

Malware Settings

While BreakingPoint makes thousands of pieces of live malware available via the BreakingPoint Strike List, occasionally you may want to run tests using your own malware. This capability is now available when you download the most current ATI Update.

Adding Customer-Supplied Malware Files to a Strike List

Running malware through a network can corrupt the network when proper precautions are not taken. Apply the following steps to ensure that malware files are not accidentally placed into a live state.

To place malware files into a Strike List:

1. Upload the customer_malware.tar file onto your BreakingPoint device into the standard resources directory.
2. Select **Managers** > **Strike Lists** from the Menu bar.
3. Enter `Customer Supplied Malware` into the Search field.
4. Select and right-click the Strike.
5. Select **Add Strike**.
6. Select **Strike List** from the Menu bar.
7. Select **Save As**.
8. Enter the name of the Strike List and click Ok.

Preparing Evasion Profile Settings

Use the FILETRANSFER options to change the Evasion Profile for your malware. The default settings for the FILETRANSFER Evasion Profile will be sufficient for most tests. However, you have the option of customizing the settings to your preference.

If you change any Evasion Profile settings, be sure to click the Apply Changes and the Save As buttons. Enter a name for the updated Evasion Profile and click OK. The FILETRANSFER options will be applied to the contents of the previously uploaded customer_malware.tar file.

Malware Settings on the facing page lists Evasion Profile settings for Malware.

Malware Settings

Option	Description	Valid Values
CompressionMethod	Different compression methods to use when sending live malware samples	none Gzip Tar Tgz Zip
SmtpEncoding	Encoding for malware transported via SMTP	base64 quoted-printable uuencode
Imap4Encoding	Encoding for malware transported via IMAP4	base64 quoted-printable uuencode
TransportProtocol	Different transport protocols to use when sending live malware samples.	HTTP IMAP4 POP3 SMTP
Pop3Encoding	Encoding for malware transported via POP3.	base64 quoted-printable uuencode
FtpTransferMethod	The FTP method to be used to transfer malware samples	PASV_RETR RETR STOR

OLE Settings

OLE Settings below lists Evasion Profile settings for OLE.

OLE Settings

Option	Description	Valid Values
RefragmentData	Output fragmented OLE documents	**true or false**

POP3 Settings

POP3 Settings below lists Evasion Profile settings for POP3.

POP3 Settings

Option	Description	Valid Values
AuthenticationType	Determines whether the IMAP server authentication is user-defined or system generated	**System-generated** – System generates the password and username **User-specified** – User defines the password and username
PadCommandWhitespace	Uses space and tab characters to pad the whitespace between commands and arguments	true or false
Password	Defines the password used for POP3 connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)
Username	Defines the username used for POP3 connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)

SELF Settings

SELF Settings below lists Evasion Profile settings for SELF

SELF Settings

Option	Description	Valid Values
AS-ID	Identifies the number used in the BGP protocol to identify an autonomous system	1 – 65,535
URL	Uniform Resource Identifier, or requested address	Any valid string
ROUTER-ID	Defines the router id to send in OSPF packets to identify a router	Any valid string

Option	Description	Valid Values
UnicodeTraversalVirtualDirectory	The virtual directory to use in IIS unicode execution strikes	scripts, msadc, iisadmpwd, _vti_bin, exchange, cgi-bin, pbserver
TraversalVirtualDirectory	The virtual directory to use in an Apache Win32 directory traversal strike	scripts or cgi-bin
UnicodeTraversalWindowsDirectory	The Windows directory to use in IIS unicode execution strikes	winnt or windows
EndingFuzzerOffset	The iteration of the fuzzing test that signals the end of the test	0 – 2,147,483,647
StartingFuzzerOffset	The iteration of the fuzzing test that signals the start of the test	0 – 2,147,483,647
Username	Defines the username to send in connections that require a username	Any valid string
Password	Defines the password to send in connections that require a password	Any valid string
MaximumRuntime	Specify the maximum amount of time a long-running strike will be run (in seconds)	0 – 86,400
Report Individual CLSIDs	Toggles verbose logging for the Killed ActiveX Instantiation strike, allowing it to report allowed or blocked status on a per-CLSID basis	true or false
TraversalRequestFilename	The filename to use in an Apache Win32 directory traversal strike	win.ini or system.ini
ApplicationPings	The choice as to whether or not to ping an application	on or off
HTMLPadding	The amount of padding to be used with HTML packets	0 – 2,147,483,647

Option	Description	Valid Values
MaximumIterations	Specifies the number of times each subtest within the test is repeated (used with fuzzers)	0 – 2,147,483,647
Repetitions	Specifies the number of times the test is to be run (used with single flows)	0 – 2,147,483,647
TraversalWindowsDirectory	The Windows directory to use in an Apache Win32 directory traversal strike	winnt or windows
AREA-ID	The OSPF Area ID which identifies which area routers belong to	Any valid 32-bit identifier
DelaySeconds	A delay in seconds for use in sending flows	0 – 2,147,483,647
AppSimSmartflow	An AppSim Super Flow Fuzzer that fuzzes various fields in an existing Super Flow	Valid Super Flow
AppSimSuperflow	An AppSim Super Flow Fuzzer that runs an existing Super Flow with the user controlling the number of times to repeat, and the time between consecutive runs	Valid Super Flow
AppSimAppProfile	This parameter defines the Application Profile that the strike will use. Used by strike: /strikes/generic/appprofileflow/appprofileflow.xml.	Valid App Profile
AppSimUseNewTuple	Use a new tuple for each Superflow	true or false

SHELLCODE Settings

SHELLCODE Settings below lists the Evasion Profile settings for SHELLCODE.

SHELLCODE Settings

Option	Description	Valid Values
RandomNops	Uses random nop-equivalent sequences instead of actual No-op instructions.	true or false

SIP Settings

SIP Settings below lists the Evasion Profile settings for SIP.

SIP Settings

Option	Description	Valid Values
CompactHeaders	Uses compact header names instead of full header names	true or false
EnvelopeType	Determines whether the To and From fields in the header are system generated or user-defined	**User-specified** – User defines To/From headers **System-generated** – System generates random To/From fields for the email header
From	Defines the From field in the email header if **EnvelopeType** is **User-specified**	String value (0 – 128 character length)
PadHeadersLineBreak	Pads headers with line breaks	true or false
ShuffleHeaders	Randomizes the order of headers in this protocol.	true or false
PadHeadersWhitespace	Pads headers with whitespace	true or false
RandomizeCase	Randomizes the case of data that is case sensitive	true or false
To	Defines the To field in the email header if **EnvelopeType** is **User-specifed**	String value (0 – 128 character length)

SMB Settings

SMB Settings on the next page lists the Evasion Profile settings for SMB.

SMB Settings

Option	Description	Valid Values
AuthenticationType	Determines whether the SMB server authentication is user-defined or system generated	**System-generated** – System generates the password and username **User-specified** – User defines the password and username
MaxReadSize	Defines the maximum read size for SMB requests	0 – 65,535
MaxWriteSize	Defines the maximum write size for SMB requests	0 – 65,535
Password	Defines the password used for POP3 connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)
RandomPipeOffset	Uses random file offsets when reading and writing to named pipes	true or false
Username	Defines the username used for POP3 connections if **AuthenticationType** is **Custom**	String value (0 – 32 character length)

SMTP Settings

SMTP Settings below lists the Evasion Profile settings for SMTP.

SMTP Settings

Option	Description	Valid Values
PadCommandWhitespace	Pads the whitespace between commands and arguments with space and tab characters	true or false
ShuffleHeaders	Randomizes the order of headers in this protocol	true or false

SSL Settings

SSL Settings on the facing page lists the Evasion Profile settings for SSL.

SSL Settings

Option	Description	Valid Values
Cipher and Hash	The cipher and hash suite to use for the encrypted communication.	DES CBC with SHA 3-DES CBC with SHA AES 128 with SHA AES 256 with SHA RC4 with SHA RC4 with MD5
ClientCertificateFile	Upload a certificate and select it from here. Client certificates will not be used if this option is not enabled.	Available file
ClientKeyFile	(OPTIONAL) Upload a private key file and select it from here. Client certificates will not be used if this option is not enabled.	Available file
DestPortOverride	Override the normal destination port for sessions using SSL	0 – 65,535
EnableOnAllHTTP	Override the normal destination port for sessions using SSL	true or false
EnableOnAllTCP	Enable SSL for every TCP session	true or false
ServerCertificateFile	Upload a certificate and select it from here. A default will be chosen if this option is not enabled.	Available file
ServerKeyFile	Upload a private key file and select it from here. A default will be chosen if this option is not enabled.	Available file

SUNRPC Settings

SUNRPC Settings on the next page lists the Evasion Profile settings for SUNRPC.

SUNRPC Settings

Option	Description	Valid Values
NullCredentialPadding	Pads the SunRPC credential block using a random set of group IDs	true or false
OneFragmentMultipleTCPSegmentsCount	Defines the number of TCP segments to use for an unfragmented RPC request	1 – 128
RPCFragmentTCPSegmentDistribution	Determines how RPC fragments are distributed across TCP segments	**AllFragmentsOneTCPSegment** – Sends all TCP SunRPC fragments in a single TCP segment **AllExceptLastFragmentOneTCPSegment** – Sends all TCP SunRPC fragments, except the last fragment, in a single TCP segment **OneFragmentPerTCPSegment** – Sends one TCP SunRPC fragment per TCP segment **OneFragmentMultipleTCPSegments** – Sends one TCP fragment in multiple TCP segments
TCPFragmentSize	Defines the maximum fragment size for TCP SunRPC requests	0 – 65,535

TCP Settings

TCP Settings on the facing page lists the Evasion Profile settings for TCP. All settings defined for TCP will affect the following protocols: IMAP4, SMTP, POP3, FTP, SMB, HTTP, and SIP.

TCP Settings

Option	Description	Valid Values
AcknowledgeAllSegments	Acknowledges all segments within the TCP window	true or false
DestinationPort	Defines the destination port to use if DestinationPortType is Static	0 – 65,535
DestinationPortType	Determines how TCP destination ports are selected	**Default** – Each Strike defines its own destination port **Static** – Strikes use a specified destination port **Random** – Strikes use a random destination port
DuplicateBadChecksum	Inserts duplicate segments with bad TCP checksums and invalid data	true or false
DuplicateBadReset	Inserts duplicate segments with the RST flag and bad sequence numbers	true or false
DuplicateBadSeq	Inserts duplicate segments with the RST flag and bad sequence numbers	true or false
DuplicateBadSyn	Inserts duplicate segments with bad sequence numbers	true or false
DuplicateLastSegment	Inserts a duplicate last segment for each write to the stream	true or false
DuplicateNullFlags	Inserts duplicate segments with null TCP flags and invalid data	true or false
MaxSegmentSize	Defines the maximum segment size for client-to-server transactions	0 – 1,460

Option	Description	Valid Values
SegmentOrder	Determines the order in which segments are sent	**Default** – Sends segments in the normal order **Reverse** – Reverses the order of all segments **Random** – Randomizes the order of all segments
SkipHandshake	Skips the three-way handshake for all connections	true or false
SourcePort	Defines the source port if **SourcePortType** is **Static**	0 – 65,535
SourcePortType	Determines how source ports are selected	**Default** – Each Strike defines its own source port **Static** – Strikes use a specified source port **Random** – Strikes use a random source port
AcknowledgeAllSegments	Causes the Security component to send a TCP acknowledgment packet to every TCP packet received.	true or false
SneakAckHandshake	Causes the Security component to perform a 4-way TCP handshake as opposed to the typical 3-way handshake.	true or false

UDP Settings

UDP Settings on the facing page lists the Evasion Profile settings for UDP. All settings defined for UDP will affect the following protocols: SIP.

UDP Settings

Option	Description	Valid Values
DestinationPort	Defines the destination port if **DestinationPortType** is **Static**	0 – 65,535
DestinationPortType	Determines how destination ports are selected	**Default** – Each Strike defines its own destination port **Static** – Strikes use a specified destination port **Random** – Strikes use a random destination port
SourcePort	Defines the source port if **SourcePortType** is **Static**	0 – 65,535
SourcePortType	Determines how source ports are selected	**Default** – Each Strike defines its own source port **Static** – Strikes use a specified source port **Random** – Strikes use a random source port

UNIX Settings

UNIX Settings below lists the Evasion Profile settings for UNIX.

UNIX Settings

Option	Description	Valid Values
PadCommandWhitespace	Pads the whitespace in Unix commands with space and tab characters	true or false
PadPathSlashes	Pads UNIX path names using space and tab characters	true or false

Editing Evasion Profiles

You can edit the settings for each Evasion Profile. These are the lowest level evasion options that will be used. Any evasion options set in an Evasion Profile through the Security component will take precedence over the options set for the Strike List.

To edit the evasion options for an Evasion Profile:

1. Select **Test > Open Test** from the Menu bar.
2. Select a security test from the list and click the Open button.
3. Select the Parameters tab.
4. Edit the Concurrent Strikes settings from the Parameter Label section (Optional). The Concurrent Strikes parameter allows you to choose between Single Strike and Default modes. Single Strike mode runs only one strike at a time, while Default mode runs up to five strikes simultaneously.
5. Under Parameter Label, select Evasion Profile.
6. From the Evasion Profile heading, click Edit. The available Evasion Profiles are displayed.
7. Select the profile you want to edit.
8. Click the check box of the parameters you want to change.
9. Make your changes to the selected parameters.
10. Click the **Apply Changes** button.
11. Click the **Save As** button.
12. Enter a name for the updated Evasion Profile.
13. Click OK.

Importing and Exporting a Strike List

You can import and export a Strike List from one system to another. To utilize this feature, all systems must be Release 1.2 or greater.

To import a Strike List:

1. Select **Managers > Import Strike List** from the Menu bar.
2. Enter a name for the Strike List in the **Strike List Name** field.
3. Click the **Browse** button.
4. Navigate to the location of the Strike List file (.bap file).
5. Click the **Open** button.
6. Select the **Allow Overwrite** check box to overwrite any existing Strike List of the same name.
7. Click the **Upload** button.

To export a Strike List:

1. Select **Managers > Strike Lists** from the Menu bar.
2. Select the Strike List you want to export from the **Strike List** list.
3. Select **Managers > Export Strike List** from the Menu bar.
4. Click the **Save** button.
5. Navigate to the location where you would like to store the Strike List file (.bap file).
6. Click the **Save** button.
7. Click the **Upload** button.

CHAPTER 14 Application Profiles and Super Flows

This section covers:

Overview

Combined, Application Profiles and Super Flows provide you with granular control over the application protocols that are on the wire by allowing you to define the individual flows for each protocol, and combine them in various ways for use in tests.

The process of creating application traffic flow is broken into several tasks. Before getting into those tasks, this section will help familiarize you with the terminology and overall work flow.

The table below defines the terms commonly used with application traffic flows.

Application Traffic Flow Terminology

Term	Definition
Application Profile	A container for the set of flow specifications (Super Flows) that Application Simulator uses to generate test traffic.
Super Flow	A container for all the individual flows and the specifications for the flows.
Flow	A flow establishes the protocol, server, and client.
Protocol Parameters	A set of parameters that is unique to each protocol. These parameters will typically define the ports and addressing for the server and client.
Actions	The events that will occur in a Super Flow. The actions that are available for each flow depends on the protocol on which the flow is based; each protocol has its own set of actions.
Action Parameters	A set of parameters that is unique to each action. Each action parameter allows you to control the data used within the action.
Weight	Determines the frequency in which the Super Flow will occur in the application traffic. Super Flows with higher weights will make up larger portion of the test traffic. The weight can be any value between 1 and 999,999,999.
Seed	The system uses the seed value to determine whether it generates static or dynamic application flows for the Super Flow. If you do not explicitly set a seed for the Super Flow, then the system will automatically randomize a seed for the Super Flow each time it is used. When you use a randomized seed, the system will dynamically generate new application traffic for the Super Flow. If you explicitly set the seed for the Super Flow, then the system will recreate the same application flows each time the Super Flow is run. Typically, you will want to use the same seed if you want to perform an apples-to-apples comparison between two devices; this enables you to determine how two devices handle the exact same stream of application traffic.
% Flows	The percentage of total flows that will be dedicated to the Super Flow. This value is affected by the weight that is assigned to the Super Flow; the larger the weight, in comparison to the other Super Flow weight assignments, the higher the % Flows will be.
% Bandwidth	The percentage of bandwidth consumed by the Super Flow. This value is affected by the weight that is assigned to the Super Flow; the larger the weight, in comparison to the other Super Flow weight assignments, the higher the % Bandwidth will be.
Sessions	The total number of sessions in the Super Flow.

Term	Definition
# Bytes	The total number of bytes in the Super Flow; this value will fluctuate for each Application Profile due to the randomization of action parameters within a flow.

Work Flow Overview

The first step is to create a Super Flow. A Super Flow contains all the individual flows that will be used for application traffic. The individual flows define the host definitions, protocol-type, and actions that the Super Flow will use. The individual flows can be customized even further with protocol parameters and action parameters.

Each flow added to a Super Flow essentially counts as a session. You can have up to 16 flows per Super Flow. This is important because BreakingPoint allows up to 7.5 million simultaneous sessions at a rate of 750,000 sessions per second on each 10 Gb slot, and it allows up to 5 million simultaneous sessions at a rate of 500,000 sessions per second on each 1 Gb blade.

Note: Aggregately, if you have two 10 Gb blades, you can generate up to 30 million simultaneous TCP sessions at a rate of 1.5 million sessions per second. If you have two 1 Gb blades, you can generate up to 10 million simultaneous TCP sessions at a rate of 1 million sessions per second.

After you have created your Super Flows, you can create your Application Profiles. The Application Profiles contain the Super Flows that the Application Simulator test component will use to generate application traffic.

The next section will illustrate how all the components of Application Profiles and Super Flows work together. It will provide you with an example of how you can set up a Super Flow based on HTTP.

Super Flow Example

For example, let's say you want to create an HTTP Super Flow that sends a request for an audio file.

The first thing you need to do is define your hosts. For an HTTP Super Flow, you will want to set up a DNS server, HTTP server, and a client as hosts. These hosts determine where requests, responses, queries, and flows are coming from.

After creating the hosts, you will need to define the flows. The flows consist of the protocol and the host definitions for the protocol. Since you are creating an HTTP Super Flow, you will want to create an HTTP flow and a DNS flow. For the DNS flow, you will set up the server-type as the DNS server, and for the HTTP flow, you will set up the server-type as the HTTP server. Both will use the client as a client-type.

Each flow has its own set of protocol parameters. For the most part, you can define the client and server port for all the protocols. Some protocols – such as FTP, HTTP, RTP, SIP, SMTP, and DNS – may provide parameters that are specific to the protocol. For example, HTTP will allow you to define the client profile (e.g., Internet Explorer 10.0) and server profile (e.g., Microsoft IIS 5.0).

Then finally, after the flows and the protocol parameters have been defined, you will need to create a sequence of actions for the Super Flow. These actions determine the events will that occur. So, this

example will create an action that sends a query to the DNS Server for the address of the HTTP Server. Then, it will create another action that sends a GET request for a specific URL file; in this case, audio.wma. Our final action will have the server send back with the file requested.

HTTP Super Flow Example

The following section provides step-by-step instructions to recreate the example above.

To set up the HTTP Super Flow:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Click the **Create New** button.
3. Enter a name for the Super Flow.
4. Click the **OK** button.
5. Click the **Manage Hosts** button.
6. Enter DNS Server in the **Name** field.

   Note: Hostnames must contain less than 255 characters, start with a letter, and consist of at least one label. Labels can contain 2-62 characters and use alphanumeric characters, dashes, or underscores; however, they cannot start or end with a dash or contain all numbers. Use the string '%n' to assign a unique number for each instance of the host.
7. Select **Server** from the **Interface** drop-down menu.
8. Enter dnsserver.bps.int in the **Nickname** field.
9. The name entered here will be viewable from the server-type and client-type drop-down menus. You can use the string '%n' to assign a unique number for each instance of the host. All tokens that are supported in token substitution can be used to create host nicknames.
10. Click the **Submit** button to enter the new host onto the Hosts table.
11. Click the **Add** button.
12. Enter HTTP Server in the **Host Name** field.

    Note: Hostnames must contain less than 255 characters, start with a letter, and consist of at least one label. Labels can contain 2-62 characters and use alphanumeric characters, dashes, or underscores; however, they cannot start or end with a dash or contain all numbers. Use the string '%n' to assign a unique number for each instance of the host.
13. Select **Server** from the **Interface** drop-down menu.
14. Enter httpserver.bps.int in the **Nickname** field.
15. The name entered here will be viewable from the server-type and client-type drop-down menus. You can use the string '%n' to assign a unique number for each instance of the host. All tokens that are supported in token substitution can be used to create host nicknames.
16. Click the **Submit** button to enter the new host onto the Hosts table.
17. Close the Manage Hosts window.
18. In the Flows region of the window, click **Add Flow**.
19. Select **Client** from the **Client** drop-down menu.
20. Select **DNS Server** from the **Server** drop-down menu.
21. Select **DNS** from the list of protocols.

22. Click the **OK** button.
23. Click **Add Flow** again.
24. Select **Client** from the **Client** drop-down menu.
25. Select **HTTP Server** from the **Server** drop-down menu.
26. Select **HTTP** from the list of protocols.
27. Click the **OK** button.
28. Select the **DNS** flow from the **Flows** list, located in the Flows region of the window.
29. In the Actions region of the window, click **Add Action**.
30. Select **Query(source client)** from the **Action** list.
31. Click the **OK** button.
32. In the Actions list, expand the newly-added Query action to show the Action Parameters.
33. Select the **Host** parameter, and double-click the **Value** field.
34. Select **HTTP Server** from the **Value** drop-down menu.
35. Click the **Save**.
36. In the Actions region of the window, click **Add Action**.
37. Select **Response(source server)** from the **Action** list.
38. Click the **OK** button.
39. In the Actions list, expand the Response action to show the Action Parameters.
40. Select the **Host** parameter, and double-click the **Value** field.
41. Select **HTTP Server** from the **Value** drop-down menu.
42. Click the **Save** button.
43. Select the DNS flow from the **Flows** list, located in the Flows region of the window.
44. In the Actions region of the window, click **Add Action**.
45. Select **GET(source client)** from the **Action** list.
46. Click the **OK** button.
47. In the Actions list, expand the GET action to show the Action Parameters.
48. Click the **Request path** check box (to enable this parameter).
49. Enter /audio.wma in the **Request path** Value field.
50. Click the **Save** button.
51. With HTTP still selected in the Flows region of the window, click **Add Action** (in the Actions region).
52. Select **Response 200 (OK)(source server)** from the **Action** list.
53. Click the **OK** button.
54. In the Actions list, expand the Response 200 (OK) action to show the Action Parameters.
55. Click the **Content-Type** check box (to enable this parameter).
56. Enter audio/x-ms-wma in the **Content Type** Value field.
57. Click **Save**.

Now that the Super Flow has been created, it will be available for you to add to any custom Application Profile.

Import and Export

The Application Profile editor and the Super Flow editor both provide an **Import** button and an **Export** button. These allow you to re-use Application Profiles and Super Flows across multiple test configurations.

Application Profiles

Application Profiles contain the set of flow specifications (Super Flows) that the Application Simulator test component will use to generate application traffic.

Each Super Flow will be assigned a weight that determines its frequency in the application traffic and a seed that determines whether the Super Flow generates static or dynamic application flows. Super Flows with higher weights will make up larger portion of the test traffic. For more information on Super Flow weight distribution, see the section Super Flow Weight Distribution below.

Super Flow Weight Distribution

Each Super Flow in an Application Profile will be assigned a weight that will determine the frequency at which Super Flow may be selected for the application traffic. BreakingPointuses a basic algorithm to determine how Super Flows are distributed. The weight of each Super Flow will be divided by the sum of all the weights in the App Profile, and then multiplied by 100. The resulting percentage represents the estimated portion of the application bytes that will be transmitted by that Super Flow.

If all Super Flows are weighted equally, longer flows will have a smaller effective weight than the shorter flows. This is due to the 'effective weight', which refers to the percentage of times a flow would need to occur for the weight distribution to equal the amount of bandwidth used by the protocol.

For example, if you have two Super Flows that are both weighted at 50%, but one flow uses twice as many bytes than the other flow, then the effective weight would be 33% for one flow and 66% for the other flow. Therefore, when you set the weight distributions for the Super Flows, you will need to take into account the number of application bytes that the Super Flow will transmit. When the system generates application traffic, it takes into account the number of transmitted application bytes and the amount of bandwidth utilized by the Super Flow.

The weighting system is most effective in tests that occur over a long period of time with a large number of flows (e.g., millions of flows). Short term tests may never reach the percentages allotted to each Super Flow because they do not provide the system with enough time to create a large number of flows or distribute the flows based on their weights.

For example, if you have a one minute Application Simulator test that uses 50% HTTP traffic and 50% BitTorrent traffic, the application traffic may consist of 100% HTTP traffic or 75% HTTP and 25% BitTorrent traffic. However, if this same test were run over a period of a day, the test traffic will be more likely to even out to 50% HTTP and 50% BitTorrent.

Weight Distribution Caveats

There are certain conditions under which the weighting system does not work exactly as expected.

- When the configured data rate is limited by some other factor (for instance, physical interface speed), the effects of rate limiting are nullified.
- When a Super Flow contains delays, it may not use the fully configured bandwidth. For example, it is difficult to make the RTP voice streams of a SIP call completely fill an allocated amount of bandwidth because they self-clamp to a smaller rate than allocated.
- Bandwidth limiting per-interface has similar issues compared to limiting in aggregate. Sometimes, a flow will not fill its allocated bandwidth in one direction or another, leading to an imbalance in a particular direction.

Super Flow Weight Distribution Example

To provide a better understanding of how the weighting system works, let's take a look at an example Application Profile called SuperFlow1. One Super Flow is called Flow1, which is based on HTTP and whose response sends 524,288 bytes of data. The second Super Flow is called Flow2, which is also based on HTTP, but sends twice as many bytes of data.

As previously mentioned, the longer a flow runs, the less its effective weight is; therefore, you will need to assign a higher weight to longer flows. So, to make both of these flows utilize the same amount of bandwidth, or have the same amount of weighting, you will need to set the weight for these two flows at 33% and 66%, respectively. Since one flow has twice the number of flows as the other, you will need to assign a weight that is 1.5 times the weight of the other.

Creating an Application Profile

The following section provides instructions for creating an Application Profile.

To create an Application Profile:

1. Select **Managers > Application Profile** from the BreakingPoint Control Center Menu bar.
2. Do one of the following:
   a. Click **Create New**, enter a name for the Application Profile, then click **OK**.
   b. Select an Application Profile from the **Application Profiles** list, click the **Save As** button, enter a name for the Application Profile, then click the **OK** button.
3. Click the **Add Superflow** button.
4. Select the Super Flow you wish to add from the **Super Flow Search Results** list. For information on Super Flows, see Super Flows on page 286.
5. Click the **Add (+)** button (located to the right of the Super Flow name).
6. You cannot have multiple instances of a Super Flow in an Application Profile.
7. Repeat steps 4 and 5 as many times as necessary to add all of the desired Super Flows to the Associated Super Flows list.
8. Click the **OK** button.
9. In the Associated Super Flows list, modify the **Weight** value for each Super Flow, as necessary. (To modify a **Weight** value, double-click it, type the new value, then press the Enter key.)
10. The weight will determine the frequency of the Super Flow occurring in application traffic. A higher weight will increase the chances of the Super Flow getting used.
11. Optionally, modify the **Seed** field for any of the Super Flows in the list.

12. To create static flows, set the seed to any arbitrary value between 1 and 999,999,999, or any 32 bit integer. If this field is not explicitly modified, or is set to '0', the system will auto-generate and randomize the seed for the Super Flow, thus creating new application flows each time the Super Flow is used.
13. Click the **Save** button once you are done.

Testing a URL Filtering Gateway

URL filtering helps control the URLs that enter and leave your network by allowing you to prevent access to unauthorized Web sites. It can also help protect your network from malware and other malicious traffic.

To test your URL filtering gateway, you can configure two Super Flows; one using a file with known authorized sites and one file with known unauthorized sites. You can set up your tests to report on whether all of the authorized sites were allowed and all of the unauthorized sites were blocked. To identify which URLs to search for, you can use the GetURIs action. With the GetURIs action, the URLs you want to use can be read sequentially from a file.

The GetURIs action is a compound action in that it carries both the client request and server response within a single action. The GetURIs action reads a list of URLs from a resource file and performs a series of GET/Response transactions, one pair for each URL.

To perform this action, you must have access to a file containing a list of URLs that you want to use. The file can be formatted as a list of URLs (one per line), or as a list of URLs along with their corresponding hostname.

Note: Be sure to adjust the Streams Per Super Flow setting to reflect the number of unique URIs you want to appear in your test.

Example 1

```
www.google.com/index.html
www.yahoo.com/music.mp3
www.breakingpoint.com/image.jpg
www.microsoft.com/manual.pdf
```

Example 2

```
www.google.com /index.html
www.yahoo.com /music.mp3
www.breakingpoint.com /image.jpg
www.microsoft.com /manual.pdf
```

After creating a file with the list of URLs to include in your test, use the following steps to upload the file and run your test.

To test a URL filtering gateway:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Select an HTTP-based Super Flow from the Super Flows list.
3. Click the **Create New** button.
4. Type a name for the new flow, then click **OK**.
5. In the Flows region of the window, click **Add Flow**.
6. Select **Client** from the **Client** drop-down menu.
7. Select **Server** from the **Server** drop-down menu.
8. Select **HTTP** from the list of protocols.
9. Click the **OK** button.
10. Click the Add Flow button and select the flow you just created.
11. In the Actions region of the window, click **Add Action**.
12. Select **GetURIs** (source **client**) from the **Action** list.
13. Click the **OK** button.
14. In the Actions list, expand the newly-added GetURIs action to show the Action Parameters.
15. Enable the **File with a list of URIs** parameter.
16. Double-click the **Value** field.
17. Select a filename (such as Welcome.html) from the drop-down list.
18. Click the **Save** button.
19. Click the **Upload** icon to upload the file.
20. Click the **Save** button.

Add the Super Flow you just created to an Application Profile and use that profile in your test.

Token Substitution

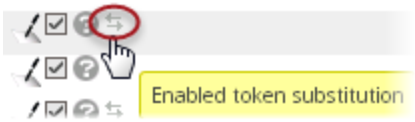
Token substitution allows application helpers to insert a token into the data stream. Before the Network Processor sends the data stream, it replaces the inserted token with content based on the token type and the optional format specifier.

Note: The maximum number of tokens per packet is 64. The maximum size of the token-replacing content is 512 bytes.

Entering Tokens in Fields

To enter a token in a field, you need to first enable token substitution (using the icon shown in Token Substitution Icon below). Once enabled, the field accepts tokens (or any value that you enter). If you disable token substitution for a field, it accepts only the values or data types defined for that field.

Token Substitution Icon



Token Format

Use the following format to create a token: ##[type](arguments)##.

Note: The presence of a "##foo(5)##" string should not cause an error if there are no "foo" token types. Such invalid expressions should be ignored and treated as literals. The same is true for tokens with invalid arguments. For example, ##int_c(5)## should be ignored, since ##int_c(5,5)## would be the correct method to specify one value for a range of integers.

Random-Character Tokens

This token produces a string composed of random characters. The token type determines the character set from which the string is created. The seed type, which is optional, determines the randomness of the string created by the token. A seed type of seed_random will produce a random string each time the token is used. A seed type of seed_flow will produce the same string each time it is used within a flow. Finally, a seed type of seed_superflow will produce the same string each time it is used within a Super Flow.

Syntax

Use the following syntax to create valid random-character tokens.

```
## type([seed type],[log type], min characters,max
characters)##
```

Example

```
## alpha(5,10)##; Specifies five to ten alpha
characters
```

The table below lists the valid types for creating random-character tokens.

Valid Types for Random-Character Tokens

Types	Meaning	Example
alpha	Alpha characters	A-Za-z
num	Numeric characters	0-9
alphanum	Alphanumeric characters	A-Za-z0-9
loweralpha	Lowercase alpha characters	a-z
upperalpha	Uppercase alpha characters	A-Z
punc	Punctuation characters	~‘!@#$%^&*()_-+={}|[]\;:’”,./<>?
text	All text characters	\x21-\x7e

Types	Meaning	Example
non_null	Non-zero 8-bit value	\x01-\xff
byte	Any 8-bit value	\x00-\xff
lowerhex	Lowercase human-readable hex	0-9a-f
upperhex	Uppercase human-readable hex	0-9A-F

Seed Type

The seed type is optional. The seed type allows you to seed randomly, or use the flow ID to seed. Using the flow ID to seed generates the same value whenever that token is used in a flow. If no value is specified for the seed type, it defaults to `seed_random`.

The table below lists valid seed types. A token may use one of the following seed types.

Note: These seed types are not exclusive to any particular type of token.

Valid Seed Types

Type	Meaning	Example
seed_ random	The value will be random every time the token is used.	alpha(seed_ random,8)
seed_flow	The value will be the same every time the token is used within a flow.	alpha(seed_flow,8)
seed_ superflow	The value will be the same every time the token is used within a Super Flow.	alpha(seed_ superflow,8)

Special Seed Types

There are two special seed types that can be used to generate sequential values rather than random ones. These special seed types are seeded per Super Flow/flow class. When these special seed types are used, a separate counter per Super Flow is added to the application profile.

The table below lists the available special seed types.

Special Seed Types

Type	Meaning	Example
seed_ sequential_flow	Increments the value independently for each flow in a Super Flow each time it is used.	user##num_range(seed_ sequential_flow, 1, 4)##
seed_ sequential_flow_ init	Initializes to a value in the range.	num_range (0,5, seed_ sequential_flow_init) 0

Type	Meaning	Example
seed_ sequential_flow_ incr	Increments to the next value in the range.	num_range (0,5, seed_ sequential_flow_incr) 1
seed_ sequential_ superflow	Increments the value in a Super Flow each time it is used.	user##num_range(seed_ sequential_superflow, 1, 4)##
seed_ sequential_ superflow_init	Initializes to a value in the range.	num_range (0,5, seed_ sequential_superflow_init) 0
seed_ sequeintial_ superflow_incr	Increments to the next value in the range.	num_range (0,5, seed_ sequential_superflow_init) 1

Random-Number Tokens

This token produces a random number where the token type determines the format of the number. The seed type, which is optional, determines the randomness of the number produced by the token. A seed type of `seed_random` will produce a random number each time the token is used. A seed type of `seed_flow` will produce the same number each time it is used within a flow. Finally, a seed type of `seed_superflow` will produce the same number each time it is used within a Super Flow. The minimum and maximum values must be specified.

Syntax

Use the following syntax to create valid random-number tokens. Note that both a minimum and a maximum value must be present. The seed type, however, is optional.

```
## type([seed type],min value,max value)##
```

Example

```
## num(1,9)##; Returns a string of random numbers, each having between 1 and 9 digits
```

The table below lists the valid types for creating random-number tokens.

Valid Types for Random-Number Tokens

Types	Meaning	Example
`int_C`	Unsigned character	`##int_C(0,255)##`
`int_c`	Character	`##int_c(0,127)##`

Types	Meaning	Example
`int_N Long`	Network (big-endian) byte order	`##int_N(1,2)##` ;produces \x00\x00\x00\x01 or \x00\x00\x00\x02
`int_n Short`	Network (big-endian) byte	`##int_n(1,2)##`
`int_V Long`	Little-endian	`##int_N(1,2)##` ;produces \x01\x00\x00\x00 or \x02\x00\x00\x00
int_v Short	Little-endian byte order	
int_Q	64-bit number (little endian, there is no big-endian 64-bit representation in Ruby.	
`int_v Short`	Little-endian	`##int_n(1,2)##` ;produces \x01\x00 or \x02\x00
`int_ Q, int_q`	64-bit number (little-endian, there is no big-endian 64-bit representation in Ruby)	`##int_q(1,2)##` ;produces \x01\x00\x00\x00\x00\x00\x00\x00 or \x02\x00\x00\x00\x00\x00\x00\x00
int_a	Human-readable ASCII	

Current Time Tokens

This token produces the current time. The token type determines the format in which the time is presented. A token type of `time` will produce the current time as a formatted string. A token type of `time_secs` will produce a string denoting the number of seconds since January 1, 1970.

Syntax

Use the following syntax to create valid tokens for the current time.

```
## type
()##
```

The table below lists the valid types for creating tokens for the current time.

Valid Types for Current Time Tokens

Types	Meaning	Example
`time`	Formatted time	`##time()##` ;produces Thu May 07 14:56:56 -0500 2009

Types	Meaning	Example
time_secs	Generates the amount of time that has elapsed (in seconds) since January 01, 1970 up to the current time	##time_secs()## ;produces 1256570659 (in ascii)
##time_formatted()##	Generates the amount of time that has elapsed (in seconds) since January 01, 1970 up to the current time. Same as time_secs, except it allows you to specify the format.	**##time_formatted (packed_be,4)##** ; produces seconds packed in 4 bytes in big endian format)

Host Information Tokens

This token provides information about the hosts in the Super Flow. The token type determines which information is displayed.

The flow id parameter, which is optional, specifies the flow from which the host information should be derived. This setting is valid only for the ip_addr and port tokens. The ID of a flow is shown in the Super Flow editor. If a flow ID is not specified, the flow in which the token occurs is assumed.

Syntax

Use the following syntax to create valid tokens for host information.

```
##type([log_type], format)## ## type(format,[flowid])##; provides host
information for a specific flow ID
```

Example

```
##ip_addr_cli
(text)##
```

The table below lists the valid types for creating valid tokens for host information.

Valid Types for Host Information Tokens

Types	Meaning	Example
ip_addr_cli	Current Client-side IP address	##ip_addr_cli(text,1)## ;will produce 1.0.0.2 (assuming client IP is 1.0.0.2)
ip_addr_srv	Current Server-side IP address	
port_cli	Client UDP or TCP port	##port_cli(text,1)## ;can produce 1234 (assuming the client port is 1234)
port_srv	Server UDP or TCP port	

Types	Meaning	Example
hostname_cli	Client hostname	##hostname_cli()## ;will produce: client123 (assuming the hostname is client123)
hostname_srv	Server hostname	

The table below lists the valid formats for creating valid tokens for IP address, port, and hostname information.

Valid Formats for IP Addresses, Ports, and Hostnames

Format	Meaning	Example
text	Plain text	1.2.3.4; or 8080
packed_le	Packed in little endian format. Long integers represent IP addresses, short integers represent port numbers.	
packed_be	Packed in big endian format	
human_be	ASCII encoding of big endian format	
text_ftp	Comma-delimited octets	"1,2,3,4", or "31,144"
with_len	(For hostnames only) A substituted hostname may be prepended with its binary length by specifying a token format of with_len.	\x05host1

Flowid Parameter

The flowid parameter is 1-based and is valid for port_cli/srv and ip_addr_cli/serv token types. However, the flowid parameter is not valid with hostname_cli or hostname_srv. A flowid of 0 indicates current flowid and is equivalent to not specifying a flowid.

Variable Tokens

The variable tokens provide a means of storing substring results from conditional requests so that those substrings may be used later in the Super Flow.

BreakingPoint supports conditional requests within both the Application Simulator and Client Simulator components. You can use a Perl Compatible Regular Expression (PCRE) within a conditional request to match data that occurs within the Super Flow.

The `bpsvar` token type produces the substring result. The substring number determines which substring the token will produce. The `bpsvar_len` token type produces the length of the substring result. Also, the substring number determines which substring length the token will produce.

Syntax

Use the following syntax to create valid tokens for variables. The maximum number of supported substrings is 10. The maximum length of each grabbed substring is 16.

```
## type(substring
number)##
```

The table below lists the valid types for creating valid tokens for variables.

Valid Types for Variables

Types	Meaning
`bpsvar`	An ASCII string containing most recent substring result from a Conditional Request
`bpsvar_len`	The length of the above bpsvar string

Valid Variable Formats

A valid variable format is any substring number between 0 and 9. This number identifies the substring from the most recent successful PCRE match of a Conditional Request.

Literal Expression

If the application needs to produce what would otherwise be a valid replacement token, the token expression should be encapsulated in a literal expression token. The result of this token is the literal string without substitution.

Syntax

Use the following syntax to create valid literal expression tokens.

```
## literal_bps
()##
```

Example

```
## literal_bps(##alpha(5,10)##)## ##literal_bps(##port_cli(text)##)##; produces
the literal string ##alpha(5,10)## and ##port_cli(text)## without substitution.
```

Increment Tokens

Increment tokens allows you to incrementally increase a unique value each time the value is encountered. You can specify a register value (0-9) and an initial value. The first time the token is

encountered, it will be replaced with the initialize value. The next time the token is encountered within the same Super Flow, it will be replaced with the previously substituted value plus one.

Syntax

Use the following syntax to create valid increment tokens.

```
##num_increment([seed_type],[log_
type],reg,init)##
```

Example

```
##num_increment(0,5)## // initializes register 0 to 5 ##num_increment(1,10)##
// initializes register 1 to 10
```

If the same register is used within the same packet, the values will increase.

It is important to note that the register size of the increment token is typically a 32-bit value, depending on the protocol. Once the maximum value has been reached, the counter will wrap back around to 0.

Range Tokens

Range tokens emit a number between the minimum and maximum. They are similar to increment tokens.

Syntax

Use the following syntax to create valid range tokens.

```
##num_range([seed_type],[log_
type],min,max)##
```

Substituted Length Tokens

Substituted length tokens are used to adjust a length field that cannot be predetermined by the application helper. There are certain tokens that the application helper cannot predetermine the payload size of the substituted data. This token provides a way to adjust for changes in payload size of subsequently substituted tokens.

This token will be substituted with the value of `base_length`, plus the difference in length of each subsequently substituted token.

The `max_tokens` is an optional parameter that specifies the scope of the calculated length. If `max_tokens` is specified, it will limit the number of subsequent tokens that are included in the calculation of the `subst_length` token value. If `max_tokens` is set to 0, or no value is specified for `max_tokens`, then the scope will go to the end of the packet.

Syntax

Use the following syntax to create valid substituted length tokens.

```
#unsubstantial(format,base_length,max_
tokens)##
```

The table below lists the valid formats for creating valid substituted length tokens.

Valid Types for Substituted Length Tokens

Format	Meaning
text	Plain text in decimal format
text_hex	Plain text in hexadecimal format
packed_le	Packed in 32-bit little endian format
packed_be	Packed in 32-bit big endian format
packed_le_16	Packed in 16-bit little endian format
packed_be_16	Packed in 16-bit big endian format
packed_8	Packed in 8-bit format
human_be	ASCII encoding of big endian format
bicc_bat	Packed for use in an ISUP APM message

Example

Before substitution:

```
subst_length(text,256,1) ##subst_length(text,256,1)## subst_length(text,256)
##subst_length(text,256)## subst_length(text,256) ##subst_length(text,256)##
```

After substitution:

```
subst_length(text,256,1) 233 subst_length(text,256) 233 subst_length(text,256)
256
```

Substituted Length Byte Token

Similar to the `subst_length` token, the `subst_length_byte` token uses the `max_bytes` parameter as the base length and also the scope in bytes. In other words, the `subst_length_byte` token will get replaced with a value representing the length of the payload specified in `max_bytes`. The substituted value will be adjusted for any tokens that exist within the range of `max_bytes`. If no tokens exist within

the `max_bytes` range, then the `subst_length_byte` token will simply be replaced with the value of `max_bytes`.

```
##subst_length_byte(format,max_
bytes)##
```

Use the following syntax to create valid substituted length byte tokens.

Substituted Length Words Token

Some token substitution fields denote the length of a value in 32-bit words. However, the substituted length token only returns the post-substitution length of a value in bytes. The `##subst_length_words (format,max_bytes)##` string will calculate the length of the substitution as 32-bit words.

```
##subst_length_words(packed_be_16, 25)##FooBar##literal_bps
(bps123)##
```

Use the following syntax to create valid 32-bit word formats for substituted length tokens.

Note: The string `##FooBar?##literal_bps(bps123)##` is 29 bytes in length, so the caller can subtract 4 up front to get the desired effect.

Length Padding Tokens

Length Padding tokens are similar in format to other substitution length tokens. The following token is located at the end of the string for which padding will be supplied:

- `##padding(alignment[,pad])##` - The text is aligned to the chosen alignment and padded with the given ASCII value for pad, which will default to 0 (the NUL byte).
- `exclude_pad_length` – This optional parameter produces no length at the beginning of the string.

The following example has embedded tokens. Note that the `subst_length_padding` string encloses the padding token:

```
{{{
##subst_length_padding(text,101,4)####subst_length_padding(packed_
le,41,2)##foo##literal_bps(bar)##baz##padding(19)####padding(32,0x40)##
00000000 20 33 32 13 00 00 00 66 6f 6f 62 61 72 | 32....foobar|
00000010 62 61 7a 00 00 00 00 00 00 00 00 00 00 40 40 40 |baz..........@@@|
00000020 40 40 40 40 40 40 |@@@@@@ |
}}}
```

The inner string is aligned to a nineteen-byte boundary and filled with NULs. The outer is aligned to a 32-byte boundary and filled with @'s.

Valid Formats

The table below lists the valid formats for creating valid length padding tokens.

Valid Types for Padding Tokens

Format	Meaning
text	Plain text in decimal format
text_hex	Plain text in hexadecimal format
packed_le	Packed in 32-bit little endian format
packed_be	Packed in 32-bit big endian format
packed_le_16	Packed in 16-bit little endian format
packed_be_16	Packed in 16-bit big endian format
packed_8	Packed in 8-bit format
human_be	ASCII encoding of big endian format
bicc_bat	Packed for use in an ISUP APM message

Dictionary Tokens

Dictionary elements are currently usable anywhere normal token elements are used. However, the source of the dictionary items itself currently has no way to be specified. The user interface or an app helper needs to define dictionary items before this will work. If you refer to a dictionary that is not defined, the word `nodict` will be substituted instead.

The table below lists the valid formats for creating valid dictionary tokens.

Valid Types for Dictionary Tokens

Format	Meaning
dict_flow	Emit a dictionary element from a flow-specific dictionary.
dict_ superflow	Emit a dictionary element from a Super Flow-specific dictionary
needle	Synonym for ##dict_superflow(log_unlimited, 0)##.
ip_checksum	Insert an IP checksum on the following bytes after the packet as if they were an IPv4 header.

Syntax

Use the following syntax to create valid dictionary tokens.

```
##type([seed type], [log type], [dictionary
id])##
```

Dictionary Tokens and Seed Types

When using flow dictionary tokens in conjunction with seed types, the seed types will impact the substituted result as follows:

- `seed_random`: The token will be substituted with a random value from the dictionary
- `seed_flow`: The token will be substituted with the same value from the dictionary every time that it is used within the same flow.
- `seed_superflow`: Not entirely useful in the case of flow dictionaries since a flow dictionary is assigned to only one flow. However, if you assign the same dictionary to two or more flows, this option would ensure that the substituted value is the same in those flows.
- `seed_sequential_flow`: Iterates through the values in the dictionary with each flow.
- `seed_sequential_flow_init`: Initializes to a value within a given range in a flow.
- `seed_sequential_flow_incr`: Increases to the next value in the range in a flow.
- `seed_sequential_superflow`: Iterates through the values in the dictionary with each Super Flow.
- `seed_goto_iteration`: Returns the current iteration of the goto loop in which the action that uses the token exists.
- `seed_sequential_superflow_init`: Initializes to a value within a given range in each Super Flow.
- `seed_sequential_superflow_incr`: Increases to the next value in a range within each Super Flow.

Note: The `seed_sequential_flow` and `seed_sequential_superflow` tokens will not increment the value when the `_init` type precedes them.

Add Flow Dictionary Action

The Add Flow Dictionary action adds a dictionary to the flow to which it belongs. With this action, you can upload a file representing the dictionary items, choose the delimiter for the file, and to assign an id to the dictionary. The id is referenced in a dictionary token parameter. The id mechanism allows you to add several dictionaries to a single flow as well as choose the dictionary from which a token will be replaced with its substituted value.

Note: The `log_type` parameter is optional. If unspecified, it defaults to `log_none`.

The table below lists the valid formats for the

Valid Types for the log_type Parameter

Format	Meaning
`log_none`	No values are logged (normal/default operation)

Format	Meaning
`log_limited`	The first 3000 occurrences of this value are logged to control (currently only used for lawful-intercept needles, limited to 5 per second)
`log_unlimited`	All occurrences of this value are logged to control (currently only used for lawful-intercept needles, limited to 5 per second)

`log_type` parameter.

Example

```
alpha(seed_random, log_unlimited,
8)
```

Note: While the log and seed parameters can be applied to any token, the seed parameter applies only to random tokens.

Add Number Dictionary

The Add Number Dictionary action produces a dictionary of telephone numbers. Super Flows that include this action can use tokens to extract a number from the dictionary.

Syntax

Use the following syntax to create a dictionary of telephone numbers within a flow.

```

```

Example

```
##dict_flow(seed_random,
0)##
```

Syntax

Use the following syntax to create a dictionary of telephone numbers within a Super Flow.

```
##dict_superflow
()##
```

Example

```
##dict_superflow(seed_superflow,
0)##
```

The table below lists the available settings for the Add Number Dictionary action.

Add Number Dictionary Actions

Setting	Description
Dictionary Type	The type of dictionary to create. When set to Flow, the resulting dictionary is available only to the flow from which this action was invoked. When set to Super Flow, the resulting dictionary may be used in any flow in the same Super Flow. Unless otherwise specified, the dictionary created will be of type Flow.
Dictionary ID	The identifier by which this dictionary is referenced.
Dictionary Source	Chooses the source of the dictionary. If the dictionary source is a file, the entries will be used as they exist in the file. If the dictionary source is set to Range, the Range Start and Range End fields will be used to produce the number dictionary. Otherwise, if a pattern is provided, special characters will be replaced as follows: `#\randomdigit, X\random alphanumeric`. In order to escape this pattern and enter a literal character, place the "\" in front of the character. For a literal "\", enter "\\".
Pattern	The pattern from which the telephone numbers are generated. The pattern may be up to 512 characters in length.
Pattern Count	The number of telephone numbers that are generated from the pattern. Valid values are between 1 and 100,000 (inclusive).
Range Start	The starting number in the range. Up to fifteen digits are allowed.
Range End	The ending number in the range. Up to fifteen digits are allowed.
File	The file from which the telephone numbers are generated.
Dictionary Delimiter Type	The type of delimiter between entries.
Dictionary Custom Delimiter	The custom delimiter for the dictionary file.
Format	The format in which the dictionary items will be encoded. A format of Ascii will result in the number appearing as it is entered. A format of E.164 will result in any non-numeric characters being stripped from the value and a plus sign (+) occurring before the number. A format of ISUP Address Signals will result in the value occurring in the format used by the ISUP Called Party Number, Calling Party Number and Generic Party Number information elements.

Mobility Session Tokens

To demonstrate whether traffic sent by a specific UE is being monitored, you can use the following token substitution strings to tag traffic in your mobility tests.

Note: The following tokens are designed specifically for mobility tests utilizing GTP tunnels. If these tokens are used when there is no GTP tunnel present, they will be substituted with an empty string.

The table below lists the valid formats for creating mobility session tokens.

Mobility Session Tokens

Format	Meaning
`##gtp_imei(format)##`	Substitute the IMEI of the current GTP session
`##gtp_imsi(format)##`	Substitute the IMSI of the current GTP session
`##gtp_msisdn(format)##`	Substitute the MSISDN of the current GTP session
`##gtp_teid(format)##`	Substitute the local TEID of the current bearer of the GTP session

Hash Algorithm

Hash tokens perform a cryptographic hash of the specified algorithm type on a substring that can consist of characters and other sub tokens.

The table below lists the valid formats for creating hash tokens.

Hash Tokens

Format	Meaning
##md5(format, "string ##token()##")##	md5 hash
##sha1(format, "string ##token()##")##	128-bit sha1 hash
##sha224(format, "string ##token()##")##	224-bit sha hash
##sha256(format, "string ##token()##")##	256-bit sha hash
##sha384(format, "string ##token()##")##	384-bit sha hash
##sha512(format, "string ##token()##")##	512-bit sha hash

Super Flows

When an Application Simulator test runs, it will first look at the Application Profile selected for the test. Then, it will look at the Super Flows that are contained within the Application Profile. Each Super Flow

contains the protocols that can be used to set up flows; server and client configurations; and the sequence of actions that will occur between the server and the client.

For example, you can use the HTTP and DNS protocols to create an HTTP Super Flow. The HTTP Super Flow would define the servers and clients that the protocols can use to simulate its requests and responses. So, in this case, you may want to create an HTTP server and a DNS server. For the HTTP server, you may want to set the HTTP version, server/client port, and client/server type. For the DNS server, you may want to assign the server/client port and the DNS Transaction ID. You can perform any of these customizations by modifying the protocol parameters.

After you have created your clients and servers, you will need to set up the actions for the Super Flow. These actions dictate the sequence of client requests and server responses, and the data that is sent during these sequences.

Creating a Super Flow

The following section provides instructions for creating a Super Flow. For more information on Super Flows, see the section Super Flows on the previous page.

To create a Super Flow:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Do one of the following:
   a. Click the **Create New** button, enter a name for the Super Flow, then click the **OK** button.
   b. Select a Super Flow from the **Super Flows** list, click the **Save As** button, enter a name for the Super Flow, then click the **OK** button.
3. Click the **Manage Hosts** button.
4. Create any additional hosts you may need. For more information on creating hosts, see the section Creating a Host below.
5. Close the Manage Hosts window when you are done creating hosts.
6. Add flows to the Super Flow. For more information on creating flows, see the section Creating a Flow on the next page.
7. Select a flow from the Flows list.
8. Create a sequence of actions based on the flow. For more information on adding actions, see the section Adding Actions to a Super Flow on page 289.
9. Click the **Save** button when you are done.

Creating a Host

At a minimum, you will need to define least two hosts: one for the server and one for the client. You may need additional hosts depending on the type of flows you are creating. For example, an HTTP Super Flow may need two server-types, one for the DNS server and one for the HTTP server, but an AOL Super Flow may only need one server and one client.

The Super Flow must have at least one host defined at all times. The system will display an error if you try to delete all the hosts. If this occurs, just click the **Close** button on the error message.

The following section provides instructions for creating a host. For more information on Super Flows, see the section Super Flows on page 286.

To create a host:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Click the **Manage Hosts** button.
3. Enter a host address in the **Name** field.
4. Click the **Interface** drop-down button, then select **Client** if the host will transmit data or **Server** if the host will receive data.
5. Enter a name for the host in the **Nickname** field.

   Note: You can use the string '%n' to assign a unique number for each instance of the host. All tokens that are supported in token substitution can be used to create host nicknames.
6. Click the **Submit** button.
7. Repeat steps 3-6 for each additional host.
8. Close the Manage Hosts window when you are done creating hosts.

Deleting a Host

Deleting a host will remove it and all flows that use the host. If you attempt to delete all hosts from the Super Flow, the system will display an error message. To resolve the error message, click the **Close** button.

To delete a host:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Click the **Manage Hosts** button.
3. Select the host in the **Hosts** list.
4. Click the **Delete** icon (to the right of the Host name).

Creating a Flow

A flow defines the protocol, servers, and clients available for the Super Flow. You create actions based on the flows that are available. The protocol parameters and flow actions that will be configurable for the flow will depend on the protocol that the flow is based on.

Note: Before creating a flow, you must have your hosts set up. For more information on creating hosts, see the Creating a Host on the previous page section.

Note: There can be up to 250 flows per Super Flow.

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. In the Flows region of the window, click **Add Flow**.
3. Click the **Client** drop-down button, then select a client from the **Client** drop-down menu.
4. Click the **Server** drop-down button, then select a server from the **Server** drop-down menu.
5. Select a protocol from the list of available protocols.

6. Click the **OK** button.
7. Select a flow in the Flows section, then expand the list of protocol parameters.
8. Enable or disable any parameter options you want to use.
9. Define the values for the protocol parameters that are enabled. For more information on protocol parameters, see the Protocol Parameters on the next page section.
10. Repeat steps 2 through 9 for each flow you'd like to add to the Super Flow.

Deleting a Flow

Deleting a flow will remove the flow and all its actions from the Super Flow.

Note: All actions based on the flow will also be removed from the Super Flow.

To delete a flow:

1. Select **Managers** > **Super Flows** from the BreakingPoint Control Center Menu bar.
2. Select a Super Flow from the **Select Super Flows** list.
3. Click the **Open** button.
4. Select a flow from the Flows list.
5. Click the **Delete** icon.
6. Click the **Yes** button when the Delete Flow confirmation window opens.

Adding Actions to a Super Flow

After you have set up the hosts and flows for the Super Flow, you can create a sequence of actions for the Super Flow. The following section provides instructions for adding actions to a Super Flow.

To add actions to a Super Flow:

1. Select **Managers** > **Super Flows** from the BreakingPoint Control Center Menu bar.
2. Select a Super Flow from the **Select Super Flows** list.
3. Select a flow from the Flows list.
4. The protocol used by the flow will determine the actions and action parameters that are available for you to configure.
5. In the Actions region of the window, click **Add Action**.
6. Either accept the sequence number provided (in the **Add new action at** field), or designate a different ordering. The first Action must be number 1 in the sequence, but you can change the order of the others.
7. Select an action from the list of available Actions, then click **OK**. For descriptions on actions, see Actions and Action Parameters on page 369.
8. To edit the action parameters: (optional)
   a. Select the Action in the Actions list, then expand it to list the parameters.
   b. Define or modify any of the parameters available for the action.
9. If you want to reference a valid file for server responses to URL requests, you must upload the file to the chassis. If uploaded files are supported by the protocol, you will see an option to upload files from the action parameters window.

10. Click the **Save** button when done.
11. Repeat steps 3 through 10 for each action you want to add to the Super Flow.
12. Click the **Save** button.

Deleting Actions from a Super Flow

Deleting an action will remove it from the Super Flow.

To delete actions to a Super Flow:

1. Select **Managers** > **Super Flows** from the BreakingPoint Control Center Menu bar.
2. Select a Super Flow from the **Select Super Flows** list.
3. Select the Action from the **Actions** list.
4. Click the **Delete** icon (located to the right of the Action name).
5. Click **Yes** when the Delete Action confirmation window opens.

Protocol Parameters

You can use protocol parameters to customize the clients and servers for a protocol. These protocol parameters allow you define the server and client port numbers and configure additional settings for certain protocols. Since the parameters vary for each protocol, you will need to have an understanding of each protocol's design and implementation to configure the server and client.

The following table lists the protocol parameters and their descriptions.

Protocol Parameters

Protocol	Protocol Parameters	Valid Values
AIM	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
AOL	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
	Locale	English French German Spanish
AppleJuice	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Bearer Independent Call Control	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
BGP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Client AS ID	N/A
	Server AS ID	N/A
BitTorrent Peer	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Encrypted	True or False
BitTorrent Tracker	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Encrypted	True or False

Protocol	Protocol Parameters	Valid Values
Chargen	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Server Profile	Ubuntu, Cygwin, or Custom
	Chargen Pattern	Up to 128 alphanumeric characters can be specified for the pattern of bytes sent to the client. This field is set only if **Server Profile** is set to **Custom**.
	Transport Protocol (Deprecated)	UDP or TCP

Protocol	Protocol Parameters	Valid Values
Citrix	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Seamless Channel Priority	0 – 3
	Window Channel Priority	0 – 3
	Disk IO Channel Priority	0 – 3
	Print Channel Priority	0 – 3
	Audio Channel Priority	0 – 3

Protocol	Protocol Parameters	Valid Values
Classic STUN	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Client External Network	N/A
	Client External CIDR Mask	0, 8, 16, or 24
	Server Network	N/A
	Server CIDR Mask	0, 8, 16, or 24
	Transaction ID	N/A
Daytime	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Transport Protocol (Deprecated)	UDP or TCP
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
DB2	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Database Name	Up to 256 alphanumeric and/or special characters can be used to define the database name.
DCE RPC	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Major Version	0 – 999
	Minor Version	0 – 999
	L5 Transport	Raw or SMB

Protocol	Protocol Parameters	Valid Values
DCE RPC Endpoint Mapping	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Major Version	0 – 999
	Minor Version	0 – 999
	L5 Transport	Raw or SMB
DCE RPC Exchange Directory	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Major Version	0 – 999
	Minor Version	0 – 999
	L5 Transport	Raw or SMB

Protocol	Protocol Parameters	Valid Values
DCE RPC MAPI	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Major Version	0 – 999
	Minor Version	0 – 999
	L5 Transport	Raw or SMB
DIAMETER	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Protocol Version	0 – 65,535
	Application ID	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Discard	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Transport Protocol (Deprecated)	UDP or TCP
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
DNS	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Ebay	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Echo	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
eDonkey	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Facebook	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Language	English, Deutsch, Espanol, and Francais
	API - Application Name	Up to 256 alphanumeric and/or special characters can be used to define the Application Name.
	API - Application URL	Up to 256 alphanumeric and/or special characters can be used to define the Application URL.
	API - Application ID	Up to 256 alphanumeric and/or special characters can be used to define the Application ID.
	API - Application Key	Up to 256 alphanumeric and/or special characters can be used to define the Application Key.
	API - Canvas Name	Up to 256 alphanumeric and/or special characters can be used to define the Canvas Name.
Finger	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
FIX	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Encoding Type	Tag = Value
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Application Version ID	Up to 50 alphanumeric and/or special characters can be used to define the ApplVerID field.
FIXT	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Encoding Type	Tag = Value
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Begin String	Up to 128 alphanumeric and/or special characters can be used to define the begin string.
	Username	Up to 128 alphanumeric and/or special characters can be used to define the username.
	Password	Up to 128 alphanumeric and/or special characters can be used to define the password.

Protocol	Protocol Parameters	Valid Values
FTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Server Profile	Internet Information Services 5.0, Pure FTPd, or Custom
	Server Name	Up to 128 alphanumeric and/or special characters can be used to define the custom Server Profile. This field is enabled only if **Server Profile** is set to **Custom**.
	Data Transfer Method	**Passive Mode (PASV)** – The FTP server opens a random port, sends the FTP's server's IP address and port number (broken into bytes) to the FTP client, and waits for a connection from the FTP client. The FTP client will bind to the source port to a random port that is greater than 1023. **Extended Passive Mode (EPSV)** – Same as passive mode (PSV), except that it transmits the port number (not broken into bytes), and the client connects to the same IP addresses it was originally connected to. **Active Mode (PORT)** – The FTP client opens a random port (> 1023), sends the FTP server the random port number on which it is listening on, and waits for a connection from the FTP server. The FTP server will bind the source port to port 20 once it initiates a connection to the FTP client. **Extended Active Mode (EPRT)** – Same as active mode, except it allows for the specification of an extended address. The extended address should define the network protocol and the network and transport addresses.
	Source Port (0=random)	0 – 65,535
	Server Data Port	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Gmail	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
GMX Webmail	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
GMX Webmail Attachment	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.

Protocol	Protocol Parameters	Valid Values
Gnutella 0.6	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Protocol Version	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
Gnutella-Leaf	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Gnutella-Ultrapeer	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Gopher	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
GTalk	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
GTalkUDP Helper	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
HTTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Client Profile	Android, Apple Safari, BlackBerry, Google Chrome, Internet Explorer 6.0, Internet Explorer 7.0, iPhone, Mozilla Firefox 2.0, Opera Mini/Mobile, Weighted mixes of all European mobile devices, Weighted mixes of all North American mobile devices, Windows Mobile, or BreakingPoint Default
	Server Profile	Microsoft IIS 5.0, Apache Server 2.0, or BreakingPoint Default
	HTTP Version Number	HTTP/1.0, HTTP/1.1
	Server Hostname	Up to 128 alphanumeric and/or special characters can be used to define the custom Server Profile. This field is enabled only if **Server Hostname** is set to **Custom**.
	Enable Cookie Persistence	on or off
	Number of random cookies	0 – 4,294,967,295 Default is 0
	Min length of each random cookie	0 – 4,294,967,295
	Max length of each random cookie	0 – 4,294,967,295
	Random cookie value persistence	on or off
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
HTTPS Simulated	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
H.225	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
H.225	H.245 Flow ID	
	Media Flow ID	
	Media Control Flow ID	
	Reference Value	
	Caller Username	

Protocol	Protocol Parameters	Valid Values
H.225	Caller Product ID	
	Caller Version ID	
	Caller Country Code	
	Caller Country Extension	
	Caller Manufacturer Code	
H.225	Callee Product ID	
	Callee Version ID	
	Callee Country Code	
	Callee Country Extension	
	Callee Manufacturer Code	
	Conference ID	
	Call Identifier	
	H.245 Connect Port	

Protocol	Protocol Parameters	Valid Values
H.225 RAS	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Call Signal Flow ID	
	Gatekeeper Identifier	
	Conference Identifier	
	Endpoint T.35 Country Code	
	Endpoint T.35 Manufacturer Code	
	Endpoint Product Identifier	
	Endpoint Product Version	
	Endpoint Identifier	
	H323 Identifier	
	Call Identifier	

Protocol	Protocol Parameters	Valid Values
H.245	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Media Flow ID	H.225 RAS:4, H.225:3, H.245:5, H.248:6, Hotmail Attachment:8, Hotmail:7, HTTP:1, HTTPS Simulated:2, IAX2:9, IDENT:10, IEC104:11, IMAPv4-Advanced:12, Informix:13, IPMI:14, IPP:15, or IRC:16
	Media Control Flow ID	H.225 RAS:4, H.225:3, H.245:5, H.248:6, Hotmail Attachment:8, Hotmail:7, HTTP:1, HTTPS Simulated:2, IAX2:9, IDENT:10, IEC104:11, IMAPv4-Advanced:12, Informix:13, IPMI:14, IPP:15, or IRC:16
H.248	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Audio Flow ID	H.225 RAS:4, H.225:3, H.245:5, H.248:6, Hotmail Attachment:8, Hotmail:7, HTTP:1, HTTPS Simulated:2, IAX2:9, IDENT:10, IEC104:11, IMAPv4-Advanced:12, Informix:13, IPMI:14, IPP:15, or IRC:16

Protocol	Protocol Parameters	Valid Values
Hotmail	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
Hotmail Attachment	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent.
IAX2	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
IDENT	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
IEC104	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Transport Protocol	UDP or TCP
IMAPv4-Advanced	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Server Name	Up to 256 alphanumeric and/or special characters can be used to define the IMAP server name.

Protocol	Protocol Parameters	Valid Values
Informix	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Database Name	Up to 128 alphanumeric and/or special characters can be used to define the database name.
Internet Relay Chat	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Local User Nickname	Up to 8 alphanumeric and/or special characters can be used to define the local user's nickname.
	Local Username	Up to 8 alphanumeric and/or special characters can be used to define the local user's name.
	Client Host Name	Up to 128 alphanumeric and/or special characters can be used to define the client's host name.
	IRC Node Name	Up to 128 alphanumeric and/or special characters can be used to define the IRC node name.
	IRC Central Node Name	Up to 128 alphanumeric and/or special characters can be used to define the IRC central node name.

Protocol	Protocol Parameters	Valid Values
IPMI	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
IPP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
ITCH	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Transport Protocol	moldUDP or moldUDP64
Jabber	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
LDAP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Transport Protocol (Deprecated)	UDP or TCP
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
LDP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
MMS MM1	Flow Mode	
	Source IP Address	
	Destination IP Address	
	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	SCTP Shared Connection Enabled	
	SCTP Shared Connection Client Offset	
	SCTP Shared Connection Server Offset	
	SCTP Shared Connection Master Only Enable	
MMS MM1	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Mobility Bearer ID	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Message Transport	

Protocol	Protocol Parameters	Valid Values
MMS MM1	Client HTTP User-Agent Header	
	Client HTTP x-upcalling-line-id Header	
	Client HTTP x-updevcap-charset Header	
	Client HTTP x-updevcap-iscolor Header	
	Client HTTP x-updevcapscreendepth Header	
	Client HTTP x-updevcapscreenpixels Header	
	Client HTTP x-upsubno Header	
	Client HTTP xwrap-profile Header	

Protocol	Protocol Parameters	Valid Values
Modbus	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
MSN-Dispatch	Client Port	0 – 65,535
	Server Port	0 – 65,535
MSN-Nexus	Client Port	0 – 65,535
	Server Port	0 – 65,535
MSN-Notification	Client Port	0 – 65,535
	Server Port	0 – 65,535

Protocol	Protocol Parameters	Valid Values
MSN-Passport	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	MSNP Version	
	Client Version	
	Passport	
	Password	
	Client GUID	
MSN-Switchboard	Client Port	0 – 65,535
	Server Port	0 – 65,535
	User Name	Up to 32 alphanumeric and/or special characters can be used to define the user name.
	User E-mail Address	Up to 256 alphanumeric and/or special characters can be used to define the user's e-mail address.
	Peer Name	Up to 32 alphanumeric and/or special characters can be used to define the peer's user name.
	Peer E-mail Address	Up to 256 alphanumeric and/or special characters can be used to define the peer's e-mail address.

Protocol	Protocol Parameters	Valid Values
MSSQL	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Multicast	Multicast: Role	
	Multicast: Group Address	
	Multicast Clients: SSM Source Address	
	Multicast Clients: Max Clients Per Subnet/VLAN	
	Multicast Clients: Max Measurable Leave Latency	
	Maximum Data Size	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF

Protocol	Protocol Parameters	Valid Values
MySQL	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
NetBIOS	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Netflix Streaming	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
NNTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
NTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Oracle	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Oscar	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Client Name	Up to 128 alphanumeric and/or special characters can be used to define the client's name.
	Screen Name	0 – 127 characters
	Client Profile	Internet Explorer 6.0, Internet Explorer 7.0, Mozilla Firefox 2.0, or Custom
OSCAR File Transfer	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
OUCH 4.1	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Outlook Web Access	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Server Hostname	

Protocol	Protocol Parameters	Valid Values
OWAMP Control	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
OWAMP Test	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Pandora	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	User Zipcode	
POP3-Advanced	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
PostgreSQL	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Protocol Version	0 – 65,535
	Username	
	Password	
	Database Name	
PPLive	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
PPTP	Flow Mode	

Protocol	Protocol Parameters	Valid Values
	Source IP Address	
	Destination IP Address	
	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	SCTP Shared Connection Enabled	
	SCTP Shared Connection Client Offset	
	SCTP Shared Connection Server Offset	
	SCTP Shared Connection Master Only Enable	
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Mobility Bearer ID	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
QQ	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Local User ID	
QQLive	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Quote of the Day	L4 Transport	Default*, TCP, or UDP *Default allows the protocol helper to automatically select the transport method. For QOTD, default is TCP.*
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
RADIUS Access	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Username	Up to 128 alphanumeric and/or special characters can be used to define the username.
	NAS IP Address	x.x.x.x, where x is a value between 0 –255
	NAS Port	0 – 256
	Framed IP Address	x.x.x.x, where x is a value between 0 –255
	Framed Netmask	x.x.x.x, where x is a value between 0 –255
	Calling Station ID	
	Called Station ID	
	Shared Secret	

Protocol	Protocol Parameters	Valid Values
RADIUS Accounting	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Username	Up to 128 alphanumeric and/or special characters can be used to define the username.
	NAS IP Address	x.x.x.x, where x is a value between 0 –255
	NAS Port	0 – 256
	Framed IP Address	x.x.x.x, where x is a value between 0 –255
	Framed Netmask	x.x.x.x, where x is a value between 0 –255
	Calling Station ID	
	Called Station ID	
	Shared Secret	

Protocol	Protocol Parameters	Valid Values
Raw	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	L7 Transport	
	TURN Channel Number	
	Maximum Data Size	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
RDP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Rexec	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	L7 Transport	
	TURN Channel Number	
	Maximum Data Size	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
RFB	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
RIPv1	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
RIPv2	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Rlogin	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
RPC Bind (Portmap)	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Credentials Flavor	
	AUTH_SYS UID	
	AUTH_SYS GID	
	Supplementary GIDs	
	Credentials Length	
	Verifier Flavor	
	Verifier Length	

Protocol	Protocol Parameters	Valid Values
RPC Mount	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Credentials Flavor	
	AUTH_SYS UID	
	AUTH_SYS GID	
	Supplementary GIDs	
	Credentials Length	
	Verifier Flavor	
	Verifier Length	

Protocol	Protocol Parameters	Valid Values
RPC NFS	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Credentials Flavor	
	AUTH_SYS UID	
	AUTH_SYS GID	
	Supplementary GIDs	
	Credentials Length	
	Verifier Flavor	
	Verifier Length	

Protocol	Protocol Parameters	Valid Values
Rsh	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Rsync	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
RTCP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
RTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
RTSP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Version	Up to 256 alphanumeric and/or special characters can be used to define the version used by RTSP.
	Media Flow ID	
	User Agent	
	Server	
	Default Request URL	

Protocol	Protocol Parameters	Valid Values
Rusers	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
S1AP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Client Port	0 – 65,535
	Server Port	0 – 65,535

Protocol	Protocol Parameters	Valid Values
SAP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
SCCP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
SIP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Media Flow ID	Rusers:1 S1AP:2 SAP:3 SCCP:4 SIP:5
	Call Flow ID	Rusers:1 S1AP:2 SAP:3 SCCP:4 SIP:5

Protocol	Protocol Parameters	Valid Values
	URL Format	IP Address Hostname
	Caller User-Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent for the client.
	Caller Name	Up to 128 alphanumeric and/or special characters can be used to define the caller's name.
	Caller Phone Number	String value consisting of up to 32 integers can be used to define the caller's phone number.
	Recipient User-Agent	Up to 128 alphanumeric and/or special characters can be used to define the user agent for the recipient.
	Recipient Name	Up to 128 alphanumeric and/or special characters can be used to define the recipient's name.
	Recipient Phone Number	String value consisting of up to 32 integers can be used to define the recipient's phone number.
Skype	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
Skype UDP Helper	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF

Protocol	Protocol Parameters	Valid Values
SMB	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Client Name	Up to 128 alphanumeric and/or special characters can be used to define the client's name.
	Client Native LM	Up to 128 alphanumeric and/or special characters can be used to define the client's native LM.
	Client Native OS	Up to 128 alphanumeric and/or special characters can be used to define the client's native OS.
	Destination Port (0=random)	0 – 65,535
	Server Name	Up to 128 alphanumeric and/or special characters can be used to define the server's name.
	Server Domain	Up to 128 alphanumeric and/or special characters can be used to define the server's domain name.
	Server GUID	Up to 128 alphanumeric and/or special characters can be used to define the server's GUID.
	Username	Up to 128 alphanumeric and/or special characters can be used to define the username.
	Password	Up to 128 alphanumeric and/or special characters can be used to define the password.

Protocol	Protocol Parameters	Valid Values
SMB File Stress	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
SMBv2	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Client Name	
	Destination Port (0=random)	0 – 65,535
	Server Name	
	Server Domain	
	Server GUID	
	Username	Up to 128 alphanumeric and/or special characters can be used to define the username.
	Password	

Protocol	Protocol Parameters	Valid Values
SMPP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	ESME System ID	
	MC System ID	
	MC Password	

Protocol	Protocol Parameters	Valid Values
SMTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Client Profile	Internet Explorer 6.0, Internet Explorer 7.0, Mozilla Firefox 2.0, or Custom
	Client Agent	Up to 128 alphanumeric and/or special characters can be used to define the custom Client Profile. This field is enabled only if **Client Profile** is set to **Custom**.
	Server Profile	Microsoft IIS 5.0, Apache Server 2.0, or Custom
	Server Name	Up to 128 alphanumeric and/or special characters can be used to define the custom Server Profile. This field is enabled only if **Server Hostname** is set to **Custom**.
	SMTP Domain	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
SNMPV1	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
SNMPV1 Traps	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
SNMPv2c	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
SNMPv2c Traps and Informs	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Soup TCP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
SQLMon	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
SSH	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
STUN2	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	L7 Transport	
	TURN Channel Number	
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
Sun RPC	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Credentials Flavor	
	AUTH_SYS UID	
	AUTH_SYS GID	
	Supplementary GIDs	
	Credentials Length	
	Verifier Flavor	0
	Verifier Length	

Protocol	Protocol Parameters	Valid Values
Sybase	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Logical Server Name	Up to 128 alphanumeric and/or special characters can be used to define the logical server name. This is the name of the Sybase server.
	Database Name	Up to 128 alphanumeric and/or special characters can be used to define the database name. This is the server name in which the authentication occur against.

Protocol	Protocol Parameters	Valid Values
Syslog	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Hostname	Up to 50 alphanumeric and/or special characters can be used to define the host name.
Telnet	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Client Profile	Silent Client Randomly Generated
	Server Profile	Microsoft IIS 5.0, Apache Server 2.0, or Custom

Protocol	Protocol Parameters	Valid Values
Time	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Transport Protocol (Deprecated)	TCP or UDP
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
TFTP	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Transport Protocol (Deprecated)	TCP or UDP
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Server Data Port (0=any)	

Protocol	Protocol Parameters	Valid Values
TR-069	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
TWAMP Control	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
TWAMP Test	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Twitter	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Twitter Username	Up to 256 alphanumeric and/or special characters can be used to define the user's login ID or username for their Twitter account.
	Twitter Password	Up to 256 alphanumeric and/or special characters can be used to define the password for the user's Twitter account.
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
uTorrent	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
VMware VMotion	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
WebDAV	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Webmail Orange	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	User Agent	

Protocol	Protocol Parameters	Valid Values
WHOIS	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
Winny	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Hostname	

Protocol	Protocol Parameters	Valid Values
World of Warcraft	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Protocol	Protocol Parameters	Valid Values
YIM (Yahoo Instant Messenger)	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535
	Protocol Version	
	Username	Up to 32 alphanumeric and/or special characters can be used to define the user's login ID or username.
	First Name	
	Last Name	
	Buddies	
	Online Buddies	
	Random Buddies	

Protocol	Protocol Parameters	Valid Values
Yahoo Mail	L4 Transport	Default, SCTP, SCTP over UDP, TCP, or UDP
	SCTP Tunneling Port (0=default)	0 – 65,535
	SCTP Checksum Type	ADLER32 or CRC32
	IPv4 TOS/DSCP	A hex value between 00 and FF
	IPv6 Traffic Class	A hex value between 00 and FF
	IPv6 Flow Label	A hex value between 00000 and FFFFF
	Source Port (0=random)	0 – 65,535
	Destination Port (0=random)	0 – 65,535

Actions and Action Parameters

You can use actions to set up server responses and client requests. Some actions have action parameters that enable you to configure the data within the responses and requests.

If you enable any of the action parameters, but leave their fields blank, the system will generate random data for that field. Additionally, the action parameters that have **(0 == random)** listed next to them will generate static data if it is set to any value other than 0.

Note: The actions and action parameters that are available to you depend on the flow (i.e., protocol) you have selected.

Transaction Flags

Most actions have an action parameter called **Transaction Flag** that enables you to set the first action to **Start** and the last action to **End**. All actions that neither denote the start nor end should have the **Transaction Flag** set to **Continue**. The application flow begins when the first Start Transaction packet is sent and ends when the End Transaction packet is sent; the period between when these two flags are sent comprise the application's response time.

Goto Action Request

The Goto action is a common or shared action among the protocols. With Goto, you can perform a group of actions multiple times without having to manually re-enter them multiple times. For example, you can create multiple HTTP request and response groups without having to re-enter the actions repeatedly.

This action has three basic parameters. The first parameter is the transaction flag which allows the Goto action to start, continue, or end the transaction. The second parameter is the number of times to perform the action. The third parameter is the action ID within the Super Flow that indicates where in the Super Flow to go to.

To create a Goto Action Request:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Select an existing Super Flow or create a new Super Flow.
3. Select a flow from the **Flows** area.
4. In the Actions region of the window, click **Add Action**.
5. Select **Goto** (source is **client**).
6. Expand the **Goto** action to show the action parameters.
7. Optionally, enable the **Transaction Flag** parameter, then select the desired value. Select **Start** to set the first action to start, or select End to set the last action to end.
8. Enable the **Goto Action** parameter, then select a value to indicate the target of the Goto operation. This will be either another Action from the action list, Goto (to create a loop), or end (to go to the end of the actions list).
9. Enable the **Iteration** parameter, then specify the number of times you want the selected action to loop.

   Note: Valid entries for the Iterations field include values ranging from 0 through 1,000,000,000. However, entering a value of zero (0) will result in an infinite loop. BreakingPoint recommends entering values between 1 and 1,000,000,000.

10. Click the **Save** button when done.

Alphanumeric, Special, and Integer Values

Some action parameters allow you to input string values; in these cases, you can either enter an integer value or a string value comprised of special and/or alphanumeric characters. This information will be noted in the **Valid Values** column of the action parameter.

If the field allows special characters, you can use the following characters: ! @ # $ % ^ * ( ) _ + = { } | \ : ; ' " , ? / .

Uploading Content to the System

You will need to upload content to BreakingPoint to have valid server responses to URL requests and to have valid files to transfer from one host to another. Some protocols – such as BitTorrent, FTP, and eDonkey – have an option that allows you to upload the data that will be used. The link to upload content will be located below the action parameter.

If you do not upload a file for the system to use, then the system will generate random data for the application payload portion of the flow.

AIM Action Parameters

The table below lists the actions and the action parameters available for AIM.

AIM Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host. This action is special in that it will automatically simulate a client/server DNS transaction, or query a proxy DNS server one-arm.	Host – The host that will be queried.	String data
		Retry Interval (ms) – The amount of time to wait for a response before failing or retrying.	1 – 1,000,000
		Retries until Failure – Number of retries to attempt before failing.	0 – 7
Client: Login	Login to the AIM6 Keyserver.	Transaction Flag	Start, Continue, End, or Start and End
Client: Post-Authenticate	Perform post-authentication tasks, such as setting availability, visibility, and retrieve buddy lists.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Buddies	0 – 1000
Client: Binding Request	Send a Binding Request to a STUN server. Deselect options to generate random values.	Transaction Flag	Start, Continue, End, or Start and End
		Change IP Flag	True or False
		Change IP Flag	True or False

Action	Description	Action Parameter	Valid Values
Server: Binding Response	Send a Binding Response to a STUN client. Deselect options to generate random values.	Transaction Flag	Start, Continue, End, or Start and End
		Mapped Address	IP address or token
		Mapped Port	0 – 65,535
		Source Address	IP address or token
		Source Port	0 – 65,535
		Changed Address	IP address or token
		Changed Port	0 – 65,535
Client: Join Chat	Join a chat session with other peers.	Transaction Flag	Start, Continue, End, or Start and End
Client: Chat	This command simulates an AIM conversation between a user and a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Peer Messages – The number of messages the peer will generate during the session.	0 – Random
		User Messages – The number of messages the user will generate during the session.	0 – Random

Action	Description	Action Parameter	Valid Values
Client: IM: Accept File Transfer	This command signals the acceptance of a file transfer from a peer to the user.	Transaction Flag	Start, Continue, End, or Start and End
Client: Receive File	Receive a file from a peer. In normal usage, an "Accept File Transfer" action from the AIM6-Switchboard AppSim should precede this action.	Transaction Flag	Start, Continue, End, or Start and End
		Client Username	String
		File Name	Any available file
		File Minsize	0 – 52,428,800
		File Maxsize	0 – 52,428,800
		File Data	String up to 128 bytes
Client: IM: Start File Transfer	This command signals the initialization a file transfer from the user to a peer.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Send File	Send a file to a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Client Username	String up to 16 bytes
		File Name	String up to 63 bytes
		File Minsize	0 – 52,428,800
		File Maxsize	0 – 52,428,800
		File Data	String up to 128 bytes

AOL Action Parameters

The table below lists the actions and the action parameters available for AOL.

AOL Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Send a random flow.	Transaction Flag	Start, Continue, End, or Start and End
Client: Resolve	Resolve the specified host.	Host – The host that will be queried.	String data
		Retry Interval (ms) – The amount of time to wait for a response before failing or retrying.	1 – 1,000,000
		Retries until Failure – Number of retries to attempt before failing.	0 – 7

Action	Description	Action Parameter	Valid Values
Client: Send Message	Send an e-mail message via AOL Web Mail.	Transaction Flag	Start, Continue, End, or Start and End
		From Address	String up to 128 bytes
		To Addresses	String up to 128 bytes
		CC Addresses	String up to 128 bytes
		BCC Addresses	String up to 128 bytes
		Subject	String up to 128 bytes
		Static Message Text File	Any available file
		Static Message Text	String up to 4096 bytes
		Language	Custom, English, French, German, Italian, Spanish
		Custom Dictionary	Any available file
		Message Wordcount Min	0 – 8192
		Message Wordcount Max	0 – 8192
		Keyword List	String up to 4096 bytes
		Attachment Filename	String
		Static Attachment	String
		Number of random attachments	Positive integer
		Random Attachment File Size	Positive integer
		Random Attachment Size Min	Positive integer
		Random Attachment Size Max	Positive integer

Action	Description	Action Parameter	Valid Values
		Attachment MIME Type	String up to 128 bytes
Client: Open Message	This action will simulate a client loading a message via the Web interface.	Transaction Flag	Start, Continue, End, or Start and End
		From Address	String up to 128 bytes
		To Addresses	String up to 128 bytes
		CC Addresses	String up to 128 bytes
		BCC Addresses	String up to 128 bytes
		Subject	String up to 128 bytes
		Static Message Text File	Any available file
		Static Message Text	String up to 4096 bytes
		Language	Custom, English, French, German, Italian, Spanish
		Custom Dictionary	Any available file
		Message Wordcount Min	0 – 8192
		Message Wordcount Max	0 – 8192
		Keyword List	String up to 4096 bytes
		Attachment Filename	String
		Static Attachment	String
		Random Attachment File Size	Positive integer
		Number of random attachments	Positive integer

Action	Description	Action Parameter	Valid Values
		Random Attachment Size Min	Positive integer
		Random Attachment Size Max	Positive integer
		Attachment MIME Type	String up to 128 bytes
Client: Open Attachment	This action will simulate a client loading an attachment via the Web interface.	Transaction Flag	Start, Continue, End, or Start and End
		Attachment Filename	String
		Static Attachment	String
		Random Attachment File Size	Positive integer
		Random Attachment Size Min	Positive integer
		Random Attachment Size Max	Positive integer
		Attachment MIME Type	String up to 128 bytes

AppleJuice Action Parameters

The table below lists the actions and action parameters for AppleJuice.

AppleJuice Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host.	Host – The host that will be queried.	String data
		Retry Interval (ms) – The amount of time to wait for a response before failing or retrying.	1 – 1,000,000
		Retries until Failure – Number of retries to attempt before failing.	0 – 7
Client: Client Setup Flow	Simulation of a Client Setup action.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Server: Server Setup Flow	Simulation of a Server Accepting a Client Setup action.	Transaction Flag	Start, Continue, End, or Start and End
Client: Client/Server Messages Flow	Simulation of a conversation between Client and Server.	Transaction Flag	Start, Continue, End, or Start and End

Border Gateway Protocol Action Parameters

The table below lists the actions and action parameters for the Border Gateway Protocol (BGP).

Border Gateway Protocol Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Server Certificate	A file in PEM format containing the server's certificate.
		Server Private Key	A file in PEM format containing the server's private key.
		Client Common Name	The client's common name (CN) as it appears in the client's certificate.
		Client CA Certificate	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the client's certificate.
		Client Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: Start TLS	Establish a TLS connection.	Enabled	true or false

Action	Description	Action Parameter	Valid Values
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.

Action	Description	Action Parameter	Valid Values
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: TLS Discard Encrypted Data	Allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Server: TLS Discard Encrypted Data	Updates bulk decryption statistics if set to true.	Count Discarded Data	true or false

Action	Description	Action Parameter	Valid Values
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	

Action	Description	Action Parameter	Valid Values
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

Action	Description	Action Parameter	Valid Values
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A

Action	Description	Action Parameter	Valid Values
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	
Client: OPEN	Simulates the BGP OPEN message. After a TCP connection is established, the first message sent by each side is an OPEN message.	Transaction Flag	Start, Continue, End, or Start and End
		My Autonomous System	0 – 1,024
		Hold Time	1 – 65,535
		BGP Identifier	A random identifier will be generated when configured with a value that is not an integer or a valid BreakingPoint token
		Multiprotocol Capability	IPv4 Unicast IPv6 Unicast

Action	Description	Action Parameter	Valid Values
Server: OPEN	Simulates the BGP OPEN message. After a TCP connection is established, the first message sent by each side is an OPEN message.	Transaction Flag	Start, Continue, End, or Start and End
		My Autonomous System	0 – 1,024
		Hold Time	1 – 65,535
		BGP Identifier	A random identifier will be generated when configured with a value that is not an integer or a valid BreakingPoint token
		Multiprotocol Capability	IPv4 Unicast IPv6 Unicast
Client: UPDATE	Simulates the BGP UPDATE message. UPDATE messages are used to transfer routing information between BGP peers.	Transaction Flag	Start, Continue, End, or Start and End
		Withdrawn Routes Type	None Manual Random
		Withdrawn Routes	Valid IP addresses
		Minimum Number of Random Withdrawn Routes	1 – 400
		Maximum Number of Random Withdrawn Routes	1 – 400

Action	Description	Action Parameter	Valid Values
		ORIGIN	EGP IGP INCOMPLETE
		AS_PATH Type	AS_SET AS_SEQUENCE
		AS_PATH	Valid IP addresses
		NEXT_HOP	Valid IP addresses
		MULTI_EXIT_ DISC	1 – 4,294,967,295
		LOCAL_PREF	1 – 4,294,967,295
		ATOMIC_ AGGREGATE	true or false
		AGGREGATOR IP	Valid IP addresses
		AGGREGATOR AS	1 – 65,535
		Include MP_ REACH_NLRI	true or false
		MP_REACH_ NLRI Address Family	IPv4 IPv6
		MP_REACH_ NLRI Next Hop	Valid IP addresses
		Use NLRI in MP_REACH_ NLRI	true or false
		Include MP_ UNREACH_ NLRI	true or false

Action	Description	Action Parameter	Valid Values
		MP_ UNREACH_ NLRI Address Family	IPv4 IPv6
		Use Withdrawn Routes in MP_ UNREACH_ NLRI	true or false
		Network Layer Reachability Info Type	None Manual Random
		Network Layer Reachability Info	Valid IP addresses
		Minimum Number of Random NLRI Prefixes	1 – 400
		Maximum Number of Random NLRI Prefixes	1 – 400
Server: UPDATE	Simulates the BGP UPDATE message. UPDATE messages are used to transfer routing information between BGP peers.	Transaction Flag	Start, Continue, End, or Start and End
		Withdrawn Routes Type	None Manual Random
		Withdrawn Routes	Valid IP addresses

Action	Description	Action Parameter	Valid Values
		Minimum Number of Random Withdrawn Routes	1 – 400
		Maximum Number of Random Withdrawn Routes	1 – 400
		ORIGIN	EGP IGP INCOMPLETE
		AS_PATH Type	AS_SET AS_SEQUENCE
		AS_PATH	Valid IP addresses
		NEXT_HOP	Valid IP addresses
		MULTI_EXIT_ DISC	1 – 4,294,967,295
		LOCAL_PREF	1 – 4,294,967,295
		ATOMIC_ AGGREGATE	true or false
		AGGREGATOR IP	Valid IP addresses
		AGGREGATOR AS	1 – 65,535
		Include MP_ REACH_NLRI	true or false
		MP_REACH_ NLRI Address Family	IPv4 IPv6

Action	Description	Action Parameter	Valid Values
		MP_REACH_ NLRI Next Hop	Valid IP addresses
		Use NLRI in MP_REACH_ NLRI	true or false
		Include MP_ UNREACH_ NLRI	true or false
		MP_ UNREACH_ NLRI Address Family	IPv4 IPv6
		Use Withdrawn Routes in MP_ UNREACH_ NLRI	true or false
		Network Layer Reachability Info Type	None Manual Random
		Network Layer Reachability Info	Valid IP addresses
		Minimum Number of Random NLRI Prefixes	1 – 400
Client: NOTIFICATION	Simulates the BGP NOTIFICATION message. The error subcodes are divided into groups by error code.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Error Code	Cease Finite State Machine Error Hold Timer Expired Message Header Error OPEN Message Error
		Message Header Error Subcode	Bad Message Length Bad Message Type Connection Not Synchronized
		OPEN Message Error Subcode	Bad BGP Identifier Bad Peer AS Deprecated Unacceptable Hold Time Unsupported Optional Parameter Unsupported Version Number

Action	Description	Action Parameter	Valid Values
		UPDATE Message Error Subcode	Attribute Flags Error Attribute Length Error Deprecated Invalid Network Field Invalid NEXT-HOP Attribute Invalid ORIGIN Attribute Malformed AS_PATH Malformed Attribute List Missing Well-Known Attribute Optional Attribute Error Unrecognized Well-Known Attribute
Server: NOTIFICATION	Simulates the BGP NOTIFICATION message. The error subcodes are divided into groups by error code.	Transaction Flag	Start, Continue, End, or Start and End
		Error Code	Cease Finite State Machine Error Hold Timer Expired Message Header Error OPEN Message Error

Action	Description	Action Parameter	Valid Values
		Message Header Error Subcode	Bad Message Length Bad Message Type Connection Not Synchronized
		OPEN Message Error Subcode	Bad BGP Identifier Bad Peer AS Deprecated Unacceptable Hold Time Unsupported Optional Parameter Unsupported Version Number

Action	Description	Action Parameter	Valid Values
		UPDATE Message Error Subcode	Attribute Flags Error Attribute Length Error Deprecated Invalid Network Field Invalid NEXT-HOP Attribute Invalid ORIGIN Attribute Malformed AS_PATH Malformed Attribute List Missing Well-Known Attribute Optional Attribute Error Unrecognized Well-Known Attribute
Client: KEEPALIVE	Simulates the BGP KEEPALIVE message.	Transaction Flag	Start, Continue, End, or Start and End
Server: KEEPALIVE	Simulates the BGP KEEPALIVE message.	Transaction Flag	Start, Continue, End, or Start and End

BitTorrent Peer Action Parameters

The table below lists the actions and action parameters for BitTorrent Peer.

If you use the **Client: Download File** action, you must dedicate an entire Super Flow to it. No other actions can be contained within the Super Flow besides the **Client: Download File** action.

BitTorrent Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Download Chunk	Downloads single chunk from a peer.	Chunk size (bytes)	1 – 1,048,576
		Response Data File	Use the **Import Specify Chunk Data** link to upload the chunk data for which the client will download, and then select the filename from the **Specify Chunk Data** drop-down menu.
Client: Download File	Downloads a complete file from six peers.	Chunk Size (bytes)	1 – 5,242,880
		Random File Min Size (bytes)	1 – 4,194,304
		Random File Max Size (bytes)	1 – 4,194,304
		Specify File Data	Use the **Import Specify File Data** link to upload the file data for which the client will download, and then select the filename from the **Specify File Data** drop-down menu.

BitTorrent Tracker Action Parameters

The table below lists the actions and action parameters for BitTorrent Tracker.

BitTorrent Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Track Register	Registers a client system with a central tracker and downloads a list of peers.	None	N/A

Chargen Action Parameters

The table below lists the actions and the action parameters available for Chargen.

Chargen Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Raw Message	Sends a file or string directly without any modification. If both are specified, the file is concatenated to the string.	Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message	Sends a file or string directly without any modification. If both are specified, the file is concatenated to the string.	Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.

Action	Description	Action Parameter	Valid Values
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	

Action	Description	Action Parameter	Valid Values
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file	Transaction Flag	Start, Continue, End, or Start and End
		File to verify	Resource file to be used.
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	
Client: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST

Action	Description	Action Parameter	Valid Values
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	

Action	Description	Action Parameter	Valid Values
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	
Server: Generate Characters	Sets the number of bytes to transmit per session and the seed value. The **Seed Value** sets the starting character of the chargen pattern; if the seed is set to any integer other than 0, the same byte pattern will be used each time the flow is used.	Tx Bytes	0 – 65,532
		Seed Value	0 – 999

Citrix Action Parameters

The table below lists the actions and the action parameters available for the Citrix protocol.

Citrix Action Parameters

Action	Description	Action Parameters	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite

Action	Description	Action Parameters	Valid Values
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Server Certificate	A file in PEM format containing the server's certificate.
		Server Private Key	A file in PEM format containing the server's private key.
		Client Common Name	The client's common name (CN) as it appears in the client's certificate.
		Client CA Certificate	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the client's certificate.

Action	Description	Action Parameters	Valid Values
		Client Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295

Action	Description	Action Parameters	Valid Values
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false

Action	Description	Action Parameters	Valid Values
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: TLS Discard Encrypted Data	Allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Server: TLS Discard Encrypted Data	Updates bulk decryption statistics if set to true.	Count Discarded Data	true or false
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.

Action	Description	Action Parameters	Valid Values
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9

Action	Description	Action Parameters	Valid Values
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

Action	Description	Action Parameters	Valid Values
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A

Action	Description	Action Parameters	Valid Values
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	
Client: Seamless Application	Simulates a Seamless Application session.	Transaction Flag	Start, Continue, End, or Start and End
Client: Window Application	Simulates a Window Application session.	Transaction Flag	Start, Continue, End, or Start and End
Server: Disk IO	Simulates a remote disk IO operation.	Transaction Flag	Start, Continue, End, or Start and End
Server: Print Spooler	Simulates a remote printing operation.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameters	Valid Values
Server: Audio Event	Simulates a remote audio operation.	Transaction Flag	Start, Continue, End, or Start and End

Daytime Action Parameters

The table below lists the actions and the action parameters available for the Daytime protocol.

Daytime Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Send Time	Sets the date and time to send to the client.	Date/Time	A date and time in the format of YYYY-MM-DD HH:MM:SS; YYYY can be replaced with a value between 1970 and 2035.

DB2 Action Parameters

The table below lists the actions and action parameters available for the DB2 protocol.

DB2 Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to the DB2 database.	Database Username	Up to 255 alphanumeric and/or special characters can be used to define the database username.
		Database Password	Up to 255 alphanumeric and/or special characters can be used to define the database password.
		Client Login Username	Up to 255 alphanumeric and/or special characters can be used to define the login username for the client.
Client: Login	Simulates a login to the DB2 database.	Database Hostname	Up to 255 alphanumeric and/or special characters can be used to define the host name for the database.

Action	Description	Action Parameter	Valid Values
Client: SQL Query	Simulates an SQL query and response. The values entered for **Columns** and **Rows** will determine the number of columns and rows the query will return. If **Column Names** are specified, then they need to be entered as comma-delimited values.	SQL Query	Alphanumeric and/or special characters can be used to define the SQL query.
		Columns	0 – 4,294,967,295
		Rows	0 – 4,294,967,295
		Column Names	Alphanumeric and/or special characters can be used to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.

DCE RPC Action Parameters

The table below lists the actions and the action parameters available for DCE RPC.

DCE RPC Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Request	Performs a Remote Procedure Call request from the client. The **Client:Request** action utilizes the corresponding action parameters to populate the standard RPC protocol client request.	Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Alloc Hint	0 – 65,535
		Context ID	0 – 65,535
		Opnum	0 – 65,535
		Payload	Use alphanumeric and/or special characters to define the message payload.

Action	Description	Action Parameter	Valid Values
Server: Response	Performs a Remote Procedure Call response from the server. The **Server:Response** action utilizes the corresponding action parameters to populate the standard RPC protocol server response.	Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Alloc Hint	0 – 65,535
		Context ID	0 – 65,535
		Cancel Count	0 – 255
		Payload	Use alphanumeric and/or special characters to define the message payload.
Server: Fault	Defines the Remote Procedure call fault. The Server	Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Alloc Hint	0 – 65,535
		Context ID	0 – 65,535
		Cancel Count	0 – 255
		Status	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
Client: Bind		Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Maximum Transmitted Fragments	0 – 65,535
		Maximum Received Fragments	0 – 65,535
		Assoc Group	0 – 65,535
		Context ID	0 – 65,535
		Num Trans Items	0 – 255
		Interface	Use alphanumeric and/or special characters to define the interface UUID.
		Interface Major Version	0 – 255
		Interface Minor Version	0 – 255
		Syntax	Use alphanumeric and/or special characters to define the syntax UUID.
		Syntax Version	Use alphanumeric and/or special characters to define the syntax version.

Action	Description	Action Parameter	Valid Values
Server: Bind ACK	Creates the Remote Procedure Call bind acknowledgment.	Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Maximum Transmitted Fragments	0 – 65,535
		Maximum Received Fragments	0 – 65,535
		Assoc Group	0 – 65,535
		Secondary Address	
		Acceptance	
		Syntax	Use alphanumeric and/or special characters to define the syntax UUID.
		Syntax Version	Use alphanumeric and/or special characters to define the syntax version.
		Challenge	True or False
		Challenge: Windows Domain	
		Challenge: DNS Domain	
		Windows Name	
		DNS Name	
		Challenge String	

Action	Description	Action Parameter	Valid Values
Client: Auth3		Transaction Flag	Start, Continue, End, or Start and End
		Call ID	0 – 4,294,967,295
		Domain Name	Use alphanumeric and/or special characters can be used to define the domain name.
		User Name	Use alphanumeric and/or special characters can be used to define the user name.
		Host Name	Use alphanumeric and/or special characters can be used to define the host name.
Server: Send Flow (Deprecated)	Sends a DCE/RPC flow from the server to the client.	None	N/A

Discard Action Parameters

The table below lists the actions and action parameters available for the Discard protocol.

Discard Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Send	Defines the data that will be discarded.	Data	Alphanumeric and/or special characters can be used to define the data that will be discarded. There is a 1,024 character limit.

DNS Action Parameters

The table below lists the actions and the action parameters available for DNS.

DNS Action Parameters

Action	Description	Action Paramete	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Raw Message	Reads the contents of a file then sends the file	Transaction Flag	Start, Continue, End, or Start and End
		Filename	Resource file stored on the BPS box
Server: Raw Message	Reads the contents of a file then sends the file	Transaction Flag	Start, Continue, End, or Start and End
		Filename	Resource file stored on the BPS box

Action	Description	Action Paramete	Valid Values
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Predefined	Select to match a predefined pattern.
		Available Action	Delay, Raw Message, Verify Rows, Goto, Close, Fail, Login, Login Request, Authenticate, Use Database, Query, Quit
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	
Client: Query	Queries the DNS Server for the host	Transaction Flag	Available Actions
		Host	Any host configured in the Super Flow
		Query Type	A or PTR

Action	Description	Action Paramete	Valid Values
Client: Resolve	Sends the DNS query	Host	Any host
		Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
Server: Resolve	Sends the DNS response	Host	Any host
		Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
Client: Fail	Causes the flow to fail		
Server: Fail	Causes the flow to fail		
Server: Response	Responds with the IP address for the host.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 4,294,967,295
		Host	Any host
		Type	A or PTR
		DNS TTL	0 – 4,294,967,295 *default - 86,400
		Response Time (ms)	0 – 1,000

Ebay Action Parameters

The table below lists the actions and the action parameters available for Ebay.

Ebay Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host.	Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
		Use Response	true or false
		Host	Any host
Client: Get eBay.com	Performs a GET request for eBay.com	Transaction Flag	Start, Continue, End, or Start and End
Server: Send homepage	The server response to a user request.	Transaction Flag	Start, Continue, End, or Start and End
Client: Load Signin Page	Client loads login page.	Transaction Flag	Start, Continue, End, or Start and End
Server: Send Signin Page	The server sends the signin page.	Transaction Flag	Start, Continue, End, or Start and End
Client: Send login credentials	Sends the username and password to the authentication server via TLS.	Transaction Flag	Start, Continue, End, or Start and End
		User ID	eBay user account to use
		Password	Use alphanumeric and/or special characters to provide the password for the User Request message.

Action	Description	Action Parameter	Valid Values
Server: Login Response	The server response to a user login.	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Search	Client sends a search term, and/or a category.	Transaction Flag	Start, Continue, End, or Start and End
		Query	N/A
		Search Category	Any available category
Server: Search Results	The server response to a user search request. Returns a random number of results between 1 and 50.	Transaction Flag	Start, Continue, End, or Start and End
Client: View Item	Client request for an item listing.	Transaction Flag	Start, Continue, End, or Start and End
Server: Send Item Listing	The server response to a user request for a specific item listing.	Transaction Flag	Start, Continue, End, or Start and End
Client: My eBay	Client request for the My eBay page.	Transaction Flag	Start, Continue, End, or Start and End
Server: Send My eBay	The server response to a user request for the My eBay page.	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Signout	Client request to log out of the eBay service.	Transaction Flag	Start, Continue, End, or Start and End
Server: Signout Confirmation	The server response to a user request to log off.	Transaction Flag	Start, Continue, End, or Start and End

Echo Action Parameters

The table below lists the actions and the action parameters available for Echo.

Echo Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Send	Sends an Echo flow from the client to the server.	Data	Alphanumeric and/or special characters can be used to define the data that will be echoed back from the server.

eDonkey Action Parameters

The table below lists the actions and action parameters available for eDonkey.

eDonkey Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Transfer	Creates a peer connection and transfers a file that is broken into chunks.	Simulation File Data	Use the **Import Simulation File Data** link to upload the data for which the client will transfer, and then select the filename from the **Simulation File Data** drop-down menu.

Facebook Action Parameters

The table below lists the actions and action parameters available for the Facebook protocol.

Facebook Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Get Initial Page	Gets the original Facebook page for logging in.	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Original Page	Returns the original page.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Authenticate	Authenticates to Facebook server from the Web interface.	Transaction Flag	Start, Continue, End, or Start and End
		Account User (email address)	Use alphanumeric and/or special characters to define the user's email address.
		Password	Use alphanumeric and/or special characters to define the user's password.
Server: Facebook Authentication Success	Server response for authentication.	Transaction Flag	Start, Continue, End, or Start and End
Client: Facebook News Feed Page	Client request for news feed.	Transaction Flag	Start, Continue, End, or Start and End
Server: Facebook News Feed Page	Server response with news feed.	Transaction Flag	Start, Continue, End, or Start and End
Client: Update Status	Updates the authenticated user's status.	Transaction Flag	Start, Continue, End, or Start and End
		Status	Use alphanumeric and/or special characters to define the user's status.
Client: Facebook Chat Message		Transaction Flag	Start, Continue, End, or Start and End
		Friend Profile ID	N/A
		Message	Use alphanumeric and/or special characters to define the message sent from the user to the peer.

Action	Description	Action Parameter	Valid Values
Server: Facebook Chat Response	Server response to a user chat message.	Transaction Flag	Start, Continue, End, or Start and End
		User Name	Use alphanumeric and/or special characters to define the user's name.
		Profile ID	N/A
		Friend Name	Use alphanumeric and/or special characters to define the friend's name.
		Friend Profile ID	N/A
		Message	Use alphanumeric and/or special characters to define the response sent from the server.
Client: Facebook Send Message	Send a message via Facebook.	Transaction Flag	Start, Continue, End, or Start and End
		Friend Profile ID	String up to 50 bytes
		Subject	Use alphanumeric and/or special characters to define the subject of the message.
		Message	Use alphanumeric and/or special characters to define the message sent from the user to the peer.

Action	Description	Action Parameter	Valid Values
Server: Facebook Message Response	Server response to a user message.	Transaction Flag	Start, Continue, End, or Start and End
Client: Facebook Logout	Sends logout request.	Transaction Flag	Start, Continue, End, or Start and End
Server: Facebook Logout Response	Server response to a user request to logout.	Transaction Flag	Start, Continue, End, or Start and End

Finger Action Parameters

The table below lists the actions and action parameters available for the Finger protocol.

Finger Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Request	Sends an empty request to the server, a request that contains a username, or a request that contains a username and the server to which the finger request will be forwarded. Set **Send /W** to **True** to generate a random server	Transaction Flag	Start, Continue, End, or Start and End
		Username	Alphanumeric and/or special characters can be used to define the username in the request.
		Send /W	True or False

Action	Description	Action Parameter	Valid Values
Server: Response	Sends a response to the client that contains the user's information. This information includes the number of users logged into the system (user count) and the contents of the resource file that will be used as the .plan file for the users. Additionally, you can set how the whether the user is valid or invalid by setting **User Exists** to **True** or **False**. If **True**, then server will return the information for the user; if **False**, the server will respond with '<username>: no such user'.	Transaction Flag	Start, Continue, End, or Start and End
		User Count	0 – 4,294,967,295
		Username	Alphanumeric and/or special characters can be used to define the username in the response.
		.plan Resource File	Use the **Import** link to upload the data in which the client will download, and then select the filename from the **.plan Resource File** drop-down menu.
		User Exists	True or False

FIX Action Parameters

The table below lists the actions and action parameters available for the FIX protocol.

FIX Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Business Message Reject	Sets up the message that the client sends to the server indicating that it has rejected a message that it had previously received.	Transaction Flag	Start, Continue, End, or Start and End
		Referenced Sequence Number	0 – 4,294,967,295
		Referenced Message Type	Use alphanumeric and/or special characters to provide the message type of the referenced FIX message.
		Referenced Business Reject ID	0 – 4,294,967,295
		Business Reject Reason	0 – 4,294,967,295
		Text	Use alphanumeric and/or special characters to provide a reason for rejecting the message.

Action	Description	Action Parameter	Valid Values
Server: Business Message Reject Message	Sets up the message that the server sends to the client indicating that it has rejected a message that it had previously received.	Transaction Flag	Start, Continue, End, or Start and End
		Referenced Sequence Number	0 – 4,294,967,295
		Referenced Message Type	Use alphanumeric and/or special characters to provide the message type of the FIX message being referenced.
		Referenced Business Reject ID	0 – 4,294,967,295
		Business Reject Reason	0 – 4,294,967,295
		Text	Use alphanumeric and/or special characters to provide a reason for rejecting the message.
Client: Network (Counterparty System) Status Request Message	Requests a Network (counterparty system) Status Response message.	Transaction Flag	Start, Continue, End, or Start and End
		Network Request Type	Use alphanumeric and/or special characters to provide the request type of the Network Status Request message.
		Network Request ID	Use alphanumeric and/or special characters to provide the ID string for the Network Status Request message.
Server: Network (Counterparty System) Status Request Message	Requests a Network (counterparty system) Status Response message.	Transaction Flag	Start, Continue, End, or Start and End
		Network Request Type	Use alphanumeric and/or special characters to provide the request type of the Network Status Request message.
		Network Request ID	Use alphanumeric and/or special characters to provide the ID string for the Network Status Request message.

Action	Description	Action Parameter	Valid Values
Client: Network (Counterparty System) Status Response Message	Requests a Network (counterparty system) Status Response message.	Transaction Flag	Start, Continue, End, or Start and End
		Network Response Type	Use alphanumeric and/or special characters to provide the response type of the Network Status Response message.
		Network Response ID	Use alphanumeric and/or special characters to provide the ID string for the Network Status Response message.
Server: Network (Counterparty System) Status Request Message	Responds to a Network (counterparty system) Status Request message.	Transaction Flag	Start, Continue, End, or Start and End
		Network Response Type	Use alphanumeric and/or special characters to provide the response type of the Network Status Response message.
		Network Request ID	Use alphanumeric and/or special characters to provide the ID string for the Network Status Request message to which the server is responding.
		Network Response ID	Use alphanumeric and/or special characters to provide the ID string for the Network Status Response message.
		Last Network Response ID	Use alphanumeric and/or special characters to provide the ID string for the Last Network Status Response message. This field is used only when the **Network Response Type** is 2.

Action	Description	Action Parameter	Valid Values
Client: User Request Message	Requests a User Response message.	Transaction Flag	Start, Continue, End, or Start and End
		User Request ID	Use alphanumeric and/or special characters to provide the Request ID for the User Request message.
		User Request Type	Use alphanumeric and/or special characters to provide the request type of the User Request message.
		Username	Use alphanumeric and/or special characters to provide the user name for the User Request message.
		Password	Use alphanumeric and/or special characters to provide the password for the User Request message.
		New Password	Use alphanumeric and/or special characters to provide a new password for the User Request message. This field is used only if **User Request Type** is 3.

Action	Description	Action Parameter	Valid Values
Server: User Request Message	Requests a User Response message.	Transaction Flag	Start, Continue, End, or Start and End
		User Request ID	Use alphanumeric and/or special characters to provide the Request ID for the User Request message.
		User Request Type	Use alphanumeric and/or special characters to provide the request type of the User Request message.
		Username	Use alphanumeric and/or special characters to provide the user name for the User Request message.
		Password	Use alphanumeric and/or special characters to provide the password for the User Request message.
		New Password	Use alphanumeric and/or special characters to provide a new password for the User Request message. This field is used only if **User Request Type** is 3.

Action	Description	Action Parameter	Valid Values
Client: User Response Message	Responds to a user request message.	Transaction Flag	Start, Continue, End, or Start and End
		User Request ID	Use alphanumeric and/or special characters to provide the Request ID of the User Request message to which the client is responding.
		Username	Use alphanumeric and/or special characters to provide the Username for the User Response message.
		User Status	0 – 4,294,967,295
		User Status Text	Use alphanumeric and/or special characters to provide the text description associated with the User Status.
Server: User Response Message	Responds to a user request message.	Transaction Flag	Start, Continue, End, or Start and End
		User Request ID	Use alphanumeric and/or special characters to provide the Request ID of the User Request message to which the client is responding.
		Username	Use alphanumeric and/or special characters to provide the Username for the User Response message.
		User Status	0 – 4,294,967,295
		User Status Text	Use alphanumeric and/or special characters to provide the text description associated with the User Status.

FIXT Action Parameters

The table below lists the actions and action parameters for the FIXT protocol.

FIXT Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Heartbeat Message	Sends a keep-alive message.	Transaction Flag	Start, Continue, End, or Start and End
		Test Request ID	Use alphanumeric and/or special characters to define this field if the heartbeat is sent in response to a Test Request Message. The value input in this field should match the **Test Request ID** sent in the Test Request Message.

Action	Description	Action Parameter	Valid Values
Server: Heartbeat Message	Sends a keep-alive message.	Transaction Flag	Start, Continue, End, or Start and End
		Test Request ID	Use alphanumeric and/or special characters to define this field if the heartbeat is sent in response to a Test Request Message. The value input in this field should the **Test Request ID** for the **Heartbeat Message**.
Client: Test Request Message	Requests a keep-alive response.	Transaction Flag	Start, Continue, End, or Start and End
		Test Request ID	Use alphanumeric and/or special characters to define the value that should be echoed in the heartbeat response. The value input in this field should the **Test Request ID** for the **Heartbeat Message**.
Server: Test Request Message	Requests a keep-alive response.	Transaction Flag	Start, Continue, End, or Start and End
		Test Request ID	Use alphanumeric and/or special characters to define the value that should be echoed in the heartbeat response. The value input in this field should the **Test Request ID** for the **Heartbeat Message**.

Action	Description	Action Parameter	Valid Values
Client: Resend Request Message	Requests a resend of a range of unreceived request messages. Use **Begin Sequence Number** to set the start of the range and **End Sequence Number** to denote the end of the range. If the **End Sequence Number** is set to **0**, all messages starting at the **Begin Sequence Number** to the current message will be sent.	Transaction Flag	Start, Continue, End, or Start and End
		Begin Sequence Number	0 – 4,294,967,295
		End Sequence Number	0 – 4,294,967,295
Server: Resend Request Message	Requests a resend of a range of unreceived request messages. Use **Begin Sequence Number** to set the start of the range and **End Sequence Number** to denote the end of the range. If the **End Sequence Number** is set to **0**, all messages starting at the **Begin Sequence Number** to the current message will be sent.	Transaction Flag	Start, Continue, End, or Start and End
		Begin Sequence Number	0 – 4,294,967,295
		End Sequence Number	0 – 4,294,967,295
Client: Reject (session-level) Message	Sends a failure message to the server. If the **Referenced Sequence Number field** is not set, the system will use the last message sent by the peer.	Transaction Flag	Start, Continue, End, or Start and End
		Referenced Sequence Number (of the rejected message)	0 – 4,294,967,295
		Reference Tag ID (of the referenced FIX field)	0 – 4,294,967,295
		Referenced Message Type	Use alphanumeric and/or special characters to define the message type of the referenced FIX message.

Action	Description	Action Parameter	Valid Values
Client: Reject (session-level) Message	Sends a failure message to the server. If the **Referenced Sequence Number field** is not set, the system will use the last message sent by the peer.	Session Reject Reason (to identify reason for a session-level Reject message.)	0 – 4,294,967,295
		Message Text	Use alphanumeric and/or special characters to define the reason for the session-level rejection.
Server: Reject (session-level) Message	Sends a failure message to the client. If the **Referenced Sequence Number field** is not set, the system will use the last message sent by the peer.	Transaction Flag	Start, Continue, End, or Start and End
		Referenced Sequence Number (of the rejected message)	0 – 4,294,967,295
		Reference Tag ID (of the referenced FIX field)	0 – 4,294,967,295
		Reference Message Type	Use alphanumeric and/or special characters to define the message type of the referenced FIX message.
		Session Reject Reason (to identify reason for a session-level Reject message.)	0 – 4,294,967,295
		Message Text	Use alphanumeric and/or special characters to define the reason for the session-level rejection.
Client: Sequence Number Reset Message	Sends a requests to have reset the sequence number.	Transaction Flag	Start, Continue, End, or Start and End
		New Sequence Number	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
Server: Sequence Number Reset Message	Sends a requests to have reset the sequence number.	Transaction Flag	Start, Continue, End, or Start and End
		New Sequence Number	0 – 4,294,967,295
Client: Logout Message	Sends a message ending the session.	Transaction Flag	Start, Continue, End, or Start and End
		Text	Use alphanumeric and/or special characters to provide the text that will be sent when the session ends.
Server: Logout Message	Sends a message ending the session.	Transaction Flag	Start, Continue, End, or Start and End
		Text	Use alphanumeric and/or special characters to provide the text that will be sent when the session ends.
Client: Logon Message	Initiates a Logon request.	Transaction Flag	Start, Continue, End, or Start and End
		Heartbeat Interval (seconds between heartbeat messages)	0 – 4,294,967,295
		Next Expected Message Sequence Number	0 – 4,294,967,295
		Maximum Message Size	0 – 4,294,967,295
		Default Application Version ID	Use alphanumeric and/or special characters to describe the version of FIX that is being carried over the FIXT session.

Action	Description	Action Parameter	Valid Values
Server: Logon Message	Sends a response to a Logon request.	Transaction Flag	Start, Continue, End, or Start and End
		Heartbeat Interval (seconds between heartbeat messages)	0 – 4,294,967,295
		Next Expected Message Sequence Number	0 – 4,294,967,295
		Maximum Message Size	0 – 4,294,967,295
		Default Application Version ID	Use alphanumeric and/or special characters to describe the version of FIX that is being carried over the FIXT session.

FTP Action Parameters

The table below lists the actions and action parameters for FTP.

FTP Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Welcome Banner	Sends the server's welcome banner.	Banner Text	Use up to 256 alphanumeric and/or special characters to define the banner text.
Client: Login	Performs a login sequence.	Username	Use up to 256 alphanumeric and/or special characters to define the user name.
		Password	Use up to 256 alphanumeric and/or special characters to define the password.
Client: Directory Listing	Lists the files in the directory.	None	N/A

Action	Description	Action Parameter	Valid Values
Client: CWD	Performs a "change directory" command.	New directory	Use up to 256 alphanumeric and/or special characters to define the new directory name.
Client: Download	Downloads a file from the server.	Size of downloaded file	1 – 10,000,000
		Response data	Use the **Import Response Data** link to upload the data in which the client will download, and then select the filename from the **Response Data** drop-down menu.
Client: Upload	Uploads a file to the server.	Size of uploaded file	1 – 10,000,000
		Response Data	Use the **Import Response Data** link to upload the data in which the client will upload to the server, and then select the filename from the **Response Data** drop-down menu.
Client: QUIT	Disconnects the session.	None	N/A

Gmail Action Parameters

The table below lists the actions and the action parameters available for Gmail.

Gmail Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host.	Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
		Use Response	true or false
		Host	Any host
Client: Send Message	Send an e-mail message via Google GMail.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Use From Username Range	true or false
		From Username Prefix	String up to 128 bytes
		From Username Range Start	String up to 5 bytes
		From Username Range End	String up to 5 bytes
		From Domain	String up to 128 bytes
		Use To Username Range	true or false
		To Username Prefix	String up to 128 bytes
		To Username Range Start	String up to 5 bytes
		To Username Range End	String up to 5 bytes
		To Domain	String up to 128 bytes
		From Address	String up to 2048 bytes
		To Address(es)	String up to 2048 bytes
		Cc Address(es)	String up to 2048 bytes
		Bcc Address(es)	String up to 2048 bytes
		Subject	String up to 2048 bytes
		Static Message Text	String up to 4096 bytes
		Language	Custom, English, French, German, Italian, Spanish
		Message Wordcount Min	0 – 1,048,576

Action	Description	Action Parameter	Valid Values
		Message Wordcount Max	0 – 1,048,576
		Keyword List	String up to 4096 bytes
		Keywords in Subject	true or false
		Custom ISO-639 Language Code	String up to 16 bytes
		Attachment Filename	String
		Random Attachment	True or False
		Random File Size	0 – 33,554,432
		Random File Size Min	0 – 33,554,432
		Random File Size Max	0 – 33,554,432
		Attachment Content-Type	String up to 128 bytes
		File to load To Address(es) from	Any available file
		Static Message Text File	Any available file
		Custom Dictionary	Any available file
		Static Attachment	Any available file
		Static Attachment 2	Any available file
Client: Receive Message	Receive an e-mail message via Google GMail.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Use From Username Range	true or false
		From Username Prefix	String up to 128 bytes
		From Username Range Start	String up to 5 bytes
		From Username Range End	String up to 5 bytes
		From Domain	String up to 128 bytes
		Use To Username Range	true or false
		To Username Prefix	String up to 128 bytes
		To Username Range Start	String up to 5 bytes
		To Username Range End	String up to 5 bytes
		To Domain	String up to 128 bytes
		From Address	String up to 2048 bytes
		To Address(es)	String up to 2048 bytes
		Cc Address(es)	String up to 2048 bytes
		Bcc Address(es)	String up to 2048 bytes
		Subject	String up to 2048 bytes
		Static Message Text	String up to 2048 bytes
		Language	Custom, English, French, German, Italian, Spanish
		Message Wordcount Min	0 – 1,048,576

Action	Description	Action Parameter	Valid Values
		Message Wordcount Max	0 – 1,048,576
		Keyword List	String up to 4096 bytes
		Keywords in Subject	true or false
		Custom ISO-639 Language Code	String up to 16 bytes
		Attachment Filename	Any available file
		Random Attachment	True or False
		Random File Size	0 – 33,554,432
		Random File Size Min	0 – 33,554,432
		Random File Size Max	0 – 33,554,432
		Attachment Content-Type	0 – 33,554,432
		File to load To Address(es) from	Any available file
		Static Message Text File	Any available file
		Custom Dictionary	Any available file
		Static Attachment	Any available file
		Static Attachment 2	Any available file

Gnutella Action Parameters

The table below lists the actions and the action parameters available for the Gnutella protocol.

Gnutella Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Server Certificate	A file in PEM format containing the server's certificate.

Action	Description	Action Parameter	Valid Values
		Server Private Key	A file in PEM format containing the server's private key.
		Client Common Name	The client's common name (CN) as it appears in the client's certificate.
		Client CA Certificate	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the client's certificate.
		Client Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite

Action	Description	Action Parameter	Valid Values
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.

Action	Description	Action Parameter	Valid Values
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: TLS Discard Encrypted Data	Allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Server: TLS Discard Encrypted Data	Updates bulk decryption statistics if set to true.	Count Discarded Data	true or false
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	

Action	Description	Action Parameter	Valid Values
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

Action	Description	Action Parameter	Valid Values
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A

Action	Description	Action Parameter	Valid Values
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	
Client: Download	Simulates a download of a file.	Transaction Flag	Start, Continue, End, or Start and End
		Download Filename	Name of the file
		File Size in Bytes	Size of the file
		Download File	Available file
Client: Connect	Simulates a client connecting to the Gnutella network via an Ultrapeer.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Ping	Simulates a client ping and an accompanying pong.	Transaction Flag	Start, Continue, End, or Start and End
		Files Shared	0 – random
		Kbytes Shared	0 – random
Server: Ping	Simulates a server ping and an accompanying pong.	Transaction Flag	Start, Continue, End, or Start and End
		Files Shared	0 – random
		Kbytes Shared	0 – random
Client: Query	Simulates a search for a file.	Transaction Flag	Start, Continue, End, or Start and End
		Query Search Term	alphanumeric characters
		QueryHits Returned	0 – random

Gopher Action Parameters

Gopher is a document search and retrieval protocol. With Gopher, servers provide links to related topics, and users can access this information using a client software. Its intended goal is to provide a simple way to pass information from the server to the clients.

For the **Server: Response (OK)** action, there are several action parameters that you can configure for the response. If **Response Data (Resource)** is set and **Response Data (String)** is not, then the specified resource file will be used. If you do not define values for either Response Data fields, **Random Data Min** and **Random Data Max** will be used to generate a random response.

Note: All action parameters that are left blank will generate random values.

Gopher Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Request	Connects to the Gopher server and sends a request.	Selector	Alphanumeric and/or special characters can be used to define the selector.

Action	Description	Action Parameter	Valid Values
Server: Response (OK)	Sends a response to the client with the specified data.	Response Data (String)	Alphanumeric and/or special characters can be used to define the response data.
		Response Data (Resource)	Use the **Import Response Data (Resource)** link to upload a resource file, then select the desired file from the **Response Data (Resource)** drop-down menu. The content of the resource file will be used as the response data.
		Random Data (Min)	0 – 4,294,967,295
		Random Data (Max)	0 – 4,294,967,295

GTalk Action Parameters

The table below lists the actions and the action parameters available for the GTalk protocol.

GTalk Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite

Action	Description	Action Parameter	Valid Values
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Server Certificate	A file in PEM format containing the server's certificate.
		Server Private Key	A file in PEM format containing the server's private key.
		Client Common Name	The client's common name (CN) as it appears in the client's certificate.
		Client CA Certificate	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the client's certificate.
		Client Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert

Action	Description	Action Parameter	Valid Values
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false

Action	Description	Action Parameter	Valid Values
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: TLS Discard Encrypted Data	Allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false

Action	Description	Action Parameter	Valid Values
Server: TLS Discard Encrypted Data	Updates bulk decryption statistics if set to true.	Count Discarded Data	true or false
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	

Action	Description	Action Parameter	Valid Values
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST

Action	Description	Action Parameter	Valid Values
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A

Action	Description	Action Parameter	Valid Values
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	
Client: Pre-auth	Simulates a GTalk login prior to authorization.	Transaction Flag	Start, Continue, End, or Start and End
Client: Post-auth	Simulates a GTalk login post-authorization.	Transaction Flag	Start, Continue, End, or Start and End
Client: Chat	Simulates a GTalk conversation between a user and a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Client Messages	0 – Random
		Peer Messages	0 – Random
Client: IM: User	Simulates a single instant message from the user to a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Message	Alphanumeric characters

Action	Description	Action Parameter	Valid Values
Server: IM: Peer	Simulates a single instant message from a peer to the user.	Transaction Flag	Start, Continue, End, or Start and End
		Message	Alphanumeric characters
Client: Presence Notification	Simulates a single presence notification from user to peer.	Transaction Flag	Start, Continue, End, or Start and End
Server: Presence Notification	Simulates a single presence notification from peer to user.	Transaction Flag	Start, Continue, End, or Start and End
Client: Request cleardot gif	Performs an HTTP Get for cleardot.gif.	Transaction Flag	Start, Continue, End, or Start and End
		Header host name used for cleardot.gifHTTP Get	
		User Agent header used for HTTP Get of cleardot.gif	
		URL used for cleardot gif file	
Server: Return cleardot gif	Returns cleardot.gif to the client.	Transaction Flag	Start, Continue, End, or Start and End
		Server id header returned with HTTP Get of cleardot.gif	
		Content type header returned with HTTP Get of cleardo	

Action	Description	Action Parameter	Valid Values
Client: Create a session with the Google server	Issues the HTTP GET for the create_ session url.	Transaction Flag	Start, Continue, End, or Start and End
		Email address of GTalk client (voicemail sender). Use	
		Session type header used for HTTP Create Session Get	
		URL used for HTTP Create Session Get	
		User Agent header used for HTTP Create Session Get	
		Host name header used for HTTP Create Session Get	
Server: Return session information	Returns the connection details for the new session.	Transaction Flag	Start, Continue, End, or Start and End
		Server id header returned by HTTP Create Session Get	
		Content type header returned by HTTP Create Session	
Client: User Send File	Simulates a single file exchange from user to peer.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: User Receive File	Simulates a single file exchange from peer to user.	Transaction Flag	Start, Continue, End, or Start and End
Client: User Send Voicemail	Simulates a single voicemail exchange from user to peer.	Transaction Flag	Start, Continue, End, or Start and End
Client: User End Voicemail	Simulates a single voicemail final message, post-recording.	Transaction Flag	Start, Continue, End, or Start and End
Client: User to Peer Voicecall	Simulates a voicecall from user to peer.	Transaction Flag	Start, Continue, End, or Start and End
Client: User End Voicemail	Simulates a voicecall hangup by user.	Transaction Flag	Start, Continue, End, or Start and End
Client: Get File Data	Receive file data via UDP in file transfer.	Transaction Flag	Start, Continue, End, or Start and End
Client: Put File Data	Put file data via UDP in file transfer.	Transaction Flag	Start, Continue, End, or Start and End
Client: Put Voicemail Data	Send voicemail Data.	Transaction Flag	Start, Continue, End, or Start and End
Client: Voice Data	Exchange call data with callee.	Transaction Flag	Start, Continue, End, or Start and End
Client: Download	Simulates a download of a file.	Transaction Flag	Start, Continue, End, or Start and End
		Download Filename	Name of the file
		File Size in Bytes	Size of the file
		Download File	Available file

Action	Description	Action Parameter	Valid Values
Client: Connect	Simulates a client connecting to the Gnutella network via an Ultrapeer.	Transaction Flag	Start, Continue, End, or Start and End
Client: Ping	Simulates a client ping and an accompanying pong.	Transaction Flag	Start, Continue, End, or Start and End
		Files Shared	0 – random
		Kbytes Shared	0 – random
Server: Ping	Simulates a server ping and an accompanying pong.	Transaction Flag	Start, Continue, End, or Start and End
		Files Shared	0 – random
		Kbytes Shared	0 – random
Client: Query	Simulates a search for a file.	Transaction Flag	Start, Continue, End, or Start and End
		Query Search Term	alphanumeric characters
		QueryHits Returned	0 – random

H248 Action Parameters

The table below lists the actions and the action parameters available for H248.

H248 Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Event: AL OF	Simulates the notification from an MG to an MGC when an analog endpoint goes off-hook.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65536, or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Notify Reply	Simulates the response to a notification.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Signal: CG DT	Simulates the notification from an MGC to an MG requesting to send dial tone.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Digit Map	An H248 digit map (see RFC 3015 section 7.1.14.3)

Action	Description	Action Parameter	Valid Values
Client: Modify Reply	Simulates the response to a termination modify request.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
Server: Delay	Delays the server's response for the amount of time specified for Number of Milliseconds.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Application Manager will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Event: DD CE	Simulates the notification from an MG to an MGC, reporting a match between collected digits and the supplied dialplan.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Dialed Digits	Up to 127 alphanumeric characters, which may contain tokens (example: 91XXXXXXXXX)

Action	Description	Action Parameter	Valid Values
Server: Add Request	Adds a termination to a context.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Client: Add Reply	Simulates the response to a termination addition request.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Add RTP Termination	Adds a RTP termination to a context.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Stream Mode	Receive, Send, Send and Receive
		Audio Data	Any available audio codec

Action	Description	Action Parameter	Valid Values
Client: Add RTP Termination Reply	Simulates the response to a RTP termination addition request.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Audio Data	Any available audio codec

Action	Description	Action Parameter	Valid Values
Server: Signal: CG RT	Simulates the signaling of a ringback tone from a MGC to a MG.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Modify Descriptor	Simulates a local or remote descriptor modification from a MGC to a MG.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Descriptor Type	Local or Remote
		Audio Data	Any available audio codec

Action	Description	Action Parameter	Valid Values
Server: Signal: None	Simulates a modify request from an MGC to a MG, removing all signals from the context.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Modify Stream Mode	Simulates the stream mode modification from a MGC to a MG.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token
		Stream Mode	Receive, Send, Send and Receive
Client: Bidirectional Stream	Simulates a bidirectional stream between a client and a server.	Transaction Flag	Start, Continue, End, or Start and End
		Payload Type (F)	Any available payload type
		Initial Sequence Number (F)	0 – 65,535
		Initial Timestamp (0xHex Format) (F)	Up to 10 hexadecimal characters (0 – 9, a - f)

Action	Description	Action Parameter	Valid Values
		Use Payload Type Defaults?	True or False
		Duration Type (F)	Size-based or Time-based
		Stream duration (ms) (F)	1 – 600,000
		Timestamp Increment (F)	0 – 255
		Buffer Size (1024 bytes max) (F)	1 – 1024
		Buffer Latency (1000 ms max)	1 – 1000
		SSRC (0xHex Format) (F)	Up to 10 hexadecimal characters (0 – 9, a - f)
		Mark First Packet? (F)	True or False
		Strip Wave Header?	True or False
		Raw File Size (5242880 bytes max) (F)	2 – 52,428,800
		Payload Type (R)	Any available payload type
		Initial Sequence Number (R)	0 – 65,535

Action	Description	Action Parameter	Valid Values
		Initial Timestamp (0xHex Format) (R)	Up to 10 hexadecimal characters (0 – 9, a - f)
		Use Payload Type Defaults?	True or False
		Duration Type (R)	Size-based or Time-based
		Stream duration (ms) (R)	1 – 600,000
		Timestamp Increment (R)	0 – 255
		Buffer Size (1024 bytes max) (R)	1 – 1024
		Buffer Latency (1000 ms max)	1 – 1000
		SSRC (0xHex Format) (R)	Up to 10 hexadecimal characters (0 – 9, a - f)
		Mark First Packet? (R)	True or False
		Strip Wave Header?	True or False
		Raw File Size (5242880 bytes max) (R)	2 – 52,428,800

Action	Description	Action Parameter	Valid Values
		Raw File to Stream (F)	Any available file
		Raw File to Stream (R)	Any available file
Client: RTCP Report	Enacts the RTCP sender report, sender description, and bye packet.	Transaction Flag	Start, Continue, End, or Start and End
		Bye Reason	Up to 254 characters
Server: RTCP Report	Enacts the RTCP sender report, sender description, and bye packet.	Transaction Flag	Start, Continue, End, or Start and End
		Bye Reason	Up to 254 characters
Client: Event: AL ON	Simulates the notification from an MG to an MGC when an analog endpoint goes on hook.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Server: Subtract Request	Removes a termination from a context.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

Action	Description	Action Parameter	Valid Values
Client: Subtract Reply	Simulates the response to a termination subtraction request.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID	0 – 65,536 or a token
		Termination ID Type	Choose, All, Manual, Random
		Termination ID (Manual)	choose, all, or manual
		Termination ID Depth	All Lower Levels, One Level
		Termination ID Wildcard Bits	1 – 64
		Context ID Type	Null, Choose, All, Random, Manual
		Context ID (Manual)	1 – 4,294,967,294 or a token

HTTP Action Parameters

The table below lists the actions and action parameters for the HTTP protocol.

HTTP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Client Delay	Delays the client response for the amount of time specified by **Number of Milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	0 – 4,294,967,295
Server: Server Delay	Delays the client response for the amount of time specified by **Number of Milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	0 – 4,294,967,295
Client: Raw Request	Sends a request that is downloaded from the specified URL.	Request Data	Use the **Import Request Data** link to import the file from which the request will be downloaded. Once the file has been uploaded, select the filename from the **Request Data** drop-down menu.
		Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Server: Raw Response	Sends a response that is downloaded from the specified URL.	Response Data	Use the **Import Response Data** link to import the file from which the response will be downloaded. Once the file has been uploaded, select the filename from the **Response Data** drop-down menu.
		Transaction Flag	Start, Continue, End, or Start and End
Client: GET	Performs a GET request for the specified URL.	Transaction Flag	Start, Continue, End, or Start and End
		Proxy Mode	on or off
		Request path	Use up to 128 alphanumeric and/or special characters to define the URL that is requested in the HTTP method.
		URL escape	true or false
		Enable persistent HTTP sessions	Set this to **on** or **off** to toggle the Keep Alive headers. When performing a Client Simulation test or a 1-arm test with more than one HTTP client action in a flow, be sure to set Enable persistent HTTP sessions of the GET action to on. If this value is set to off, the server will close the TCP connection as soon as it sends the first HTTP response, causing the actions within the Conditional Request to be ignored. The default setting for Enable persistent HTTP sessions of the GET action is set to on.
		Custom Accept header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data will override default values used in the Accept header.
		Custom Encoding header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data will override default values used in the Accept-Encoding HTTP header.

Action	Description	Action Parameter	Valid Values
Client: GET	Performs a GET request for the specified URL.	Custom Language header	Use up to 128 alphanumeric and/or special characters to define the Custom Language header. This data will override default values used in the Accept-Language HTTP header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This data will override default values used in the User-Agent HTTP header.
		Custom If-None-Match	Sets the 'If-None-Match' header for the server response.
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1024 alphanumeric and/or special characters to define the value of the cookie session.
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
Client: GET Authenticated	Performs a GET request for the specified URL with authentication.	Transaction Flag	Start, Continue, End, or Start and End
		HTTP Authentication scheme to use	Digest Authentication Basic Authentication No Authentication
		User name for authentication	Use up to 128 alphanumeric and/or special characters to define the username that will be encoded into the request.
		Password for authentication	Use up to 128 alphanumeric and/or special characters to define the username that will be encoded into the request.

Action	Description	Action Parameter	Valid Values
		Request path	Use up to 128 alphanumeric and/or special characters to define the URL that is requested in the HTTP method.
		Enable persistent HTTP sessions	Set this to **on** to toggle the Keep Alive headers. Set this to **off** to disable this option. When performing a Client Simulation test or a 1-arm test with more than one HTTP client action in a flow, be sure to set Enable persistent HTTP sessions of the GET action to on. If this value is set to off, the server will close the TCP connection as soon as it sends the first HTTP response, causing the actions within the Conditional Request to be ignored. The default setting for Enable persistent HTTP sessions of the GET action is set to on.
		Custom Accept header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data will override default values used in the Accept HTTP header.
Client: GET Authenticated	Performs a GET request for the specified URL with authentication.	Custom Encoding header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data will override default values used in the Accept-Encoding HTTP header.
		Custom Language header	Use up to 128 alphanumeric and/or special characters to define the Language header. This data will override default values used in the Accept-Language HTTP header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This data will override default values used in the User-Agent HTTP header.
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.

Action	Description	Action Parameter	Valid Values
Client: GET Authenticated	Performs a GET request for the specified URL with authentication.	Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
Client: GetURIs	Simulates a client GET Request, and a server 200 OK response.	Transaction Flag	Start, Continue, End, or Start and End
		String for response data	The string that will be appended to the response data that is sent.
		Random response min length	0 – 4,294,967,295
		Random response max length	0 – 4,294,967,295
		File Generator	
		URL escape	true or false
		Enable persistent HTTP sessions	Set this to **on** to toggle the Keep Alive headers. Set this to **off** to disable this option. When performing a Client Simulation test or a 1-arm test with more than one HTTP client action in a flow, be sure to set Enable persistent HTTP sessions of the GET action to on. If this value is set to off, the server will close the TCP connection as soon as it sends the first HTTP response, causing the actions within the Conditional Request to be ignored. The default setting for Enable persistent HTTP sessions of the GET action is set to on.
		Custom Accept Header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data will override default values used in the Accept HTTP header.
		Custom Encoding Header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data will override default values used in the Accept-Encoding HTTP header.

Action	Description	Action Parameter	Valid Values
		Custom Language Header	Use up to 128 alphanumeric and/or special characters to define the Language header. This data will override default values used in the Accept-Language HTTP header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This data will override default values used in the User-Agent HTTP header.
		Custom If-None-Match	
		Cookie Name	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Cookie Value	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.
		Custom Header Name	Use up to 256 alphanumeric and/or special characters to define the name of the header that will be added to the HTTP session.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
		File with a list of URIs	N/A
		File for response data	N/A
		Custom Headers File	N/A

Action	Description	Action Parameter	Valid Values
Client: POST	Performs a POST request for the specified URL.	Transaction Flag	Start, Continue, End, or Start and End
		'Content-MD5' header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Enable chunked encoding	Set this to **on** to split data up, and send it in chunks. Set this to **off** to disable this option.
		Default size for HTTP chunked responses	0 – 4,294,967,295 When the chunk size is set to 0, BreakingPoint will assume that the user-provided file is already chunk encoded.
		Requested path	Use up to 128 alphanumeric and/or special characters to define the URL that is requested in the HTTP method.
		Content-Type	Use up to 128 alphanumeric and/or special characters to define the Content-Type header.

Action	Description	Action Parameter	Valid Values
Client: POST	Performs a POST request for the specified URL.	POST content	Use alphanumeric and/or special characters to define the data that will be sent with the POST request.
		POST content	Use the **Import POST content** link to import the file that contains the data that will be sent with the POST request. Once the file has been imported, you can select it from the **Post content** drop-down menu.
		URL for POST content	Use alphanumeric and/or special characters to define the URL that will be appended to the POST request body.
		Min amt of random data	0 – 65,535
		Max amt of random data	0 – 65,535
		Custom Accept header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data overrides values used in the Accept HTTP header.

Action	Description	Action Parameter	Valid Values
Client: POST	Performs a POST request for the specified URL.	Custom Encoding header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data overrides values used in the Accept-Encoding HTTP header.
		Custom Language header	Use up to 128 alphanumeric and/or special characters to define the Language header. This data overrides values used in the Accept-Language HTTP header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This overrides values used in the User-Agent HTTP header.
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the User-Agent field. name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.
Client: POST	Performs a POST request for the specified URL.	Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.

Action	Description	Action Parameter	Valid Values
Client: PUT	Performs a PUT request for the specified URL.	Transaction Flag	Start, Continue, End, or Start and End
		Content MD-5 header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Enable chunked encoding	Set this to **on** to split data up, and send it in chunks. Set this to **off** to disable this option.
		Default size for HTTP chunked responses	0 – 4,294,967,295 When the chunk size is set to 0, BreakingPoint will assume that the user-provided file is already chunk encoded.
		Requested path	Use up to 128 alphanumeric and/or special characters to define the URL that is requested in the HTTP method.

Action	Description	Action Parameter	Valid Values
Client: PUT	Performs a PUT request for the specified URL.	Content-Type	Use up to 128 alphanumeric and/or special characters to define the value of the Content-Type header.
		String for PUT data	Use alphanumeric and/or special characters to define the data that will be sent with the PUT request.
		URL for PUT data	Use alphanumeric and/or special characters to define the URL that will be appended to the PUT request body.
		Min amt of random data	0 – 65,535
		Max amt of random data	0 – 65,535
		Custom Accept header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data overrides values used in the Accept HTTP header.
		Custom Encoding header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data overrides values used in the Accept-Encoding HTTP header.

Action	Description	Action Parameter	Valid Values
Client: PUT	Performs a PUT request for the specified URL.	Custom Language Header	Use up to 128 alphanumeric and/or special characters to define the Language header. This data overrides values used in the Accept-Language HTTP header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This overrides values used in the User-Agent HTTP header.
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the User-Agent field. name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
Client: THINK	Inserts a delay into the HTTP flow. Used to model client think time, or server response time latency.	Think time in milliseconds	0 – 65,535

Action	Description	Action Parameter	Valid Values
Server: Response 200 (OK)	Returns an HTTP error using the information detailed for the Server Response 200 (OK) options.	HTTP Compression	Select gzip, deflate, or none to compress the response data.
		'Content-MD5' header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Enable chunked encoding	Set this to **on** to split data up, and send it in chunks. Set this to **off** to disable this option.
		Content-Type	Use up to 128 alphanumeric and/or special characters to define the value of the Content-Type header.
		HTTP chunk response size	0 – 4,294,967,295 When the chunk size is set to 0, BreakingPoint will assume that the user-provided file is already chunk encoded.
		HTTP response data	Use up to 256 alphanumeric and/or special characters to define data sent in the HTTP response.
Server: Response 200 (OK)	Returns an HTTP error using the information detailed for the Server Response 200 (OK) options.	Transaction Flag	Start, Continue, End, or Start and End
		HTTP Compression	Select gzip, deflate, or none to compress the response data.
		'Content-MD5' header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Enable chunked encoding	Set this to **on** to split data up, and send it in chunks. Set this to **off** to disable this option.

Action	Description	Action Parameter	Valid Values
Server: Response 200 (OK)	Returns an HTTP error using the information defined for the Server Response 200 (OK) options.	Content-Type	Use up to 128 alphanumeric and/or special characters to define the value of the Content-Type header.
		HTTP chunk response size	0 – 4,294,967,295 When the chunk size is set to 0, BreakingPoint will assume that the user-provided file is already chunk encoded.
		Response data	Use up to 128 alphanumeric and/or special characters to define the data returned by the Web server.
		Response data file	Click the **Import Response data** link to import the file that will be used as the response data.
		Random response min length	0 – 4,294,967,295
		Random response max length	0 – 4,294,967,295
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.

Action	Description	Action Parameter	Valid Values
Server: HTTP 404 Error	Returns an HTTP error using the information detailed for the Server: HTTP 404 Error options.	Transaction Flag	Start, Continue, End, or Start and End
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Generate 'Content-MD5' header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		Response Data	Use up to 128 alphanumeric and/or special characters to define the data returned by the Web server.
		Response Data	Click the **Import Response data** link to import the file that will be used as the response data.
		Random response min length	0 – 65,535
		Random response max length	0 – 65,535
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.

Action	Description	Action Parameter	Valid Values
Server: Response 401 Unauthorized	Returns an error for an unauthorized request from the client.	Transaction Flag	Start, Continue, End, or Start and End
		HTTP Authentication scheme to use	Digest Authentication Basic Authentication No Authentication
		Response Data	Use up to 128 alphanumeric and/or special characters to define the data returned by the Web server.
		Response Data	Use the **Import Response data** link to import the file that will be used as the response data. Once the file has been uploaded, you can select it from the **Response Data** drop-down menu.
		Random response min length	0 – 65,535
		Random response max length	0 – 65,535
		Name of cookie to save	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to save	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.

Action	Description	Action Parameter	Valid Values
HTTP Generic Error	Returns a generic error with a custom code and message.	Transaction Flag	Start, Continue, End, or Start and End
		HTTP status code	A three digit status code (e.g., 200, 404, 503)
		HTTP status message	Use up to 256 alphanumeric and/or special characters to define the status message returned in the error.
		Keep Alive	Set this to **on** to use persistent connections. Set this to **off** to disable this option.
		Generate Content-MD5 header	Set this to **on** to include a header with the MD5 hash of the POST request body data. Set this to **off** to disable this option.
		String for response data	Use up to 128 alphanumeric and/or special characters to define the data returned by the Web server.
		Response Data	Use the **Import Response data** link to import the file that will be used as the response data. Once the file has been uploaded, you can select it from the **Response Data** drop-down menu.
		Random response min length	0 – 65,535
HTTP Generic Error	Returns a generic error with a custom code and message.	Random response max length	0 – 65,535
		Name of cookie to set	Use up to 256 alphanumeric and/or special characters to define the name of the cookie that will be added to the HTTP session.
		Value of cookie to set	Use up to 1,024 alphanumeric and/or special characters to define the value of the cookie session.

HTTPS Action Parameters

The table below lists the actions and action parameters for HTTPS.

Note: This traffic is simulated SSL traffic. To generate actual SSL traffic, add the Start TLS and/or Accept TLS action to any TCP flow in the Super Flow editor.

HTTPS Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Client Delay	Delays the client response for the amount of time specified by **Number of Milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	0 – 4,294,967,295
Server: Server Delay	Delays the server response for the amount of time specified by **Number of Milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	0 – 4,294,967,295
Client: Client Hello	Initiates a TLS handshake with the SSL server	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Hello	Continues the TLS handshake with the client	Transaction Flag	Start, Continue, End, or Start and End
Server: Certificates	Sends a list of certificates to the client to authenticate	Transaction Flag	Start, Continue, End, or Start and End
		Number of certificates	1 – 100

Action	Description	Action Parameter	Valid Values
Server: Hello Done	Signals the end of the transaction negotiation	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Key Exchange	Sends the premastersecret encrypted with the server's public key	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Change Cipher Spec	Instructs the server that all subsequent data will be encrypted using the connection settings that were negotiated	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Change Cipher Spec	Instructs the client that all subsequent data will be encrypted using the connection settings that were negotiated	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Finished`	Sends an encrypted hash to verify the new encryption parameters	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Finished	Sends an encrypted hash to verify the new encryption parameters	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Alert	Sends an alert record from the client	Transaction Flag	Start, Continue, End, or Start and End
		Alert Level	Warning or Fatal

Action	Description	Action Parameter	Valid Values
		Alert Type	Close Notify Unexpected Message Bad Record Mac Decryption Failed Record Overflow Decompression Failure Handshake Failure Bad Certificate Unsupported Certificate Certificate Revoked Certificate Expired Certificate Unknown Illegal Parameter Unknown CA Access Denied Decode Error Decrypt Error Export Restriction Protocol Version Insufficient Security Internal Error User Canceled No Renegotiation

Action	Description	Action Parameter	Valid Values
Server: Server Alert	Sends an alert record from the server	Transaction Flag	Start, Continue, End, or Start and End
		Alert Level	Warning or Fatal

Action	Description	Action Parameter	Valid Values
		Alert Type	Close Notify Unexpected Message Bad Record Mac Decryption Failed Record Overflow Decompression Failure Handshake Failure Bad Certificate Unsupported Certificate Certificate Revoked Certificate Expired Certificate Unknown Illegal Parameter Unknown CA Access Denied Decode Error Decrypt Error Export Restriction Protocol Version Insufficient Security Internal Error User Canceled No Renegotiation

Action	Description	Action Parameter	Valid Values
Client: Client Application Data (encrypted)	Sends a block of encrypted data	Transaction Flag	Start, Continue, End, or Start and End
		Minimum data length	1 – 16,383
		Maximum data length	1 – 16,383
Server: Server Application Data (encrypted)	Sends a block of encrypted data	Transaction Flag	Start, Continue, End, or Start and End
		Minimum data length	1 – 16,383
		Maximum data length	1 – 16,383

IAX2 Action Parameters

The table below lists below actions and action parameters for IAX2.

IAX2 Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLSv TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Cipher	Any available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Common Name	N/A
		Client Cert Verify Mode	Do Not Check Cert, Allow Untrusted Cert, or Require Trusted Cert
		Server Certificate	Available PEM formatted cert file
		Server Private Key	Available PEM formatted key file
		Client CA Certificate	Available PEM formatted cert file
		Decryption Mode	Auto Decrypt Discard/Count Discard/NoCount
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.

Action	Description	Action Parameter	Valid Values
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.

Action	Description	Action Parameter	Valid Values
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: Discard Encrypted Data	An optimization that allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Server: Discard Encrypted Data	An optimization that allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false

Action	Description	Action Parameter	Valid Values
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	

Action	Description	Action Parameter	Valid Values
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

Action	Description	Action Parameter	Valid Values
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A

Action	Description	Action Parameter	Valid Values
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	Available file

IMAPv4-Advanced Action Parameters

The table below lists the actions and action parameters for IMAPv4.

IMAP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of discountability this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to a POP3 server.	Starting Tag Value	0 – 4,294,967,295
		Username	Enter up to 128 alphanumeric and/or special characters to define the username used to log into the IMAP server.
		Password	Enter up to 128 alphanumeric and/or special characters to define the password used to log into the IMAP server.
* Client: Logout	Simulates a logout request from the client to the server.	Command Tag Value	0 – Random

Action	Description	Action Parameter	Valid Values
Client: Retrieve Mail	Simulates the retrieval of an e-mail message.	Starting Tag Value	0 – 4,294,967,295
		Raw Message URL	Select a file from the **Raw Message URL** drop-down menu. The contents of this file will comprise the entire mail message – including the headers. If the desired file is not listed on the menu, use the **Import Raw Message URL** link to upload a file that will be used for the entire URL message.
* When a flow uses the client IMAP Logout action, the external IMAP server will send a logout response and then close the TCP connection. Any actions after the TCP connection starts to close will have inconsistent results. Therefore, no other actions for that flow should come after the IMAP Logout action.			
Client: Retrieve Mail	Simulates the retrieval of an e-mail message.	From:	Enter up to 128 alphanumeric and/or special characters to define the **From:** field.
		To:	Enter up to 128 alphanumeric and/or special characters to define the **To:** field.
		Subject:	Enter up to 128 alphanumeric and/or special characters to define the **Subject:** field.
		Text:	Enter up to 128 alphanumeric and/or special characters to define the body of the message.
		Attachment URL	Select a file from the **Attachment URL** drop-down menu. This file will be incorporated into the generated mail message. If the desired file is not listed on the menu, use the **Import Attachment URL** link to upload the attachment file that will be used.
		Attachment filename	Enter up to 256 alphanumeric and/or special characters to define the attachment filename.
		Attachment size	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
Client: Quit	Simulates a QUIT command from the IMAP server.	Starting Tag Value	0 – 4,294,967,295
* When a flow uses the client IMAP Logout action, the external IMAP server will send a logout response and then close the TCP connection. Any actions after the TCP connection starts to close will have inconsistent results. Therefore, no other actions for that flow should come after the IMAP Logout action.			

Informix Action Parameters

The table below lists the actions and action parameters for Informix.

Informix Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Login	Simulates a login to the Informix database	Database Username	Enter up to 255 alphanumeric and/or special characters for the username that will be used to log into the database.
		Database Password	Enter up to 255 alphanumeric and/or special characters for the password that will be used to log into the database.
		Database Hostname	Enter up to 255 alphanumeric and/or special characters for the database's hostname.
Client: SQL Query	Simulates an SQL query and response. The values entered for **Columns** and **Rows** will determine the number of columns and rows the query will return. If **Column Names** are specified, then they need to be entered as comma-delimited values.	SQL Query	Alphanumeric and/or special characters can be used to define the SQL query.
		Columns	0 – 4,294,967,295
		Rows	0 – 4,294,967,295
		Column Names	Alphanumeric and/or special characters can be used to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.

IPP Action Parameters

The Internet Printing Protocol (IPP) is a standard network protocol that supports access control, authentication, and encryption. Using this implementation of IPP, you can create a connection to an IPP server, send a print job request, and receive a response from the server.

The table below lists the action parameters you can use to set up an IPP flow.

IPP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Print	Connects to the IPP server, sends a print request, and receives a response from the server.	Print Job Size (bytes)	1 – 1,048,576
		Print Job Data File	Use the Import Print Job Data File link to upload the content for the print request.
		Username	Use up to 128 alphanumeric and/or special characters to define the user's name who is making the request.

IRC Action Parameters

The table below lists the actions and action parameters for IRC.

IRC Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds** milliseconds.	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to the chat application		N/A

Action	Description	Action Parameter	Valid Values
Client: Chat	Simulates an IRC channel conversation with multiple virtual peers	Channel	Enter up to 31 alphanumeric and/or special characters to define the IRC channel. The value defined for this parameter must begin with '#'.
		Client Messages	0 – 4,294,967,295
		Peer Messages	0 – 4,294,967,295

iTunes Action Parameters

The table below lists the actions and the action parameters available for the iTunes protocol.

iTunes Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Cipher	Any available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
		Client Authentication Enabled	true or false
		Client Common Name	N/A
		Client Cert Verify Mode	Do Not Check Cert, Allow Untrusted Cert, or Require Trusted Cert
		Server Certificate	Available PEM formatted cert file
		Server Private Key	Available PEM formatted key file
		Client CA Certificate	Available PEM formatted cert file
		Decryption Mode	Auto Decrypt Discard/Count Discard/NoCount
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		1st Cipher	An available ciphersuite
		2nd Cipher	An available ciphersuite

Action	Description	Action Parameter	Valid Values
		3rd Cipher	An available ciphersuite
		4th Cipher	An available ciphersuite
		5th Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Certificate	A file in PEM format containing the client's certificate.
		Client Private Key	A file in PEM format containing the client's private key.
		Server Common Name	The server's common name (CN) as it appears in the server's certificate.
		Server CA Cert	A file in PEM format containing the certificate of the Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.
		Server Cert Verify Mode	Allow Untrusted Cert Do Not Check Cert Require Trusted Cert

Action	Description	Action Parameter	Valid Values
		Decryption Mode	Auto Decrypt Discard/Count Discard/No Count
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Client: Discard Encrypted Data	An optimization that allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Server: Discard Encrypted Data	An optimization that allows encrypted data received on this flow to be discarded before decrypting it.	Count Discarded Data	true or false
Client: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000
Server: Delay	Pause for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds	1 – 1,000,000
		Maximum Number of Milliseconds	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Server: Raw Message		Transaction Flag	Start, Continue, End, or Start and End
		String	Enter the string to be used.
		Filename	Select a file to import.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters

Action	Description	Action Parameter	Valid Values
Client: Update Dest Address	Updates the destination address in subsequent flows with the value from a previous PCRE match.	Destination Host	Client or Server
		Match Variable (0-9)	0 – 9
Client: Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID	The ID of the flow to update.
		Match Variable (0-9)	0 – 9
Client: Update Receive Window	Updates the receive window with the specified value.	Receive Window Size (bytes)	
Server: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters

Action	Description	Action Parameter	Valid Values
Client: Verify File	Verifies data coming from the server with a specified resource file.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		File to verify	Available file
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	
Client: Close	Close the connection on the TCP transport level.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Server: Close	Close the connection on the TCP transport level.	Transaction Flag	Start, Continue, End, or Start and End
		FIN or RST	FIN or RST
Client: Fail	N/A	N/A	N/A
Server: Fail	N/A	N/A	N/A
Client: Log as Target	For Lawful Intercept tests, this action is used to generate a target Super Flow that does not contain a needle. Including this action results in a Lawful Intercept test logging the presence of the Super Flow as if it contained a needle.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Add Flow Dictionary	Provides the ability to add a dictionary to the flow.	Dictionary ID	0 – 9
		Dictionary File	The resource file to be used.
		Dictionary Delimiter Type	New Line or Custom
		Dictionary Custom Delimiter	N/A
Client: Add Markov Flow Dictionary	Provides the ability to add a dictionary of Markov text bodies to the flow.	Dictionary ID	
		Quantity	
		Markov Minimum Word Count	
		Markov Maximum Word Count	
		Markov Text Length	
		Markov Keywords	
		Markov Language	English French Italian German Spanish Japanese
		Markov Database	Available file
Client: Get Bag	Gets the bag file.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Server: Send Bag	Sends the bag file.	Transaction Flag	Start, Continue, End, or Start and End
Client: Search	Submits a search query for this flow's media type.	Transaction Flag	Start, Continue, End, or Start and End
		Search Terms	Up to 256 alphanumeric characters
Server: Search Results	Provides search results for the requested query.	Transaction Flag	Start, Continue, End, or Start and End
		Search Terms	Up to 256 alphanumeric characters
Client: View Item Request	Requests an item's iTunes description page from the search results.	Transaction Flag	Start, Continue, End, or Start and End
Server: View Item Response	Sends an item description page for the appropriate media type.	Transaction Flag	Start, Continue, End, or Start and End
Client: Purchase Request	Sends a client purchase request.	Transaction Flag	Start, Continue, End, or Start and End
Server: Purchase Response	Sends a server Apple WebObjects response for a purchase request.	Transaction Flag	Start, Continue, End, or Start and End
Client: Download Request	Sends a download request for the flow's media type.	Transaction Flag	Start, Continue, End, or Start and End
Server: Download Response	Responds to a client Download Request with the appropriate media type.	Transaction Flag	Start, Continue, End, or Start and End

Jabber Action Parameters

Jabber is the core IM protocol for many internal networks. This particular version of Jabber for Application Simulator currently only supports Google Talk. You can use the actions and action parameters in the table below to simulate a Google Talk conversation between a user and the client.

Jabber Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a Jabber login for a specific Jabber Service Provider.	Jabber Service	Google Talk
Client: Chat	Represents the number of Jabber conversations between a user and a peer.	Client Messages	0 – 4,294,967,295
		Peer Messages	0 – 4,294,967,295
Client: IM User	Simulates a message from the user to a peer.	Peer Name	Up to 32 alphanumeric and/or special characters can be defined for the peer's name.
		Message	Use alphanumeric and/or special characters to define the message sent from the user to the peer.

Action	Description	Action Parameter	Valid Values
Server: IM: Peer	Simulates a message from the peer to the user.	Peer Name	Up to 32 alphanumeric and/or special characters can be defined for the peer's name.
		Message	Use alphanumeric and/or special characters to define the message sent from the peer to the user.

LDAP Action Parameters

The table below lists the actions and action parameters for LDAP.

LDAP Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a random LDAP flow from the server to the client	None	N/A

MMS MM1 Action Parameters

Multimedia Message Service MM1 (MMS MM1) is the 3GPP interface between the MMS User Agent, which generally resides on the Mobile Station (MS), and the MMS Center (MMSC). MMS MM1 is used for sending and retrieving Multimedia Messages to and from the MMSC and for managing the subscriber's Multimedia Mailbox (MMBox) on the MMSC. This is essentially how cell phones send and retrieve picture or video messages.

The table below lists the actions and action parameters for the Multimedia Messaging Service MM1 Protocol (MMS MM1).

MMS MM1 Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Raw Request	Sends a raw request message.	Transaction Flag	Start, Continue, End, or Start and End
		URL	String value (0 – 128 character length)
		Body Data	filename
		HTTP Connection Header	String value (0 – 128 character length)
		HTTP Cookie2 Header	String value (0 – 128 character length)
		Header Data	filename
		HTTP Via Header	String value (0 – 128 character length)

Action	Description	Action Parameter	Valid Values
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Available Action	Delay, Raw Message, Update Dest Address, Update Dest Port, Update Receive Window, Verify File, Goto, Close, Fail, Log as Target, Add Flow Dictionary, Add Markov Flow Dictionary, Generate Characters
Client: Resolve	Resolve the specified host.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Host	Any host
		Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
		Use Response	true or false

Action	Description	Action Parameter	Valid Values
Server: Raw Response	Sends a raw response message.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Body Data	filename
		HTTP Connection Header	String value (0 – 128 character length)
		Header Data	filename
		HTTP Server Header	String value (0 – 128 character length)
Client: Raw Body Request	Sends a raw body request message.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		URL	String value (0 – 128 character length)
		Body Data	filename
		HTTP Connection Header	String value (0 – 128 character length)
		Content Type	Use up to 128 alphanumeric and/or special characters to define the value of the Content-Type header.
		HTTP Cookie2 Header	String value (0 – 128 character length)
		Delivery Report	true or false

Action	Description	Action Parameter	Valid Values
		First Content Type	String value (0 – 128 character length)
		From	String value (0 – 128 character length)
		Message Type	Use alphanumeric and/or special characters to provide the message type of the MMS message being referenced.
		MMS Version Code	N/A
		Start Marker	String value (0 – 128 character length)
		Subject	String value (0 – 128 character length)
		To	String value (0 – 128 character length)
		Transaction ID	0 – 65,536 or a token
		HTTP Via Header	String value (0 – 128 character length)
Server: Raw Body Response	Sends a raw body response message.	Transaction Flag	Start, Continue, End, or Start and End
		Body Data	filename
		HTTP Connection Header	String value (0 – 128 character length)
		Content Type	N/A

Action	Description	Action Parameter	Valid Values
		First Content Type	String value (0 – 128 character length)
		Message ID	String value (0 – 128 character length)
		Message Type	N/A
		MMS Version Code	N/A
		Response Status	N/A
		Response Text	String value (0 – 128 character length)
		HTTP Server Header	String value (0 – 128 character length)
		Start Marker	String value (0 – 128 character length)
		Transaction ID	String value (0 – 128 character length)

Action	Description	Action Parameter	Valid Values
Client: WSP Connect	Sends a WSP Connect message.	Transaction Flag	Start, Continue, End, or Start and End
		WTP Transaction ID	0 – 32,767
		WSP Client SDU Size	0 – 1,048,576
		WSP Server SDU Size	0 – 1,048,576
		WSP Method MOR	0 – 255
		WSP Push MOR	0 – 255
Server: WTP Ack	Sends a WTP Ack message.	Transaction Flag	Start, Continue, End, or Start and End
		WTP Transaction ID	0 – 32,767
Server: WSP ConnectReply	Sends a WSP ConnectReply message.	Transaction Flag	Start, Continue, End, or Start and End
		WTP Transaction ID	0 – 32,767
		WSP Client SDU Size	0 – 1,048,576
		WSP Server SDU Size	0 – 1,048,576
		WSP Method MOR	0 – 255
		WSP Push MOR	0 – 255

MSNP Action Parameters

The table below lists the actions and the action parameters available for MSNP.

MSNP Action Parameters

Action	Description	Action Parameter	Valid Values

MSSQL

The table below lists the action and action parameters for MSSQL.

MSSQL Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Login	Simulates a login to a Microsoft SQL server.	Username	Enter up to 14 alphanumeric and/or special characters to define the username used to log into the database.
		Password	Enter up to 14 alphanumeric and/or special characters to define the password used to log into the database.
		Server Name	Enter up to 63 alphanumeric and/or special characters to define the server name.
		Client Name	Enter up to 63 alphanumeric and/or special characters to define the client name.
Client: Query: Use Database	Executes a 'use database' statement.	Database Name	Enter up to 63 alphanumeric and/or special characters to define the database name.

Action	Description	Action Parameter	Valid Values
Client: Query: Select	Simulates an SQL query and response. The values entered for **Columns** and **Rows** will determine the number of columns and rows the query will return. If **Column Names** are specified, then they need to be entered as comma-delimited values.	SQL Query	Alphanumeric and/or special characters can be used to define the SQL query.
		Columns	0 – 4,294,967,295
		Rows	0 – 4,294,967,295
		Column Names	Enter up to 255 alphanumeric and/or special characters to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.

Multicast

The table below lists the action and action parameters for Multicast.

Multicast allows routers to work together to efficiently deliver copies of data to interested receivers. Instead of sending a separate copy of the data to each host, the server sends the data only once. Routers along the pathway to the clients make copies as needed.

Multicast Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Multicast Join	Causes the flow to request a join to the multicast group defined in the flow settings.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Delay	Pauses the flow for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
		Maximum Number of Milliseconds	1,000,000
Client: Multicast Leave	Causes the flow to request a join to the multicast group defined in the flow settings.	Transaction Flag	Start, Continue, End, or Start and End
Server: Send Random Data	Sends randomized data.	Transaction Flag	Start, Continue, End, or Start and End
		Minimum Size	
		Maximum Size	
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

MySQL

The MySQL authenticate action expects the first packet received to include the server greeting. This means that the server greeting must be the first action in a flow. When the greeting is not the first flow, MySQL will not proceed to the next action.

The table below lists the actions and action parameters for MySQL.

MySQL Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Authenticate	Simulates a login to the MySQL database.	Transaction Flag	Start, Continue, End, or Start and End
		Database Username	Up to 63 alphanumeric and/or special characters can be used to define the database username.
		Database Password	Up to 63 alphanumeric and/or special characters can be used to define the database password.
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Predefined	Select to match a predefined pattern.
		Available Action	Delay, Raw Message, Verify Rows, Goto, Close, Fail, Login, Login Request, Authenticate, Use Database, Query, Quit

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to a MySQL server.	Transaction Flag	Start, Continue, End, or Start and End
		Server Version String	
		Database Username	Enter up to 14 alphanumeric and/or special characters to define the username used to log into the database.
		Database Password	Enter up to 14 alphanumeric and/or special characters to define the password used to log into the database.
Client: Query: Use Database	Executes a 'use database' statement.	Database Name	Enter up to 63 alphanumeric and/or special characters to define the database name.
Server: Query: Use Database OK	Executes a 'use database' statement.	Transaction Flag	Start, Continue, End, or Start and End
Client: Quit	This action simulates a Quit command from a client to a MySQL server.	Transaction Flag	Start, Continue, End, or Start and End
Client: Query	Queries the MySQL Server for the host	Transaction Flag	Start, Continue, End, or Start and End
		SQL Statement	

Action	Description	Action Parameter	Valid Values
Server: Query Response	Sends a response to the client with the specified data.	Transaction Flag	Start, Continue, End, or Start and End
		Database Name	Enter up to 63 alphanumeric and/or special characters to define the database name.
		Table Name	Enter up to 255 alphanumeric and/or special characters to define the table name referenced in the SQL query. If the SQL query is random (or left blank), the table name will be consistent with the randomly generated FROM clause of the SQL statement.
		Columns	0 – 4,294,967,295
		Minimum Chars/Column	Enter up to 255 alphanumeric and/or special characters to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.
		Max Chars/Column	Enter up to 255 alphanumeric and/or special characters to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.
		Rows	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
		Column List	The columns to return, or alternatively, an *. If no value is provided, the system generates a random set of column names.
		System Value	

NetBIOS Action Parameters

The table below lists the actions and action parameters for NetBIOS.

NetBIOS Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Session Request	Sends a request to the server to start a session.	Transaction Flag	Start, Continue, End, or Start and End
		Client Name	An alphanumeric string that defines the client's name
		Server Name	An alphanumeric string that defines the server's name
Server: Positive Session Response	Sends an OK response to the client.	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Session Message	Sends a message to the server.	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Session Message	Sends a response message to the client.	Transaction Flag	Start, Continue, End, or Start and End

NFS Action Parameters

The table below lists the actions and action parameters for NFS.

NFS Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a NFS flow from the server to the client.	None	N/A

NNTP Action Parameters

The table below lists the actions and action parameters for NNTP.

NNTP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Generic Command	Sends an NNTP command	Transaction Flag	Start, Continue, End, or Start and End
		Keyword	Alphanumeric and/or special characters can be used to define the Keyword command.
		Variant	Alphanumeric and/or special characters can be used to define the Variant command.
		Argument 1	Alphanumeric and/or special characters can be used to define the first command argument.
		Argument 2	Alphanumeric and/or special characters can be used to define the second command argument.
		Argument 3	Alphanumeric and/or special characters can be used to define the third command argument.
		Argument 4	Alphanumeric and/or special characters can be used to define the fourth command argument.

Action	Description	Action Parameter	Valid Values
Server: Generic Response	Sends an NNTP response	Transaction Flag	Start, Continue, End, or Start and End
		Response Code	0 – 4,294,967,295
		Argument 1	Alphanumeric and/or special characters can be used to define the first command argument.
		Argument 2	Alphanumeric and/or special characters can be used to define the second command argument.
		Argument 3	Alphanumeric and/or special characters can be used to define the third command argument.
		Argument 4	Alphanumeric and/or special characters can be used to define the fourth command argument.

NTP Action Parameters

The table below lists the actions and action parameters for NTP.

NTP Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a NTP flow from the server to the client.	None	N/A

Oracle Action Parameters

The table below lists the actions and action parameters for Oracle.

Oracle Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Login	Simulates a login to the Oracle database using the values defined for the action parameters.	Database Username	Enter up to 14 alphanumeric and/or special characters to define the username used to log into the Oracle database.
		Database Password	Enter up to 14 alphanumeric and/or special characters to define the password used to log into the Oracle database.
		Server Name	Enter up to 63 alphanumeric and/or special characters to define the server name for the Oracle database.
		Database Name	Enter up to 63 alphanumeric and/or special characters to define the Oracle database name.
		Server Banner	Enter up to 128 alphanumeric and/or special characters to define the Oracle banner.
		Client Username	Enter up to 63 alphanumeric and/or special characters to define the Oracle workstation username.
Client: Login	Simulates a login to the Oracle database using the values defined for the action parameters.	Client Machine Name	Enter up to 63 alphanumeric and/or special characters to define the Oracle workstation name.

Action	Description	Action Parameter	Valid Values
Client: Query Select	Simulates an SQL query and response. The values entered for **Columns** and **Rows** will determine the number of columns and rows the query will return. If **Column Names** are specified, then they need to be entered as comma-delimited values.	SQL Query	Alphanumeric and/or special characters can be used to define the SQL query.
		Columns	0 – 4,294,967,295
		Rows	0 – 4,294,967,295
		Column Names	Enter up to 255 alphanumeric and/or special characters to define the names of the columns from which data will be returned. The information listed here must be comma-delimited format.

OSCAR Action Parameters

The table below lists the actions and the action parameters available for OSCAR.

OSCAR Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host.	Retry Interval (ms) – The amount of time to wait for a response before failing or retrying.	1 – 1,000,000
		Retries until Failure – The number of retries to attempt before failing.	0 – 7
		Use Response – Indicates if the response to this resolve attempt should be used for subsequent flows.	true or false
		Host – The host that will be queried.	Auth Server, Boss Server, Client, or DNS Server
Client: Connect (Login Request)	Simulates a connect request to an authentication server.	Transaction Flag	Start, Continue, End, or Start and End
Server: Connect Response	Server sign on response.	Transaction Flag	Start, Continue, End, or Start and End
Client: Sign On	Simulates a logon request to the authentication server.	Transaction Flag	Start, Continue, End, or Start and End
Server: Sign On Challenge	Simulates the reply to the authentication server's challenge by sending the screen name, password hash, and client details.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Sign On Challenge Reply	Simulates the reply to the authentication server's challenge by sending the screen name, password hash, and client details.	Transaction Flag	Start, Continue, End, or Start and End
Server: Disconnect (Login Response)	Simulates the message that an authentication server provides after a login request occurs. It provides the authentication cookie and the disconnect request in the same message.	Transaction Flag	Start, Continue, End, or Start and End
Client: Disconnect	Client disconnect request.	Transaction Flag	Start, Continue, End, or Start and End
Client: Binding Request	Sends a Binding Request to a STUN server.	Transaction Flag	Start, Continue, End, or Start and End
		Change IP Flag	True or False
Server: Binding Response	Send a Binding Response to a STUN client.	Transaction Flag	Start, Continue, End, or Start and End
		Mapped Address	String representing IP Address. A token may be used. Values that are not tokens or valid IP addresses will result in a random IP address.
		Mapped Port	String representing port number. A token may be used. Values that are not tokens or valid ports will result in 0.

Action	Description	Action Parameter	Valid Values
		Source Address	String representing IP Address. A token may be used. Values that are not tokens or valid IP addresses will result in a random IP address.
		Source Port	String representing port number. A token may be used. Values that are not tokens or valid ports will result in 0.
		Changed Address	String representing IP Address. A token may be used. Values that are not tokens or valid IP addresses will result in a random IP address.
		Changed Port	String representing port number. A token may be used. Values that are not tokens or valid ports will result in 0.
Server: Accept TLS	Accept a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.

Action	Description	Action Parameter	Valid Values
		Cipher	Any available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Client Common Name	N/A
		Client Cert Verify Mode	Do Not Check Cert, Allow Untrusted Cert, or Require Trusted Cert
		Server Certificate	Available PEM formatted cert file
		Server Private Key	Available PEM formatted key file
		Client CA Certificate	Available PEM formatted cert file
Client: Start TLS	Establish a TLS connection.	Enabled	true or false
		Min Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.

Action	Description	Action Parameter	Valid Values
		Max Version	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
		Cipher	An available ciphersuite
		Resume Max Reuse	0 – 4,294,967,295
		Resume Expire (seconds)	0 – 4,294,967,295
		Handshake Timeout (milliseconds)	0 – 4,294,967,295
		Client Authentication Enabled	true or false
		Server Common Name	N/A
		Server Cert Verify Mode	Do Not Check Cert, Allow Untrusted Cert, or Require Trusted Cert
		Client Certificate	Available PEM formatted cert file
		Client Private Key	Available PEM formatted key file
		Server CA Certificate	Available PEM formatted cert file
Client: Connect (Authenticated)	Simulates an authenticated connect request to a BOS server.	Transaction Flag	Start, Continue, End, or Start and End
Server: List Services	Lists server supported services.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Request ICQ Services	Requests ICQ services.	Transaction Flag	Start, Continue, End, or Start and End
Server: Provide ICQ Services	Provides ICQ services.	Transaction Flag	Start, Continue, End, or Start and End
Client: Rate Limit Request	Client requests rate limits.	Transaction Flag	Start, Continue, End, or Start and End
Server: Rate Limit Response	Server provides rate limits.	Transaction Flag	Start, Continue, End, or Start and End
Client: Rate Limit Acknowledgment	Client acknowledges rate limit information received from server.	Transaction Flag	Start, Continue, End, or Start and End
Client: Rights Request	Client rights request. Client provides timestamp of last contact list update.	Transaction Flag	Start, Continue, End, or Start and End
Server: Rights Response	Server rights response. The BOSS server provides the contact list in this response.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Buddies	0 – 127
Client: Client Activate	Informs the server that the client has accepted the SSI information (buddy list, rate limits, capabilities, etc.) and that it is ready to proceed.	Transaction Flag	Start, Continue, End, or Start and End
Server: Buddy Presence Event	Enacts the notification that is sent to a client when a buddy comes online or goes offline.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
		Event Type	Buddy Offline or Buddy Online

Action	Description	Action Parameter	Valid Values
Server: IM User	Simulates an incoming message event that is sent from the BOSS server to the OSCAR client.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
		Static Message Text	0 – 4096 characters
		Language	Custom, English, French, German, Italian, Spanish
		Message Wordcount Min	0 – 1,048,576
		Message Wordcount Max	0 – 1,048,576
		Keyword List	0 – 4096 characters
		Keywords in Subject	true or false
		Static Message Text File	Available file
		Custom Dictionary	Available file

Action	Description	Action Parameter	Valid Values
Client: IM Peer	Simulates an outgoing message event that is sent from the OSCAR client to the BOSS server.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
		Static Message Text	0 – 4096 characters
		Language	Custom, English, French, German, Italian, Spanish
		Message Wordcount Min	0 – 1,048,576
		Message Wordcount Max	0 – 1,048,576
		Keyword List	0 – 4096 characters
		Keywords in Subject	true or false
		Static Message Text File	Available file
		Custom Dictionary	Available file
Client: Send FIle: Peer	Simulates a file transfer request that is sent from the OSCAR client to the BOSS server.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
		File Name	0 – 63 characters
Server: Accept File: Peer	Simulates a file transfer response from the peer, accepting the file transfer.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters

Action	Description	Action Parameter	Valid Values
Client: Send File	Send a file to a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Client Username	0 – 16 characters
		File Name	0 – 63 characters
		File Minsize	0 – 52,428,800
		File Maxsize	0 – 52,428,800
		File Data	Available file
Server: Send File: User	Simulates a file transfer request that is sent from the BOSS server to the OSCAR client on behalf of the remote chat peer.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
		File Name	0 – 63 characters
Client: Accept File: Peer	Simulates a file transfer response from the peer, accepting the file transfer.	Transaction Flag	Start, Continue, End, or Start and End
		Buddy Screen Name	0 – 127 characters
Client: Receive File	Receive a file from a peer.	Transaction Flag	Start, Continue, End, or Start and End
		Client Username	0 – 16 characters
		File Name	0 – 63 characters
		File Minsize	0 – 52,428,800
		File Maxsize	0 – 52,428,800
		File Data	Available file
Client: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false
Server: TLS Close Notify	Send a TLS Close Notify alert.	Enabled	true or false

Pandora Action Parameters

The table below lists the actions and the action parameters available for Pandora.

Pandora Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Resolve the specified host.	Retry Interval (ms) – The amount of time to wait for a response before failing or retrying.	1 – 1,000,000
		Retries until Failure – The number of retries to attempt before failing.	0 – 7
		Use Response – Indicates if the response to this resolve attempt should be used for subsequent flows.	true or false
		Host – The host that will be queried.	Any host
Client: Client Hello	Initiates a TLS handshake with the server.	Transaction Flag	Start, Continue, End, or Start and End
		Session ID	0 – 48 characters
Server: Server Hello	Continues the TLS handshake with the client.	Transaction Flag	Start, Continue, End, or Start and End
		Cipher Suite	Supported Cipher Suite
Server: Certificates	The server sends a list of certificates for the client to verify.	Transaction Flag	Start, Continue, End, or Start and End
		Subject: Common Name	Alphanumeric and/or special characters can be used to define the common name in the certificate.

Action	Description	Action Parameter	Valid Values
		Subject: Country Code	Alphanumeric and/or special characters can be used to define the country code in the certificate.
		Subject: State/Province	Alphanumeric and/or special characters can be used to define the state/province in the certificate.
		Subject: City/Locality	Alphanumeric and/or special characters can be used to define the city/locality in the certificate.
		Subject: Organization	Alphanumeric and/or special characters can be used to define the organization in the certificate.
		Issuer: Common Name	Alphanumeric and/or special characters can be used to define the common name in the certificate.
		Issuer: Country Code	Alphanumeric and/or special characters can be used to define the country code in the certificate.
		Issuer: State/Province	Alphanumeric and/or special characters can be used to define the state/province in the certificate.

Action	Description	Action Parameter	Valid Values
		Issuer: City/Locality	Alphanumeric and/or special characters can be used to define the city/locality in the certificate.
		Issuer: Organization	Alphanumeric and/or special characters can be used to define the organization in the certificate.
		Number of random certificates	1 – 100
		Upload an x.509 certificate	Any available certificate
Server: Hello Done	Signals the end of the transaction negotiation	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Key Exchange	Sends the premastersecret encrypted with the server's public key	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Change Cipher Spec	Instructs the server that all subsequent data will be encrypted using the connection settings that were negotiated	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Encrypted Handshake Message	The client sends a handshake message that occurs after the key exchange has occurred.	Transaction Flag	Start, Continue, End, or Start and End
Server: Server Encrypted Handshake Message	The server sends a handshake message that occurs after the key exchange has occurred.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Server: Server Change Cipher Spec	This message instructs the client that all subsequent server data will be encrypted according to the parameters that were negotiated.	Transaction Flag	Start, Continue, End, or Start and End
Client: Client Application Data (Encrypted)	Sends a block of encrypted data.	Transaction Flag	Start, Continue, End, or Start and End
		Minimum data length	1 – 16,383
		Maximum data length	1 – 16,383
Server: Server Application Data (Encrypted)	Sends a block of encrypted data.	Transaction Flag	Start, Continue, End, or Start and End
		Minimum data length	1 – 16,383
		Maximum data length	1 – 16,383
Client: Get Radio	Client requests access to Pandora radio.	Transaction Flag	Start, Continue, End, or Start and End
Server: Radio OK	Server sends an OK response.	Transaction Flag	Start, Continue, End, or Start and End
Client: Get Friends	Client requests a friends list from the server.	Transaction Flag	Start, Continue, End, or Start and End
Server: Friends OK	Server returns a friends list.	Transaction Flag	Start, Continue, End, or Start and End
Client: Get Stations		Transaction Flag	Start, Continue, End, or Start and End
Server: Stations OK	The server responds with a list of preset stations.	Transaction Flag	Start, Continue, End, or Start and End
		Seed Artists. A comma-separated list of artists to populate the preset stations with.	0 – 2048 characters

Action	Description	Action Parameter	Valid Values
Client: Switch Station	Switch to a different station within Pandora.	Transaction Flag	Start, Continue, End, or Start and End
		Artist. Name of the artist.	0 – 1024 characters
Server: Switch Station OK	Server approves the station change and acknowledges it.	Transaction Flag	Start, Continue, End, or Start and End
Client: Get Fragment	Client requests access to song information.	Transaction Flag	Start, Continue, End, or Start and End
Server: Fragment OK	Server responds with a fragment.	Transaction Flag	Start, Continue, End, or Start and End
		Artist. Name of the artist.	0 – 1024 characters
		Album. Name of the album.	0 – 1024 characters
		Song. Name of the song.	0 – 1024 characters
		Genre. The song's genre.	0 – 1024 characters
Client: Get Song	Client requests song data.	Transaction Flag	Start, Continue, End, or Start and End
Server: Song OK	Server responds with song data.	Transaction Flag	Start, Continue, End, or Start and End

POP3-Advanced Action Parameters

POP3 is an application protocol that allows clients to retrieve e-mail from a server over a TCP/IP connection. Using the actions and action parameters listed in the table below, you can create a POP3 flow that simulates a client logging into the POP3 mail server and retrieving an e-mail message.

For the **Client: Retrieve Mail** action, there are several action parameters that you can use to compose the message part of the e-mail. **Raw Message URL** allows you to import your own message URL. If you opt to do this, the contents of the file will comprise the entire message – including the headers. If no file is selected for the Raw Message URL, then Application Simulator will use the values input for **From:**, **To:**, **Subject:**, **Attachment filename**, and **Attachment size** to generate random content for the message. If you have chosen to use your own file for the attachment URL, then this file will be incorporated into the message. However, if no attachment URL is selected, then Application Simulator will use the values input for **Attachment filename** and **Attachment size**.

Note: All action parameters that are left blank will generate random values. Some action parameters will allow you to use '0' to utilize random value generation; these action parameters will have **(0 == random)** listed next to them.

POP3 Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to a POP3 server.	Username	Enter up to 128 alphanumeric and/or special characters to define the username used to log into the POP3 server.
		Password	Enter up to 128 alphanumeric and/or special characters to define the password used to log into the POP3 server.
Client: Retrieve Mail	Simulates the retrieval of an e-mail message.	Raw Message URL	Select a file from the **Raw Message URL** drop-down menu. The contents of this file will comprise the entire mail message – including the headers. If the desired file is not listed on the menu, use the **Import Raw Message URL** link to upload a file that will be used for the entire URL message.

Action	Description	Action Parameter	Valid Values
Client: Retrieve Mail	Simulates the retrieval of an e-mail message.	From:	Enter up to 128 alphanumeric and/or special characters to define the **From:** field.
		To:	Enter up to 128 alphanumeric and/or special characters to define the **To:** field.
		Subject:	Enter up to 128 alphanumeric and/or special characters to define the **Subject:** field.
		Text:	Enter up to 128 alphanumeric and/or special characters to define the body of the message.
		Attachment URL	Select a file from the **Attachment URL** drop-down menu. This file will be incorporated into the generated mail message. If the desired file is not listed on the menu, use the **Import Attachment URL** link to upload the attachment file that will be used.
		Attachment filename	Enter up to 256 alphanumeric and/or special characters to define the attachment filename.
		Attachment size	0 – 4,294,967,295

PostgreSQL Action Parameters

The table below lists the actions and action parameters for PostgreSQL.

PostgreSQL Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a PostgreSQL flow from the server to the client.	None	N/A

Quote of the Day Action Parameters

The table below lists the actions and action parameters for Quote of the Day.

Quote of the Day Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Prompting Message	Sends a prompting message to start the session	Transaction Flag	Start, Continue, End, or Start and End
		Prompt	Alphanumeric and/or special characters can be used to define the prompting message that is sent at the beginning of a UDP session.
Server: Quote of the Day Message	Sends the quote to the client	Transaction Flag	Start, Continue, End, or Start and End
		Quote	Alphanumeric and/or special characters can be used to define the quote that is sent by the server.

RADIUS Access Action Parameters

The table below lists the actions and action parameters for RADIUS Access.

RADIUS Access Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Request	Sends a request for access to the RADIUS server.	Transaction Flag	Start, Continue, End, or Start and End
		Identifier	
		Username	Alphanumeric and/or special characters can be used to define the username in the request.
		Password	
		Password Type	
		NAS IP Address	x.x.x.x, where x is a value between 0 –255
		NAS Port	0 – 256
		Calling Station ID	0 – 253 characters
		Called Station ID	0 – 253 characters
Server: Challenge	Sends a request for additional information from the user.	Transaction Flag	Start, Continue, End, or Start and End
		Identifier	0 – 253 characters

Action	Description	Action Parameter	Valid Values
Server: Accept	Allows the user access to network resources.	Transaction Flag	Start, Continue, End, or Start and End
		Client Flow	An integer expressing the flow ID as it appears in the Super Flow screen.
		Identifier	0 – 253 characters
		Username	Alphanumeric and/or special characters can be used to define the username being accepted.
		Framed IP Address	x.x.x.x, where x is a value between 0 –255
		Framed Netmask	x.x.x.x, where x is a value between 0 –255
Server: Reject	Rejects the user's request for access to network resources.	Transaction Flag	Start, Continue, End, or Start and End
		Identifier	0 – 253 characters
		Username	Alphanumeric and/or special characters can be used to define the username being rejected.

RADIUS Accounting Action Parameters

The table below lists the actions and action parameters for RADIUS Accounting.

RADIUS Accounting Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Request	Sends a request for access to the RADIUS server.	Transaction Flag	Start, Continue, End, or Start and End
		Identifier	
		Username	Alphanumeric and/or special characters can be used to define the username being accepted.
		Password	

Action	Description	Action Parameter	Valid Values
		Password Type	
		NAS IP Address	x.x.x.x, where x is a value between 0 –255
		NAS Port	0 – 256
		Calling Station ID	0 – 253 characters
		Called Station ID	0 – 253 characters
Server: Response	Sends the server's response to the client.	Transaction Flag	Start, Continue, End, or Start and End
		HTTP Compression	deflate, gzip, or none
		Enable persistent HTTP sessions	on or off
		Enable Content-MD5	on or off
		Enable chunked encoding	on or off
		HTTP chunk size	N/A
		Content-Type	N/A
		File Generator	N/A

Action	Description	Action Parameter	Valid Values
		File Generator Padding	N/A
		File Generator Exact Length	N/A
		String for response data	N/A
		Random response min length	0 – 4,294,967,295
		Random response max length	0 – 4,294,967,295
		Expires	N/A
		Last-Modified	N/A
		Cache-Control	N/A
		Date	N/A
		ETag	N/A
		Cookie Name	N/A
		Cookie Value	N/A
		Customer Header Name	N/A
		Customer Header Value	N/A

Action	Description	Action Parameter	Valid Values
		File for response data	N/A
		Custom Headers File	N/A
Server: Accept	Allows the user access to network resources.	Transaction Flag	Start, Continue, End, or Start and End
		Client Flow	An integer expressing the flow ID as it appears in the Super Flow screen.
		Identifier	0 – 253 characters
		Username	Alphanumeric and/or special characters can be used to define the username being accepted.
		Framed IP Address	x.x.x.x, where x is a value between 0 –255
		Framed Netmask	x.x.x.x, where x is a value between 0 –255
Client: GET	Performs a GET request for the specified URL.	Transaction Flag	Start, Continue, End, or Start and End
		Proxy Mode	on or off
		Request path	Use up to 128 alphanumeric and/or special characters to define the URL that is requested in the RADIUS method.
		URL escape	true or false
		Enable persistent HTTP sessions	on or off
		Custom Accept Header	Use up to 128 alphanumeric and/or special characters to define the Accept header. This data will override default values used in the Accept header.

Action	Description	Action Parameter	Valid Values
		Custom Encoding Header	Use up to 128 alphanumeric and/or special characters to define the Encoding header. This data overrides values used in the Accept-Encoding header.
		Custom Language Header	Use up to 128 alphanumeric and/or special characters to define the Custom Language header. This data will override default values used in the Accept-Language header.
		Custom User-Agent	Use up to 128 alphanumeric and/or special characters to define the User-Agent field. This data will override default values used in the User-Agent header.
		Custom 'If-None-Match'	N/A
		Cookie Name	N/A
		Cookie Value	N/A
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
		Custom Headers File	Any available file

RIPv1 Action Parameters

The table below lists the actions and action parameters for RIPv1.

RIPv1 Action Parameters

Action	Description	Action Parameter	Valid Values
Client: RIPv1 Request	Sends an RIPv1 Request.	Command Field Value	An 8-bit integer value. (1 = Request, 2 = Response)
		Address Family Field Value	A 16-bit integer value. (2 = IP)
		Version Field Value	An 8-bit integer value
		IP/Metric Generation Method	Comma Separated, Uploaded File
		Comma Separated Groups of IP: Metric	IP addresses in the format: [IP]:[METRIC], [IP]:[METRIC]...
		Transaction Flag	Start, Continue, End, or Start and End
		IPs and Metrics File	IP addresses in the format: [IP]:[METRIC] [IP]:[METRIC]
Server: RIPv1 Response	Sends an RIPv1 Response.	Command Field Value	An 8-bit integer value. (1 = Request, 2 = Response)
		Address Family Field Value	A 16-bit integer value. (2 = IP)
		Version Field Value	An 8-bit integer value
		IP/Metric Generation Method	Comma Separated, Uploaded File
		Comma Separated IP: Metric	IP addresses in the format: [IP]:[METRIC], [IP]:[METRIC]...
		Transaction Flag	Start, Continue, End, or Start and End
		IPs and Metrics File	IP addresses in the format: [IP]:[METRIC] [IP]:[METRIC]

Rlogin Action Parameters

The table below lists the actions and action parameters for Rlogin.

Rlogin Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Client Connection	Initiates a connection to the server using the connection settings specified	Transaction Flag	Start, Continue, End, or Start and End
		Local Username	Enter alphanumeric and/or special characters to define the username for the client host
		Server Username	Enter alphanumeric and/or special characters to define the username for the server host
		Terminal Type	Enter alphanumeric and/or special characters to define the type of terminal that will be used for the connection
		Terminal Speed	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
Server: Server Connection Response	Sends a null byte to the client to acknowledge the connection	Transaction Flag	Start, Continue, End, or Start and End
Server: Password Prompt	Prompts the user for their password	Transaction Flag	Start, Continue, End, or Start and End
		Password Prompt	Enter alphanumeric and/or special characters to define the password prompt displayed to the client
Server: Shell Prompt	Displays the shell prompt from the server	Transaction Flag	Start, Continue, End, or Start and End
		Shell Prompt	Enter alphanumeric and/or special characters to define the shell prompt displayed to the client
Client: Shell Command	Issues a command to the server and can be used to send an arbitrary bytestream to the server	Transaction Flag	Start, Continue, End, or Start and End
		Shell Command	Enter alphanumeric and/or special characters to define the shell command sent to the server
Client: Send String	Sends a string to the Rlogin server	Transaction Flag	Start, Continue, End, or Start and End
		String	Enter alphanumeric and/or special characters to define the string sent to the remote server

Action	Description	Action Parameter	Valid Values
Server: Shell Reply	Sends the shell's reply to a command sent by the client	Transaction Flag	Start, Continue, End, or Start and End
		Shell Reply	Enter alphanumeric and/or special characters to define the shell reply sent to the client
Server: Flush Output Command	Sends a command to the client the remove, or "flush" all buffered output	Transaction Flag	Start, Continue, End, or Start and End
Server: Flow Control Command	Sends a command to the client to stop honoring flow control	Transaction Flag	Start, Continue, End, or Start and End
Server: Window Size Command	Sends a request for the client's window size	Transaction Flag	Start, Continue, End, or Start and End
Client: Window Size Response	Responds to the server's request for the client's window size	Transaction Flag	Start, Continue, End, or Start and End
		Rows	0 – 4,294,967,295
		Columns	0 – 4,294,967,295
		X Pixels	0 – 4,294,967,295
		Y Pixels	0 – 4,294,967,295

RPC Bind

RPC BIND is used to interface with a system's portmapper, so applications can register with the local portmapper to inform it of what port it listens on, and remote systems can query the portmapper to find registered applications.

You can use the actions and action parameters listed in the table below to set up an RPC Bind flow.

RPC Bind Action Parameters

Action	Description	Action Parameter	Valid Values
Client: GetAddr Call	Contacts the server machine to determine the address where RPC requests should be sent.	Transaction Flag	Start, Continue, End, or Start and End
		Program	Enter up to 50 alphanumeric and/or special characters to define the remote program for which a port is being mapped.
		Program Version	Enter up to 50 alphanumeric and/or special characters to define the version of the remote program.
		Network ID	Enter up to 50 alphanumeric and/or special characters to define the network ID for the network ID. The default value is 'UDP'.
Server: GetAddr Reply	Replies to the client with the universal address.	Transaction Flag	Start, Continue, End, or Start and End
		Universal Address	Enter up to 50 alphanumeric and/or special characters to define the universal address that will be returned by the server.

Rsync Action Parameters

The table below lists the action and action parameters for Rsync.

Rsync Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds.If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Initialization	Provides client identification to the server	Transaction Flag	Start, Continue, End, or Start and End
Server: Initialization	Provides server identification to the client	Transaction Flag	Start, Continue, End, or Start and End
Client: Module Query	Sends a query for a list of available Rsync modules to the server	Transaction Flag	Start, Continue, End, or Start and End
Server: Module List	Sends a list of available Rsync modules to the client	Transaction Flag	Start, Continue, End, or Start and End
Server: Disconnect	Disconnects the server from the client	Transaction Flag	Start, Continue, End, or Start and End

RTP Unidirectional Stream Action Parameters

The table below lists the actions and action parameters for RTP Unidirectional Stream.

RTP Unidirectional Stream Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Stream	Sends an RTP stream.	None	N/A

RTSP Action Parameters

The table below lists the actions and action parameters for RTSP.

RTSP Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a RTSP flow from the server to the client.	None	N/A

SCCP Action Parameters

The table below lists the actions and action parameters for SCCP.

SCCP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Off hook message	Off hook message	Transaction Flag	Start, Continue, End, or Start and End
		Line instance	0 – 4,294,967,295
		Call identifier	0 – 4,294,967,295
Server: Start tone message	Starts the tone	Transaction Flag	Start, Continue, End, or Start and End
		Tone. Which tone to play	0 – 4,294,967,295
		Tone (override). Manual entry for testing undocumented values.	0 – 4,294,967,295
		Line instance	0 – 4,294,967,295
		Call identifier	0 – 4,294,967,295

Action	Description	Action Parameter	Valid Values
Client: Keypad button message	The keypad button message.	Transaction Flag	Start, Continue, End, or Start and End
		Keypad button. The button pressed on the phone.	Choose one of the button types from the drop-down list.
		Keypad button (override). Manual entry for testing undocumented values.	0 – 4,294,967,295
		Line instance	0 – 4,294,967,295
		Call identifier	0 – 4,294,967,295
Server: Display text message	The display text message.	Transaction Flag	Start, Continue, End, or Start and End
		Display message. The message displayed on the phone.	0 – 33 characters
Server: Stop tone message	Stop tone message.	Transaction Flag	Start, Continue, End, or Start and End
		Line instance	0 – 4,294,967,295
		Call identifier	0 – 4,294,967,295
Client: On hook message	On hook message.	Transaction Flag	Start, Continue, End, or Start and End
		Line instance	0 – 4,294,967,295
		Call identifier	0 – 4,294,967,295

SIP Call Action Parameters

The table below lists the actions and action parameters for SIP.

SIP Call Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Invite	Establishes a session.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters
		Session Username	The login of the originating host
		Session Name	The name of the session
		Session Start	The start time of the session represented as seconds since 1900
		Session Stop	The stop time of the session represented as seconds since 1900
		Media Type	A string describing the type of media
		Media Protocol	A string describing media protocol
		Media Payload	A string describing the media payload
		Media Clock Rate	The media clock rate from 1 to 102,400
		Custom SDP Attributes	A CRLF delimited file that contains attributes that are to be appended

Action	Description	Action Parameter	Valid Values
Server: Proxy Auth	Sends a Proxy Authentication response.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters
Server: Trying	Sends a TRYING message to acknowledge the call request.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters

Action	Description	Action Parameter	Valid Values
Server: Ringing	Sends a RINGING response to the caller.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters
Client: OK	Sends an OK response.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters

Action	Description	Action Parameter	Valid Values
Server: OK	Sends an OK response.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters
		Session Username	The login of the originating host
		Session Name	The name of the session
		Session Start	The start time of the session represented as seconds since 1900
		Session Stop	The stop time of the session represented as seconds since 1900
		Media Type	A string describing the type of media
		Media Protocol	A string describing the media protocol
		Media Payload	A string describing the media payload
		Media Clock Rate	The media clock rate from 1 to 102,400
		Custom SDP Attributes	A CRLF delimited file that contains attributes that are to be appended

Action	Description	Action Parameter	Valid Values
Client: ACK	Sends an ACK.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters
Server BYE	Sends a BYE message to end the session.	Transaction Flag	Start, Continue, End, or Start and End
		Custom Header Name	Use up to 1,024 alphanumeric and/or special characters to define the custom header name.
		Custom Header Value	Use up to 1,024 alphanumeric and/or special characters to define the custom header value.
		Caller Tag	Up to 16 alphanumeric and/or special characters
		Recipient Tag	Up to 16 alphanumeric and/or special characters

Skype Call Action Parameters

The table below lists the actions and action parameters for Skype.

Skype Call Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Login	Login to Skype server.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Call Establishment	Initial packet exchange immediately before voice packets.	Transaction Flag	Start, Continue, End, or Start and End
Client: Voice Data	Exchange call data with callee.	Transaction Flag	Start, Continue, End, or Start and End
Client: Call Teardown	Final packet exchange immediately after voice packets.	Transaction Flag	Start, Continue, End, or Start and End
Client: Search	Query the supernode for a list of nodes to query for users.	Transaction Flag	Start, Continue, End, or Start and End
Client: Query Nodes	Query nodes in search of Skype users.	Transaction Flag	Start, Continue, End, or Start and End

SMB Action Parameters

The table below lists the actions and action parameters for SMB.

SMB Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Session Request		Transaction Flag	Start, Continue, End, or Start and End
		Client Name	Up to 128 alphanumeric and/or special characters can be used to define the client's name.
		Server Name	Up to 128 alphanumeric and/or special characters can be used to define the server's name.
Server: Positive Session Response	Sends an OK response to the client.	Transaction Flag	Start, Continue, End, or Start and End
Client: Negotiate Request	Sends a request to start an SMB session.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Server: Negotiate Response	Negotiates the protocol used to login into the remote host.	Transaction Flag	Start, Continue, End, or Start and End
Client: Session Setup Clear Request	Requests that the set up request is accepted.	Transaction Flag	Start, Continue, End, or Start and End
Client: Session Setup NTLMv1 Request	Sends a request for NTLMv1 authentication.	Transaction Flag	Start, Continue, End, or Start and End
		Challenge Key	A string value of alphanumeric and/or special characters.
Client: Session Setup NTLMv2 Negotiate Request	Sends a request for NTLMv2 authentication.	Transaction Flag	Start, Continue, End, or Start and End
Server: Session Setup NTLMv2 Response Challenge	Sends a request to the client for additional information.	Transaction Flag	Start, Continue, End, or Start and End
Client: Session Setup NTLMv2 Negotiate Request Authenticate	Sends a request for NTLMv2 authentication.	Transaction Flag	Start, Continue, End, or Start and End
Server: Session Setup NTLMv2 Response Denied	Denies the request for NTLMv2 authentication.	Transaction Flag	Start, Continue, End, or Start and End
Server: Session Setup NTLMv2 Response Success	Sends a SUCCESS response.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Tree Connect Request	Sends a request to connect to a tree.	Transaction Flag	Start, Continue, End, or Start and End
		Tree Name	A string value consisting of alphanumeric and/or special characters.
		Tree Password	A string value consisting of alphanumeric and/or special characters.
Server: Tree Connect Response	Sends a response to the tree connect request.	Transaction Flag	Start, Continue, End, or Start and End
		Service	A string value consisting of alphanumeric and/or special characters.
Client: Find First 2 Request	Finds the first two requests made by the client	Transaction Flag	Start, Continue, End, or Start and End
Server: Find First 2 Response	Finds the first two responses sent by the server	Transaction Flag	Start, Continue, End, or Start and End
Client: Query Path Info Request	Sends a query to request a path to the tree.	Transaction Flag	Start, Continue, End, or Start and End
Server: Query Path Info Response	Sends a response containing the path to the path request.	Transaction Flag	Start, Continue, End, or Start and End
Client: NT Create Request	Sends an NT create request.	Transaction Flag	Start, Continue, End, or Start and End
Server: NT Create Response	Sends an NT create response.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Read Request	Sends a READ request.	Transaction Flag	Start, Continue, End, or Start and End
		File ID	0 – 4,294,967,295
		File Byte Offset	0 – 4,294,967,295
		File Read Size	0 – 4,294,967,295
Server: Read Response	Sends a READ response.	Transaction Flag	Start, Continue, End, or Start and End
		File ID	0 – 4,294,967,295
		File Bytes Remaining	0 – 4,294,967,295
		File Read Size	0 – 4,294,967,295
		Compact Mode	0 – 4,294,967,295
		Simulation Server File Data URL	An uploaded file
Client: NULL Session	Sends a request for an unauthenticated connection to the SMB server.	Transaction Flag	Start, Continue, End, or Start and End
Client: Close an SMB session		Transaction Flag	Start, Continue, End, or Start and End
Server: Close an SMB session		Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Client File Download Session	Downloads a file from the specified URL.	Transaction Flag	Start, Continue, End, or Start and End
		Challenge Key	A string value consisting of alphanumeric and/or special characters.
		Tree Name	A string value consisting of alphanumeric and/or special characters.
		Tree Password	A string value consisting of alphanumeric and/or special characters.
		Service	A string value consisting of alphanumeric and/or special characters.
		File ID	0 – 4,294,967,295
		File Read Size	0 – 4,294,967,295
		Compact Mode	0 – 4,294,967,295
		Simulation Server File Data URL	An uploaded file

Action	Description	Action Parameter	Valid Values
Authenticate		Transaction Flag	Start, Continue, End, or Start and End
		Protocol Version	SMB or SMB2 Mandatory Parameter
		Share Name	The SMB or SMB2 Share Name to authenticate against.
		User Name	A local User Name on the remote system that has the appropriate permissions to access the SMB or SMB2 Share Name.
		Password	The password for the local User Name on the remote system that has appropriate permissions to access the SMB or SMB2 Share Name.
Write file to share	Writes the File Name to the SMB or SMB2 Share Name, superseding the file if it already exists.	Transaction Flag	Start, Continue, End, or Start and End
		File Name	The File Name to be appended on the SMB or SMB2 Share Name.
		Minimum Random Filesize	1
		Maximum Random Filesize	100,000,000
		File Contents	Random file or a file that you upload.

Action	Description	Action Parameter	Valid Values
Append to file on share	Appends the File Contents to the end of File Name on the SMB or SMB2 Share Name, opening the file if it already exists or creating a new file if it does not exist.	Transaction Flag	Start, Continue, End, or Start and End
		File Name	The File Name to be appended on the SMB or SMB2 Share Name.
		Minimum Random Filesize	1
		Maximum Random Filesize	100,000,000
		File Contents	Random file or a file that you upload.
Verify file from share	Verifies the File Contents against data read from File Name on SMB or SMB2 Share Name.	Transaction Flag	Start, Continue, End, or Start and End
		File Name	The File Name to be appended on the SMB or SMB2 Share Name.
		File Contents	Random file or a file that you upload.
Delete file from share	Deletes the File Name from the SMB or SMB2 Share Name.	Transaction Flag	Start, Continue, End, or Start and End
		File Name	The File Name to be appended on the SMB or SMB2 Share Name.
Disconnect	Disconnect from the SMB or SMB2 Share Name and logoff local User Name. This should be the last action.	Transaction Flag	Start, Continue, End, or Start and End

SMTP Action Parameters

The table below lists the actions and action parameters for SMTP.

SMTP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Resolve	Sends the SMTP query	Host	The host that will be queried
		Retry Interval (ms)	1 – 1,000,000
		Retries until Failure	0 – 7
Client: Client connect	Connect to the server and wait for the 220 banner message. This action does not send any SMTP data packets. It allows the Application Simulator to perform TCP setup for the SMTP flow at the point at which the action is included.	Transaction Flag	Start, Continue, End, or Start and End
Server: Server connected	Send a 220 message and banner to the client.	Transaction Flag	Start, Continue, End, or Start and End
Client: Send EHLO	Sends an EHLO greeting to the server.	Transaction Flag	Start, Continue, End, or Start and End
Server: Server 250 Hello	Send a 250 Server greeting message.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
Client: Send FROM	Sends a FROM command to the server.	Transaction Flag	Start, Continue, End, or Start and End
		Use AUTH FROM	True or False
		Constant Username	Sender name
		Domain	Domain name
		Username Range	True or False
		Username Prefix	Up to 128 alphanumeric and/or special characters
		Username Range Start	Up to 5 alphanumeric and/or special characters
		Username Range End	Up to 5 alphanumeric and/or special characters
		FROM Response	Accepted or Unaccepted
Server: Send OK	Respond to client FROM message with OK.	Transaction Flag	Start, Continue, End, or Start and End
Client: Send RCPT	Sends an RCPT command to the server.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameter	Valid Values
		Coalesce addresses into single RCPT message	True or False
		Constant RCPT Username	Recipient username
		RCPT Cc Username	User who will receive a Carbon Copy of the message
		RCPT Bcc Username	User who will receive a Blind Carbon Copy of the message
		Use Local Domain	True or False
		RCPT Domain	Recipient domain
		RCPT Username Range	True or False
		RCPT Username Prefix	Up to 128 alphanumeric and/or special characters
		RCPT Username Range Start	Up to 5 alphanumeric and/or special characters
		RCPT Username Range End	Up to 5 alphanumeric and/or special characters
		RCPT Response	True or False

Action	Description	Action Parameter	Valid Values
Client: Send DATA	This action begins a DATA transaction. No message data is passed in this message. See Message Data.	Transaction Flag	Start, Continue, End, or Start and End
Server: 354 Start	Send a 354 START message to the client.	Transaction Flag	Start, Continue, End, or Start and End
Client: Message Data	The actual body of the message to be sent by the client.	Transaction Flag	Start, Continue, End, or Start and End
		DATA Sequence – If only partial communication is required, select the appropriate configuration under "DATA Sequence."	True or False
		Use SMTP FROM in Envelope	True or False
		Envelope From Header	Up to 256 alphanumeric and/or special characters
		Use RCPT TO in Envelope	True or False
		Envelop To Header	Recipient address
		Subject	Up to 256 alphanumeric and/or special characters
		Mbox Message – If an "Mbox Message" resource file is selected, that file will constitute the entire message, including headers.	A file that contains the mail message

Action	Description	Action Parameter	Valid Values
		Mbox File – If an "Mbox File" is provided, each session will use a message randomly chosen from that mbox file, overriding other options. (Note that mbox files must conform to the "mboxrd" file format standard to be processed correctly.)	A file that contains multiple messages
		Text Content-Type – If an attachment is selected, or if the "Text Content-Type" and "Text Transfer-Encoding" options are selected (as they are by default), then a MIME-Encoded multipart/mixed message will be generated.	String describing the content type
		Text Transfer-Encoding – If an attachment is selected, or if the "Text Content-Type" and "Text Transfer-Encoding" options are selected (as they are by default), then a MIME-Encoded multipart/mixed message will be generated.	An enumerated list of choices
		Static Message Text File – If "Static Message Text File" or "Static Text" is chosen for an e-mail message, choices for keywords and word counts will be ignored. Otherwise, a random message will be generated, using the specified language word list and, if provided, all the words from a comma-delimited list of Keywords. If a Custom Dictionary is selected, the supplied dictionary file should be newline delimited, with one word (or phrase) per line.	A file containing the data used to generate random text
		Static Message Text – If "Static Message Text File" or "Static Text" is chosen for an e-mail message, choices for keywords and word counts will be ignored. Otherwise, a random message will be generated, using the specified language word list and, if provided, all the words from a comma-delimited list of Keywords. If a Custom Dictionary is selected, the supplied dictionary file should be newline delimited, with one word (or phrase) per line.	A string containing the data used to generate random text

Action	Description	Action Parameter	Valid Values
		Language	Custom, English, French, German, Italian, Spanish
		Custom Dictionary	File name
		Message Wordcount Min	Integer
		Message Wordcount Max	Integer
		Keyword List	String up to 4096 bytes
		Attachment Filename	File name
		Static Attachment	True or False
		Random Attachment	True or False
		Random File Size – Controls the size of a randomly generated attachment before it is encoded.	0 – 52,428,800
		Random File Size Min – Controls the size of a randomly generated attachment before it is encoded.	Integer
		Random File Size Max – Controls the size of a randomly generated attachment before it is encoded.	Integer
		Attachment Content-Type	String up to 128 bytes

Action	Description	Action Parameter	Valid Values
Client: Conditional Request	Defines the specific responses you expect to see from the DUT. Matches responses and checks whether response has the pattern in the payload.	Transaction Flag, Available Actions	Start, Continue, End, or Start and End
		Wait for Success	Select to wait for a response that matches the selected pattern.
		Match	The pattern you want to match against.
		Simple String	Select to match a simple string pattern.
		Regex	Select to match a regular expression pattern.
		Predefined	Select to match a predefined pattern.
		Available Action	Delay, Raw Message, Verify Rows, Goto, Close, Fail, Login, Login Request, Authenticate, Use Database, Query, Quit

Action	Description	Action Parameter	Valid Values
Client: Send email	Sends email.	None	No Response
Server: 250 Queued	Sends a 250 Queued response. Respond that the transmitted message has been queued.	Transaction Flag	Start, Continue, End, or Start and End
Client: Send QUIT	Sends a QUIT message to the server.	Transaction Flag	Start, Continue, End, or Start and End
Server: 221 Closing	Sends a 221 BYE response and closes the connection.	Transaction Flag	Start, Continue, End, or Start and End

SNMP Action Parameters

The table below lists the actions and action parameters for SNMP.

SNMP Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Walk	Simulates walking an agent's MIB with a series of snmp-get-next requests.	Transaction Flag	Start, Continue, End, or Start and End
		Community String	String up to 255 characters
		Initial Request ID	Integer

Action	Description	Action Parameter	Valid Values
Server: Trap	Simulates a single SNMP trap. Fields left blank will generate random values.	Transaction Flag	Start, Continue, End, or Start and End
		Community String	String up to 255 characters
		Object ID	String up to 65,535 characters
		Generic Trap Type	coldStart, warmStart, linkDown, linkUp, authenticationFailure, egpNeighborLoss, enterpriseSpecific
		Specific Trap Type	String
Server: Inform Request	Simulates a single inform request. Fields left blank will generate random values.	Transaction Flag	Start, Continue, End, or Start and End
		Community String	String up to 255 characters
		Request ID	Integer
		Object ID	String up to 65,535 characters

SSH Action Parameters

The table below lists the actions and action parameters for SSH.

SSH Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends an SSH flow.	Transaction Flag	Start, Continue, End, or Start and End

STUN Action Parameters

The table below lists the actions and the action parameters available for STUN.

STUN Action Parameters

Action	Description	Action Parameter	Valid Values
Binding Request	Sends a Binding Request to a STUN server.	Transaction Flag	Start, Continue, End, or Start and End
		Change IP Flag	True or False

SUN RPC Action Parameters

Sun RPC is used to perform remote procedure calls from a program running at a remote host.You can use the actions and action parameters listed in the table below to set up a Sun RPC flow.

Sun RPC Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Call	Sends a call to the server and waits for a response.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID (XID)	0 – 4,294,967,295
		Sun RPC Version	0 – 4,294,967,295
		Program	0 – 4,294,967,295
		Program Version	0 – 4,294,967,295
		Procedure	0 – 4,294,967,295
		Credentials Flavor	0 – 4,294,967,295
		Credentials Length	0 – 4,294,967,295
		Verifier Flavor	0 – 4,294,967,295
		Verifier Length	0 – 4,294,967,295
		Payload	Enter up to 50 alphanumeric and/or special characters to define the payload.

Action	Description	Action Parameter	Valid Values
Server: Reply	Sends a reply message to the client.	Transaction Flag	Start, Continue, End, or Start and End
		Transaction ID (XID)	0 – 4,294,967,295
		Reply State	0 – 4,294,967,295
		Verifier Flavor	0 – 4,294,967,295
		Verifier Length	0 – 4,294,967,295
		Accept State	0 – 4,294,967,295
		Payload	Enter up to 50 alphanumeric and/or special characters to define the payload.

Sybase Action Parameters

The table below lists the actions and action parameters Sybase.

Sybase Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a login to a Sybase database	Database Username	Enter up to 255 alphanumeric and/or special characters for the username that will be used to log into the database.
		Database Password	Enter up to 255 alphanumeric and/or special characters for the database user's password.
		Database Hostname	Enter up to 255 alphanumeric and/or special characters for the database's hostname.
Client: SQL Query	Simulates an SQL query and response	SQL Query	Alphanumeric and/or special characters can be used to define the SQL query.
		Columns	0 – 4,294,967,295
		Rows	0 – 4,294,967,295
		Column Names	Enter up to 255 alphanumeric and/or special characters to define the names of the columns from which data will be returned. If the SQL Query references any column names, then you should enter those column names in this field. The information listed here must be comma-delimited format.
		Table Name	Enter up to 255 alphanumeric and/pr special characters to define the table name referenced in the SQL query. If the SQL query is random (or left blank), the table name will be consistent with the randomly generated FROM clause of the SQL statement.

Syslog Action Parameters

The table below lists the actions and action parameters for Syslog.

Syslog Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Syslog Message	Sends a Syslog entry to the server	Transaction Flag	Start, Continue, End, or Start and End
		Process ID (PID)	0 – 4,294,967,295
		Priority	0 – 4,294,967,295
		Timestamp	Use alphanumeric and/or special characters to define the timestamp for the log entry
		Tag	Use alphanumeric and/or special characters to define the tag for the log entry
		Content	Use alphanumeric and/or special characters to define the contents of the log entry

TDS Action Parameters

The table below lists the actions and action parameters for TDS.

TDS Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Performs a login to the Microsoft SQL Server.	Username	Enter up to 14 special and/or alphanumeric characters for the username that will be used to log into the Oracle Database Server.
		Password	Enter up to 14 special and/or alphanumeric characters for the password that will be used to log into the Microsoft SQL Server.
		Server Name	Enter up to 63 special and/or alphanumeric characters for the server name.
		Client Name	Enter up to 63 special and/or alphanumeric characters for the client name.

Action	Description	Action Parameter	Valid Values
Client: Query: Use Database	Executes a "use database" statement.	Database Name	Enter up to 63 special and/or alphanumeric characters to define the database name.
Client: Query: Select	Executes a SELECT statement.	SELECT Modifier	True or False
		SELECT Modifier Value	Enter up to 127 special and/or alphanumeric characters to define the SELECT modifier that will be used. If no value is supplied, then a SELECT statement will be randomly generated with the modifier of ALL, DISTINCT, or no modifier. Otherwise, the value supplied will immediately follow the SELECT keyword (e.g., This functionality can be used to create union selections or other complex database queries.
		Column List	The columns to return, or alternatively, an *. If no value is provided, the system generates a random set of column names.
		FROM Table Name	The table name from which the resultant rows are returned (e.g., CustomerDB, Users).
		WHERE Comparison	True or False
Client: Query: Select	Executes a SELECT statement.	WHERE Comparison Value	Enter up to 127 special and/or alphanumeric characters to define the WHERE comparison that will be used. If supplied, the SELECT statement will include a WHERE comparison clause using the value defined here. If no values are given, a random comparison using one of the columns provided in the SELECT statement is generated (e.g., Username IS NOT NULL, LastLogon BETWEEN '01/01/2007' AND '12/31/2008').
		ORDER BY Expression	True or False
		ORDER BY Expression Value	Enter up to 127 special and/or alphanumeric characters to define the ORDER BY expression that will be used. If supplied, the SELECT statement will include an ORDER BY expression. Do not include the ORDER BY keyword in the expression.

Telnet

The table below lists the actions and action parameters for Telnet.

Telnet Action Parameters

Action	Description	Action Parameter	Valid Values
Server: Send Flow	Sends a telnet flow.	None	N/A

TIME Action Parameters

The table below lists the actions and action parameters for the TIME protocol.

TIME Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Send Time	Sets the date and time to send to the client.	Date/Time	A date and time in the format of YYYY-MM-DD HH:MM:SS; YYYY can be replaced with a value between 1970 and 2035.

TNS Action Parameters

The table below lists the actions and action parameters for TNS.

TNS Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Client: Login	Performs a login to the Oracle Database Server.	Database Username	Enter up to 14 special and/or alphanumeric characters for the username that will be used to log into the Oracle Database Server.
		Database Password	Enter up to 14 special and/or alphanumeric characters for the password that will be used to log into the Oracle Database Server.
		Server Name	Enter up to 63 special and/or alphanumeric characters for the server name.
		Database Name	Enter up to 63 special and/or alphanumeric characters for the database name.
		Server OS	Enter up to 63 special and/or alphanumeric characters for the server OS type.
		Server Banner	Enter up to 128 special and/or alphanumeric characters for the TNS server banner.
		Client Username	Enter up to 63 special and/or alphanumeric characters to define the username for the client.
Client: Login	Performs a login to the Oracle Database Server.	Client Machine Name	Enter up to 32 special and/or alphanumeric characters to define the client machine's name.
		Client Machine OS	Enter up to 32 special and/or alphanumeric characters to define the client machine's operating system.
		Client Program Path	Enter up to 32 special and/or alphanumeric characters to define the client program path. If no values are provided, a random path is generated appropriate to the OS. Otherwise, the values provided are passed as part of the authentication sequence (e.g., /usr/local/bin/oracle/).
		Client Program Name	Enter up to 32 special and/or alphanumeric characters to define the client program name. If no values are provided, a random program name is generated appropriate to the OS. Otherwise, the values provided are passed as part of the authentication sequence (e.g., isql.exe).
		Client Domain	Enter up to 32 special and/or alphanumeric characters to define the client's domain.

Action	Description	Action Parameter	Valid Values
Client: Query: Select	Executes a SELECT statement using the statement modifier values defined for **WHERE Comparison** and **ORDER BY Expression**.	Column List	The columns to return, or alternatively, an *. If no value is provided, the system generates a random set of column names.
		FROM Table Name	The table name from which the resultant rows are returned (e.g., CustomerDB, Users).
		WHERE Comparison	True or False
		WHERE Comparison Value	Enter up to 127 special and/or alphanumeric characters to define the WHERE comparison that will be used. If supplied, the SELECT statement will include a WHERE comparison clause using the value defined here. If no values are given, a random comparison using one of the columns provided in the SELECT statement is generated (e.g., Username IS NOT NULL, LastLogon BETWEEN '01/01/2007' AND '12/31/2008').
Client: Query: Select	Executes a SELECT statement using the statement modifier va**lues defined for WHERE Comparison** and **ORDER BY Expression**.	ORDER BY Expression	True or False
		ORDER BY Expression Value	Enter up to 127 special and/or alphanumeric characters to define the ORDER BY expression that will be used. If supplied, the SELECT statement will include an ORDER BY expression. Do not include the ORDER BY keyword in the expression.

World of Warcraft

The table below lists the actions and action parameters for the World of Warcraft.

World of Warcraft Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

YIM Action Parameters

Yahoo IM supports many features such as instant messaging, file transfer, webcam, and voice communications; however, the most common use is text IM-based chatting. You can use the actions and action parameters to simulate instant messaging conversations between a user and a peer.

Note: All action parameters that are left blank will generate random values. Some action parameters will allow you to use '0' to utilize random value generation; these action parameters will have **(0 == random)** listed next to them.

YIM Action Parameters

Action	Description	Action Parameter	Valid Values
Client: Delay	Delays the client's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000

Action	Description	Action Parameter	Valid Values
Server: Delay	Delays the server's response for the amount of time specified for **Number of milliseconds**	Transaction Flag	Start, Continue, End, or Start and End
		Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Super Flow editor will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
Client: Login	Simulates a Yahoo Instant Messenger login.	Buddy List	A comma-separated list of buddy names can be entered in this field. If this field is defined, the **Number of Buddie**s parameter will be ignored.
		Number of Buddies	0 – 99
Client: Chat	Simulates a Yahoo IM conversation between a user and a peer.	Client Messages	0 – 4,294,967,295
		Peer Messages	0 – 4,294,967,295
		Peer Name	Up to 32 alphanumeric and/or special characters can be defined for the peer's name.
Client: IM User	Simulates an instant message from the user to a peer.	Peer Name	Up to 32 alphanumeric and/or special characters can be defined for the peer's name.
		Message	Use alphanumeric and/or special characters to define the message sent from the user to the peer.
Server: IM: Peer	Simulates an instant message from the peer to the user.	Peer Name	Up to 32 alphanumeric and/or special characters can be defined for the peer's name.
		Message	Use alphanumeric and/or special characters to define the message sent from the peer to the user.

Conditional Requests

Test components that feature the Conditional Request Super Flow action enable you to send a single Super Flow to a server device under test and search for specific responses from the server. In order to do this, you must configure the test component to use a single Super Flow that utilizes Conditional Requests.

Note: The Conditional Request Super Flow action is not available in all test components.

Conditional Requests enable you to set up to several matches for a packet; these matches represent the specific responses (i.e., strings or patterns) that you expect to see from the device under test.

Note: Starting with BPS version 8.01, the maximum number of matches for Conditional Request actions is 63. The maximum number of matches for versions prior to BPS 8.01 is three. The matches for Conditional Requests can be defined individually or a Multi-Match Response 200 OK action may be a more efficient method for you to define multiple match conditions.

The system will process each match listed in the Conditional Request in the order in which it is listed. Additionally, you can define one mismatch for the Conditional Request; this occurs when there is no match or time out (no response) from the DUT.

It is important to understand the precedence employed by these components when running tests with Conditional Requests. Conditional Request allows up to three strings to be specified. The first string has precedence over the second string; the second string has precedence over the third string.

However, the first packet has precedence over the second packet, and the second packet has precedence over the third packet, etc. Because packets are placed above strings in the hierarchy, if the third string shows up in the first packet, then that is the match (out of the three) that is counted. Conversely, if the first string shows up in the third packet, that match will not be counted if there was a match in the first two packets.

The first successful string match will increment the appropriate counter. If there is no match in the server response, or a timeout occurred, the "no match" counter is incremented. This is true even when **Wait for Success** is selected.

Exception: if the server response is "chunked", that means there are more packets coming from the server. Client Simulator will defer incrementing the "no match" counter until the last packet in the "chunked" response is received, or until there is a match.

When using Conditional Request without Wait for Success being selected, be aware that the Conditional Request will consume the first packet. This results in the first packet being unavailable for other actions such as File Verify.

Note: BreakingPoint does not decode chunked data for Verify actions. Tests that use chunked data will fail if a Verify action is used.

The Time out field works in conjunction with the Wait for Success option.

When the Wait For Success option is checked, the Time out field indicates the number of seconds that Conditional Request will wait for the next packet. The server will continue to send packets until the timeout period ends, or until a match is found.

When the Wait For Success option is not checked, the Time out field indicates the number of seconds that the Conditional Request will wait for the first packet from the server. Once the first packet arrives, the Conditional Request will determine whether that packet is a match or a mismatch. The first packet is the only packet used in this scenario.

A zero (0) in the Time out field will cause the Conditional Request to continue to wait (indefinitely) until a packet arrives. This practice is not recommended by BreakingPoint.

Matches

For each match, you will need to specify the string the system should look for (e.g., 200 OK). If the string matches, then the system will respond with the Action you have specified for that string (e.g., Server: Response 200 (OK)). When specifying the Action for the string match, you can configure the Action Parameters as you normally would. For more information on Action Parameters, see Actions and Action Parameters on page 369.

The test component will track the number of responses from the server that match the string matches defined within the Conditional Requests for the Super Flow. This data will be available in the Response Summary of the test report.

Multi-Match Response 200 OK

The Multi-Match Response 200 OK action defines the HTTP 200 OK response that occurs when criteria is met for a Conditional Request that has multiple matches. This feature addresses complex scenarios involving a distribution of response size and related extension file types.

Match responses are defined by way of a JSON configuration file.

Unsupported Features

At the time of the BPS 8.01 release, the following features are not supported:

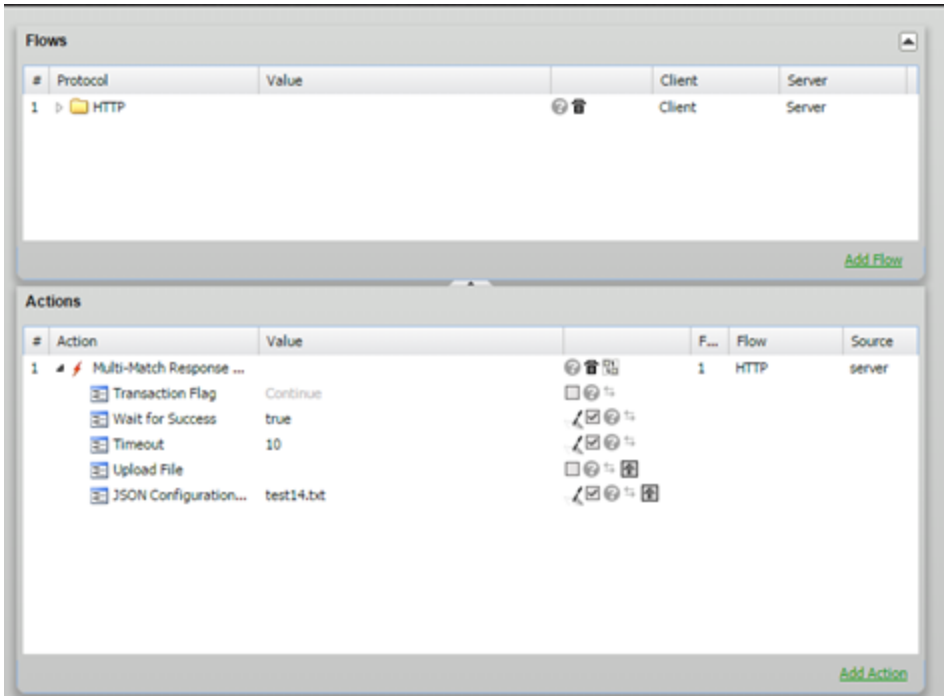
- Delay before "Multi-Match Response 200(OK)"
- Goto reference of "Multi-Match Response 200(OK)" action inside in conditional request

JSON Configuration File Description

You will need to create a JSON configuration file that contains a key of matches at the root with values that are an array of objects. Each object in the array configures the corresponding match. There may be up to 63 different matches and any subsequent matches will be ignored.

How to Reference a JSON Configuration File

The image below displays how a Multi-Match Response 200 OK action references a JSON configuration file named "test14.txt".



JSON Configuration Data

Within the JSON Configuration file, each object in the array of matches configures the corresponding match and may contain the configuration data described below.

Upload File: This action lets the user reference the JSON configuration file as described in the previous sections.

JSON Configuration: This file contains a JSON structure that describes each match and the corresponding HTTP 200 response that occurs when the criteria is met.

The JSON file should contain at the root a key of matches, whose value is an array of objects. There may be up to 63 different matches, with subsequent matches ignored. Each object in the matches array configures the corresponding match and may contain the following configuration data:

Pattern: The pattern to match. This can be a regular expression (regex) or a simple string (text).

Pattern_type: The type of pattern: regex or text. A default value of regex is used when a pattern type is not provided.

File_type: The type of file to generate for the response body of the HTTP 200 response. A default value of HTML is used when a file type is not provided. The File Types table below lists each choice and its corresponding MIME type.

File_size: The size of the file. The specified file size can be no larger than 104,857,600 bytes. The value may be a range, formatted as MIN-MAX where MIN is the minimum file size and MAX is the maximum file size. The default file size will be 256KB when a file size is not provided. Please note that some generated file types may not accommodate relatively small file sizes.

File: A static file to be placed in the message body of the HTTP 200 response. When a file is specified, the file_type and file_size parameters are ignored. Files may be uploaded by way of the upload file setting for this action.

Content_type: The MIME type of the file referenced in the file setting. This value is ignored unless the file setting is provided.

Available File Types

html	text/html
html5	text/html
xhtml_mb	text/html
css	text/css
flash	video/flv
flash-static	video/flv
yahoo-flash	video/flash
myspace-flash	video/x-flv
javascript	text/javascript
mp3	audio/mpeg
wav	audio/wav
docx	application/vnd.openxmlformats-officedocument.wordprocessingml.document
pdf	application/pdf
rtf	application/rtf
exe	application/octet-stream
pe	application/octet-stream
bmp	image/bmp
gif	image/gif
jpg	image/jpeg
png	image/png

JSON Configuration Examples

The following are examples of how the matches within a JSON configuration file are defined.

```
{
"matches" [
{
"pattern" "index.php?file=mp3",
"pattern_type" "text",
"file_type" "mp3",
"file_size" "300-500"
},
{
"pattern" "index.php?file=jpg",
"pattern_type" "text",
"file_type" "jpg",
"file_size" "1000"
},
{
"pattern" "index.php?file=jpg",
"pattern_type" "text",
"file_type" "jpg",
"file_size" "1000-2000"
}
]
}
```

Mismatches

A mismatch occurs when the Conditional Request does not receive a response from the DUT or when it receives a response with no match to the specified match strings. If the test component does not receive a response within the time-out period specified in the **Timeout** field, then test component will act as if it has received a response with no match. In the case of a mismatch, the test component will proceed to the action defined for the mismatch. This prevents the test component from remaining in the waiting state.

Note: If you specify a timeout value of 0, the component will wait for a response from the DUT. This practice is not recommended by BreakingPoint.

Creating a Conditional Request

From the Parameters area of the test component Test Editor, you will need to select a Super Flow that uses Conditional Requests. Before creating a Conditional Request, please review the following restrictions and guidelines:

- The Regex option allows you to use PCRE syntax to perform advanced matching.
- When using Conditional Request without Wait for Success being selected, be aware that the Conditional Request will consume the first packet. This results in the first packet being unavailable for other actions such as File Verify.
- When Wait for Success is selected, the Conditional Request will process subsequent packets until it finds a match, or until the timeout period is exceeded.
- The Conditional Request action should always follow the server action. The Network Processor stops searching for packets once the Conditional Request action has occurred. Placing the

Conditional Request before the server action (or the action that generates the data) can cause a "no match" to erroneously occur.

To create a Conditional Request:

1. Select **Managers > Super Flows** from the BreakingPoint Control Center Menu bar.
2. Open an existing Super Flow or create a new Super Flow.

Note: Only Super Flows based on the HTTP and SMTP protocols currently provide full support for Conditional Requests.

3. Select a flow from the **Flows** area.
4. In the Actions region of the window, click **Add Action**.
5. Select **Conditional Request** for the **Client** Source.
   a. Expand the Conditional Request action to show the action parameters.
   b. Select **Wait for Success** (change it from False to True) if you want BreakingPointto wait for a response that matches the configured patterns. If you do not select this option, BreakingPoint will examine the first packet in the response. If a match is not found, BreakingPoint will move on to the next action.
   c. Select the first **Match** parameter, and configure the conditional request:
   d. Expand the Match parameter to show the **Type** option.
   e. Set the **Type** to either text or Regex.
   f. Enter a Match string in the **Value** field. You can use a regular expression if you have set the Type to Regex.
   g. Click the **Plus (+)** icon to display the list of available actions.
   h. Select the action you would like to occur if the string is a match.
   i. Expand the newly-added action to show the parameters.
   j. Configure the parameters for the action.
   k. Optionally, add additional actions for this **Match** parameter. For more information on Action Parameters, see Actions and Action Parameters on page 369.
6. Repeat step 8 for one or both of the other **Match** parameters.
7. Select and expand the **Mis-match** parameter, then enter (in the **Timeout** field) the amount of time the system must wait before aborting the Super Flow when it encounters a mismatch.
8. If 0 is defined, then the Super Flow will not be aborted.
9. Click the **Plus (+)** icon to display the list of available actions for the Mis-match parameter.
10. Select the action you would like to occur when a mismatch occurs.
11. Expand the newly-added action, then configure its parameters.
12. Click the **Save** button when done.

Conditional Request Action Parameters

The table below lists the Conditional Request action parameters.

Conditional Request Action Parameters

Action	Description	Parameters	Valid Values
Update Dest Address	Updates the server address in subsequent flows with the value from a previous PCRE match	Destination Host – The server host name whose address will be updated.	Client or Server
		Match Variable – The PCRE match variable slot where the server address is stored.	0 - 9
Update Dest Port	Updates the destination port of a flow with the value from a previous PCRE match.	Flow ID – Represents the flow to update	1 - 16
		Match Variable – The PCRE match variable slot where the port number is stored	0 - 9
Verify File	Verifies data coming from the server with a specified resource file	Transaction Flag	Continue, End, Start, StartEnd
		File to verify – The resource file specified here will be used to verify the response from the server.	Available file
Log as Target	While configuring a Lawful Intercept test, it may be necessary to generate a target Super Flow that does not contain a needle. Including this action in the Super Flow will cause a Lawful Intercept test to log the presence of the Super Flow as if it contained a needle.	Transaction Flag	Continue, End, Start, StartEnd

Action	Description	Parameters	Valid Values
Add Flow Dictionary	Provides the ability to add a dictionary to the flow to which it belongs.	Dictionary ID – The identifier by which this dictionary is referenced.	
		Dictionary File – The name of the file containing entries to choose from. Up to one thousand entries will be read from the specified file.	Available file
		Dictionary Delimiter Type – The type of delimiter between entries.	Custom, New Line
		Dictionary Custom Delimiter – The delimiter for the custom dictionary file.	

Regular Expression

In addition to simple string matching, the regular expression (Regex) option allows you to match test strings using the advanced capabilities of Perl Compatible Regular Expressions (PCRE). With the Regex option, you can match strings and substrings in data packets as well as capture the data in those packets. Once the data has been captured, it can be placed into a set of token variables that may be reused later within the Super Flow. This allows support for HTTP redirects, persistent cookies, and items that require extracting data from either the server or an inline device.

Note: Releases 1.4.1 and higher of BreakingPoint uses PCRE version 7.8 functionality. Please see the latest PCRE documentation for acceptable PCRE syntax patterns.

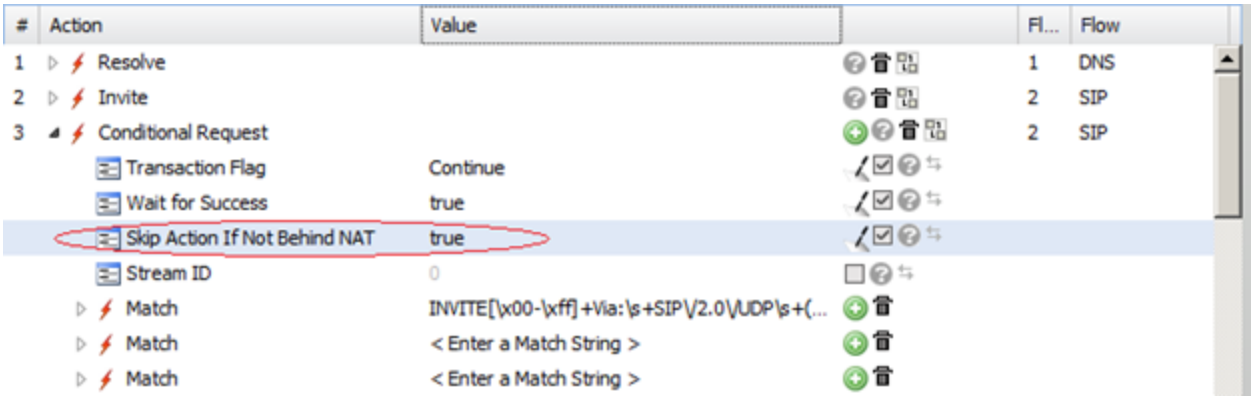
NAT Environment Options

This section describes options that are available when creating application profiles/super flows that will be used in a test environment that includes NAT.

Skip Action if Not Behind NAT

This option helps to accommodate performance testing in non-NAT scenarios by allowing an action to be skipped.

Skip Action if Not Behind NAT is a Conditional Request parameter. When the parameter is set to "true" the Conditional Request action will be skipped if the NAT check box is disabled in the Network Neighborhood that is associated with the test.



##nat_cookie(0)## Custom Token

BreakingPoint provides a custom token to solve the problem where UDP matching by first packet is invalidated due to information changing in the packet payloads by Application-level Gateway capable DUTs. This option should be used to ensure that UDP-flow based super flows work through NAT.

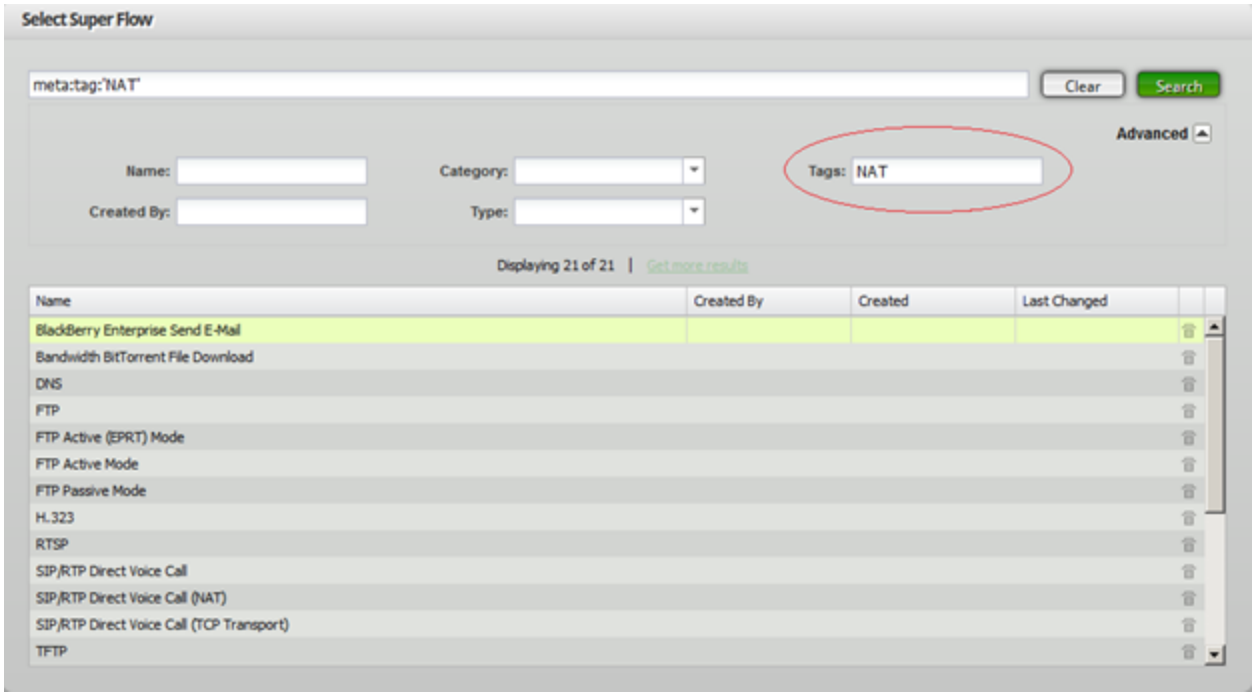
Note: NAT Hash collision errors can occur when multiple packets have the same hash or same payload and have not been transmitted by the end of the test.

NAT Tagging

Superflows and appmixes that have been validated in NAT scenarios are "NAT" tagged. Custom, user-created super flows that work through NAT can also be manually tagged with the same "NAT" tag.

The test components that are supported with NAT are AppSim and Session Sender.

You can easily filter the superflows and appmixes that are qualified for NAT by using the NAT tag in your search as shown in the image below.


Select Super Flow
meta:tag:'NAT'
Clear
Search
Advanced
Name:
Category:
Tags: NAT
Created By:
Type:
Displaying 21 of 21 | Get more results
Name
Created By
Created
Last Changed
BlackBerry Enterprise Send E-Mail
Bandwidth BitTorrent File Download
DNS
FTP
FTP Active (EPRT) Mode
FTP Active Mode
FTP Passive Mode
H.323
RTSP
SIP/RTP Direct Voice Call
SIP/RTP Direct Voice Call (NAT)
SIP/RTP Direct Voice Call (TCP Transport)
TFTP

This page intentionally left blank.

CHAPTER 15 Load Profiles

This section covers:

Load Profiles Overview

Load Profiles allow you to customize the behavior of sessions during the different phases of an Application Simulator, Client Simulator, Recreate, or Session Sender test.

A Load Profile consists of multiple phases. Each phase is based on a phase type, represents a specific time frame, and determines the behavior of the sessions that are opening and/or closing during that time frame. You can further configure each phase by setting the maximum number of sessions, the session rate, and the data rate. All the settings are applicable to only that particular phase.

Each Load Profile must have one ramp up phase, one ramp-down phase, and at least one steady-state phase. By default, all Load Profiles will have a Ramp Up and Ramp Down phase. You can add multiple steady-state phases to the Load Profile; however, there is a 300-phase limit for each Load Profile.

The controllable attributes for the phases include the following: phase duration, data rate, session behavior, number of sessions per second, and maximum number of sessions. The Load Profile parameters you specify will override the configurations in the Parameters section of the Component Settings page.

The Load Profile button is located on the Component Settings page. There are three types of Load Profiles:

- Default Load Profiles
- Stair-Step Load Profiles
- Custom/Saved Load Profiles

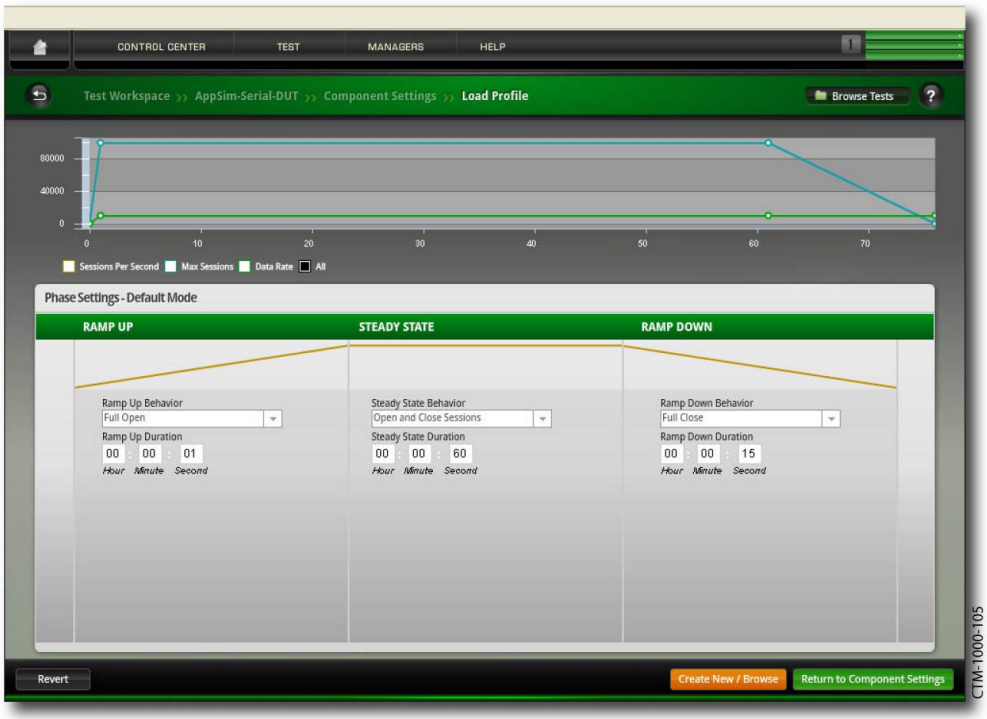
Default Load Profile

When you initially click the Load Profile button on the Component Settings page, the settings for the Default Load Profile are displayed on the Load Profile page.

The Default Load Profile is the most basic load profile offered. It has one ramp up phase, one ramp-down phase, and one steady-state phase. You have the option of running a test using the default settings, or you can adjust the settings.

The image below displays the parameters associated with a Default Load Profile.

Default Load Profile Window



Default Load Profile Phase Settings

The table below lists the parameters in the Phase Settings section of the Default Load Profile and provides descriptions for each.

Default Load Profile Parameters

Parameter	Description	Valid Values
Ramp Up Behavior	Sets how the component will open sessions during the ramp up phase.	**Full Open** – The full TCP handshake performed when sessions are opened. **Full Open + Data** – The full TCP handshake performed when sessions are opened. Data sent once the session opens. **Full Open + Data + Full Close** – The full TCP handshake performed when sessions are opened. Data sent once the session opens. Sessions are closed as they finish sending data and new sessions are opened in their place. **Full Open + Data + Close with Reset** – The full TCP handshake performed when sessions are opened. Data sent once the session opens. A TCP close with a RST is initiated and the TCP close state machine is bypassed. **Half Open** – The full TCP handshake performed when sessions are opened, but the final ACK is omitted. **SYN Only** – Only SYN packets are sent. **Data Only** – Only PSH data packets are sent. The state machine is bypassed, so no connections are set up; therefore, the ACKs will be invalid.
Steady-State Behavior	Sets how the component will handle sessions during the steady-state phase.	**Open and Close Sessions** – Sessions are closed as they finish sending data, and new sessions are opened. **Hold Sessions Open** – No existing sessions opened during Ramp Up are closed. **Open and Close with Reset** – A TCP close with a RST is initiated and the TCP close state machine is bypassed. **Open and Close with Reset Response** – Once a session is closed, the server will respond with a RST and change to the TCP CLOSED state. This option bypasses the TCP TIME_WAIT state.
Steady-State Duration	Sets the amount of time sessions have to open, send data, and close. The system will have to maintain the number of open session for this time period.	hh:mm:ss

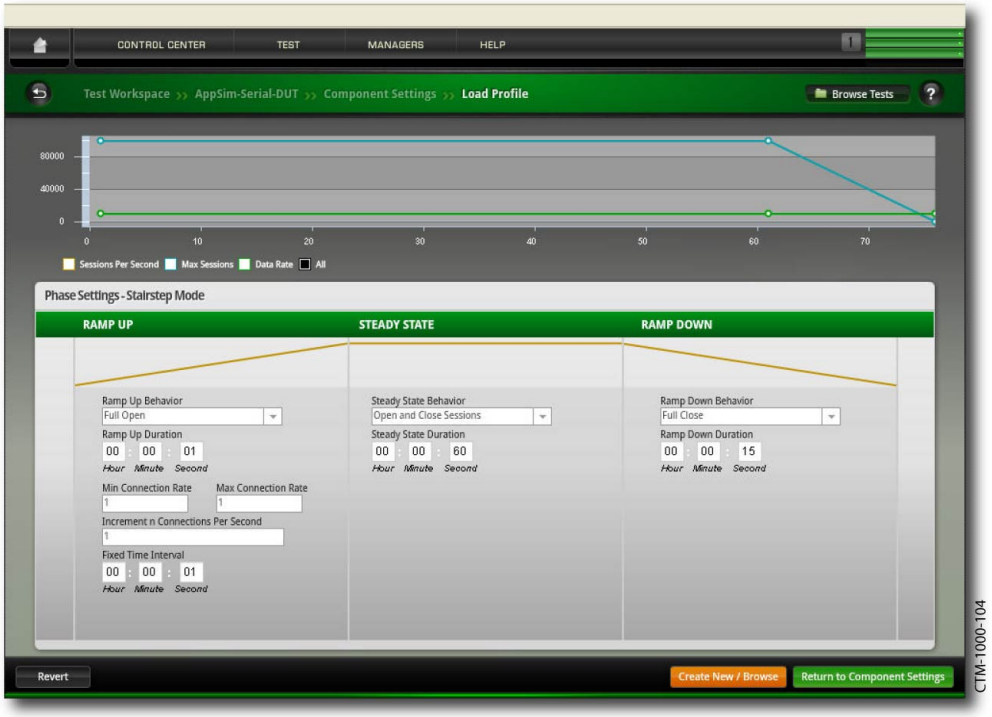
Parameter	Description	Valid Values
Ramp Down Behavior	Sets how the component will close sessions during thentime period specified for **Ramp Down Duration**.	**Full Close** – The full TCP session close is performed. **Half Close** – The full TCP session close is performed, but the final ACK is omitted. **Reset** – Close all sessions by sending TCP RST (reset) packets. **Reset Response** – Respond to a FIN with a RST. This bypasses the TCP TIME_WAIT state.
Ramp Down Duration	Sets the amount of time open sessions have to close.	hh:mm:ss

Stair-Step Load Profile

Stair-Step Load Profiles allow you to set the minimum connection rate that the system will start with and increment until it reaches the maximum connection rate or until the ramp up phase elapses. With a Stair-Step Load profile, you can determine whether the connection establishment rate is a constant rate or an incremental rate.

The image below displays the parameters associated with the Stair-Step Load Profile.

Stair-Step Load Profile Window



Stair-Step Load Profile Phase Settings

The parameters found on the Stair-Step Load Profile screen are very similar to those found on the Default Load Profile screen. However, the Stair-Step Load Profile Screen has additional Ramp-Up settings that the Default Load Profile does not. The parameters that are unique to the Stair-Step Load Profile are listed in the table below.

Stair-Step Load Profile Parameters

Parameter	Description	Valid Values
Ramp Up Duration	Sets the duration for which new sessions can be opened.	hh:mm:ss
Min Connection Rate	Sets the minimum connection establishment rate that will be used to start the ramp up phase.	1 – 1,000,000
Max Connection Rate	Sets the maximum connection establishment rate that the system will attempt to reach during the ramp up phase. Once the system reaches this rate, it will continue to hold this rate until the ramp up phase ends.	1 – 1,000,000
Increment n Connections Per Second	Sets the number of connections that the connection establishment rate will be incremented by for the time specified for Fixed Time Interval.	1 – 500,000
Fixed Time Interval	The interval at which connection establishment rate will be changed when not in Calculated mode.	hh:mm:ss

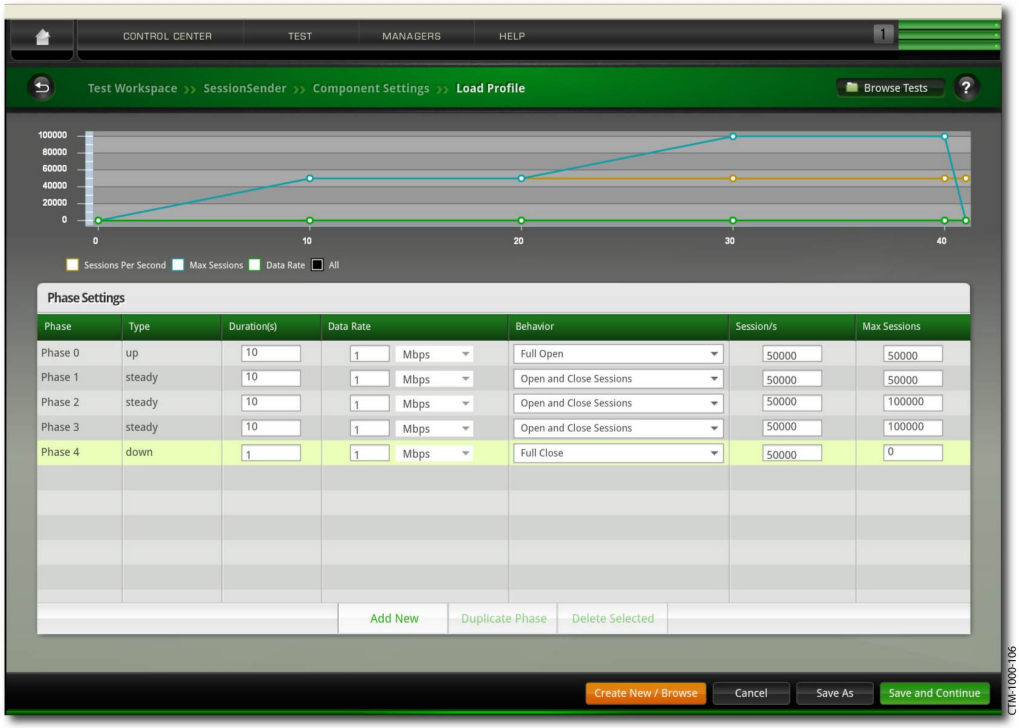
Custom/Saved Load Profiles

The system offers several Custom Load Profiles that you can use without modification if you do not want to create your own Profiles.

However, there may be instances when the settings of the Custom Load Profiles do not match the requirements of your testing environment. In those cases, you may want to create a customized load profile. To create your own customized load profile, you can edit any of the Custom/Saved Load Profiles and save them under a different name.

The image below displays the parameters associated with the Custom/Saved Load Profile.

Custom/Saved Load Profile Window



Custom/Saved Load Profile Phase Settings

The table below lists the parameters in the Phase Settings section of the Custom/Saved Load Profile and provides descriptions for each.

Custom/Saved Load Profile Phase Settings Parameters

Parameter	Description	Valid Values
Duration(s)	Sets the duration for which new sessions can be opened.	0 – 1,000,000
Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces.	1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps

Parameter	Description	Valid Values
Behavior (Ramp Up)	Sets how the component will open sessions during the ramp up phase.	**Full Open** – The full TCP handshake is performed when sessions are opened. **Full Open + Data** – The full TCP handshake is performed when sessions are opened. Data will be sent once the session opens. **Full Open + Data + Close** – The full TCP handshake is performed when sessions are opened and data will be sent once the session opens. Sessions are closed as they finish sending data and new sessions are opened in their place. **Half Open** – The full TCP handshake is performed when sessions are opened, but the final ACK is omitted. **SYN Only** – Only SYN packets are sent.
Behavior (Steady-State)	Sets how the component will open sessions during the steady-state phase.	**Open and Close Sessions** – Sessions are closed as they finish sending data, and new sessions are opened. **Hold Sessions Open** – No existing sessions opened during Ramp Up are closed. Open and Close with Reset – A TCP close with a RST is initiated and the TCP close state machine is bypassed. **Open and Close with Reset Response** – Once a session is closed, the server will respond with a RST and change to the TCP CLOSED state. This option bypasses the TCP TIME_WAIT state.
Behavior (Ramp Down)	Sets how the component will close sessions during the ramp down phase.	**Full Close** – The full TCP session close is performed. **Half Close** – The full TCP session close is performed, but the final ACK is omitted. **Reset** – Close all sessions by sending TCP RST (reset) packets. Reset Response – Respond to a FIN with a RST. This bypasses the TCP TIME_WAIT state.
Session/s	Sets the number of maximum sessions for this phase	1 – 1,000,000
Max Sessions	Sets the number of new sessions per second for this phase	1 – 1,000,000

Custom/Saved Load Profile Descriptions

See the table below for a listing of Custom/Saved Load Profiles and their descriptions.

Custom/Saved Load Profiles

Load Profile	Description
BreakingPoint 10K Maximum Megabits per second	This Load Profile contains 11 phases and is useful if you want to increment the data rate for each phase until it reaches the maximum data rate. The test will ramp up at 1,000 Mbps. During each steady-state, the system will increment 1,000 Mbps until it reaches 10 Gbps, and it will maintain 50,000 sps and 50,000 maximum simultaneous sessions for each phase. During ramp down, the system will drop to 9,000 Mbps. You must use the Small Network or Medium Network template for the test to get the expected results for this Load Profile. The template must be selected before you choose this Load Profile for the test. To select the Small to Medium Business Apps template, go to the Component Settings page; click the Load a Template button; select the Small to Medium Business Apps template; and click the Load button. After you have done this, you can go to the Parameters tab to choose this Load Profile.
BreakingPoint 1K Maximum Megabits per second	This Load Profile contains 11 phases and is useful if you want to increment the data rate for each phase until it reaches the maximum data rate. The test will ramp up at 100 Mbps. During each steady-state, the system will increment 100 Mbps until it reaches 1 Gbps, and it will maintain 50,000 sps and 50,000 maximum simultaneous sessions for each phase. During ramp down, the system will drop to 900 Mbps. You must use the Small to Medium Business Apps template for the test to get the expected results for this Load Profile. The template must be selected before you choose this Load Profile for the test. To select the Small to Medium Business Apps template, go to the Component Settings page; click the Load a Template button; select the Small to Medium Business Apps template; and click the Load button.After you have done this, click the Load Profile button to select this Load Profile.
BreakingPoint 10K Maximum Simultaneous Sessions	This Load Profile contains 152 phases and is useful if you want to gradually increment the maximum number of simultaneous sessions until it reaches 20,000,000. The test will run at 900 Mbps.The system will gradually increment Max Sessions at each steady-state phase until it reaches the maximum number of simultaneous sessions supported by the system. You must use the Small Network or Medium Network preset for the test to get the expected results for this Load Profile. The preset must be selected before you choose this Load Profile for the test. To select the Small to Medium Business Apps template, go to the Component Settings page; click the Load a Template button; select the Small to Medium Business Apps template; and click the Load button. After you have done this, click the Load Profile button to select this Load Profile.

Load Profile	Description
BreakingPoint 1G Maximum Simultaneous Sessions	This Load Profile contains 152 phases and is useful if you want to gradually increment the maximum number of simultaneous sessions until it reaches 7,500,000. The test will run at 900 Mbps.The system will gradually increment Max Sessions at each steady-state phase until it reaches the maximum number of simultaneous sessions supported by the system. You must use the Small to Medium Business Apps template for the test to get the expected results for this Load Profile. The template must be selected before you choose this Load Profile for the test. To select the Small to Medium Business Apps template, go to the Component Settings page; click the Load a Template button; select the Small to Medium Business Apps template; and click the Load button. After you have done this, click the Load Profile button to select this Load Profile.
BreakingPoint Default	This Load Profile contains three phases: Ramp Up, Phase 1, and Ramp Down. This Load Profile is useful because it essentially provides a blank template for creating additional phases. The system will transmit traffic at a constant rate of 900 Mbps. When the test ramps up, the system will open 500,000 sessions at a rate of 50,000 sps. Once the system reaches steady-state (Phase 1), it will keep the 50,000 sessions open for 28 seconds. During the ramp down phase, all opened sessions will be closed.
BreakingPoint Maximum Sessions per second	This Load Profile is only available for Session Sender. It contains 153 phases and is useful if you want to gradually increment the session rate until it reaches the maximum session rate supported by the system. The system will transmit traffic at a constant rate of 1,000 Mbps. When the test ramps up, the system will open 50,000 sessions at a rate of 5,000 sps. It will hold the 50,000 simultaneous sessions at 5,000 sps during the first steady-state. After the first steady-state (Phase 1), the system will increment Sessions Per Second by 3,300 for each new steady-state phase. During ramp down, the system will close all open sessions. You must use the Maximum Possible template for the test to get the expected results for this Load Profile. The template must be selected before you choose this Load Profile for the test. To select the Maximum Possible template, go to the **Component Settings** page; click the **Load a Template** button; select **Maximum Possible**; and click the **Apply Changes** button. Before you can choose the Load Profile, you will need to go to the Parameters tab and set the following parameter configurations: • Segment Size Distribution.Distribution type – constant • Segment Size Distribution.Minimum value – 1 • Payload Packets Per Session – 1 • Payload.Type – 1

Phases

The previous section discussed the different phases in a Session Sender, Application Simulator, and Recreate test. This section will provide a brief overview of the different phases in a test so that you have a better understanding of how each phase works.

Ramp up Phase

During the ramp up phase, the system will open as many connections as possible, based on the values input for the phase duration, sessions per second, and maximum number of simultaneous sessions. For the system to open the maximum number of simultaneous sessions, you will need to determine the value to input for session rate and the duration of the ramp up phase. If you do not allot enough time or set the necessary session rate, the system will not open the maximum number of simultaneous sessions.

You can use the following equation to calculate the maximum number of sessions to open:

```
Maximum Simultaneous Sessions = Phase Duration x Sessions Per Second
```

For example, if you want to open 5,000,000 simultaneous sessions at a rate of 500,000 sessions per second, you will need to set the ramp up duration to 10 seconds.

Note: Since the system can only have one ramp up phase, you may need to use steady-state phases to replicate ramp up behavior. For example, if you want to increase the number of simultaneous sessions that were opened during the ramp up phase from 1,000,0000 to 2,000,000, you may want to add a steady-state phase ramps up to 2,000,000 sessions, and then add another steady-state phase that maintains those 2,000,000 sessions for the desired amount of time. For more information on steady-state phases, see Steady-State Phase below.

Steady-State Phase

Typically, during the steady-state phase, the system will open and close sessions at the specified session rate, while maintaining the maximum number of sessions opened during the ramp up phase. So, if the system opened 5,000,000 connections during the ramp up phase, the system will open and close sessions so that it maintains that number of connections.

With Load Profiles, you can create multiple steady-state phases, so it is possible to have steady-state phases that are maintaining a certain number of sessions and steady-state phases that are ramping up or ramping down to a certain number of sessions.

Note: If you need to use steady state phases to replicate ramp up behavior, you will need choose **Hold Sessions Open** as the phase behavior.

The following example sets up a Load Profile configuration that ramps up to 50,000 sessions, maintains the 50,000 sessions for 10 seconds, and then ramps up to 100,000 sessions. After the test reaches 100,000 sessions, it will keep those sessions open for 10 seconds.

Ramp Up

Parameter	Value
Duration	10
Sessions Per Second	50,000
Max Sessions	50,000

Phase 1 (Steady-State)

Parameter	Value
Duration	10
Sessions Per Second	50,000
Max Sessions	50,000

Phase 2 (Steady-State)

Parameter	Value
Duration	10
Sessions Per Second	50,000
Max Sessions	100,000

Phase 3 (Steady-State)

Parameter	Value
Duration	10
Sessions Per Second	50,000
	100,000

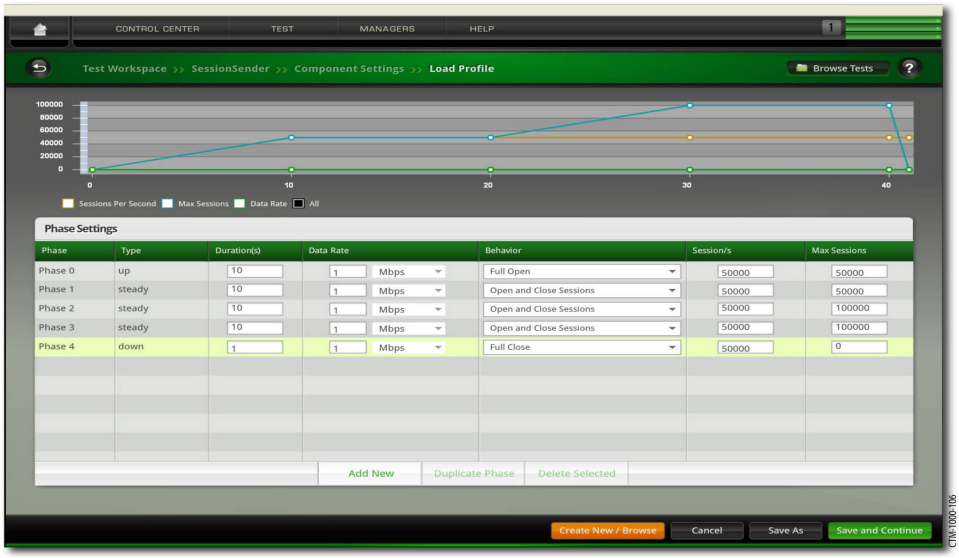
Ramp Down

Parameter	Value
Duration	1

Parameter	Value
Sessions Per Second	50,000
Max Sessions	0

The graph shown in Load Profile Example below is generated using these configurations and a constant data rate of 900. As you can see, the graph for Max Sessions looks like a stair case. Whenever you use constant values for the parameters, with one variable value, the graph that tracks the variable value will resemble a staircase.

Load Profile Example



Ramp Down Phase

During the ramp down phase, the system will close all open sessions. No new sessions will be opened.

Creating a Load Profile

The following section will describe how to create a Load Profile. All parameters must be completed.

> **Note:** The phase type phases cannot be changed. Each Load Profile must have one ramp up phase and one ramp down phase. All new phases that are added to the Load Profile will be steady-state phases.

To create a Load Profile:

Select Test > Open Test from the Menu bar.

Open an existing Session Sender test or create a new Session Sender test. For more information on creating a new test, see Creating a New Test on page 816

Note: You can also use an Application Simulator, Client Simulator, or Recreate test for this exercise.

1. Select the Session Sender component in the Test Components section.
2. Click the edit component icon.
3. Click the **Load Profile** button.
4. Click the **Create New/Browse** button at the bottom of the Load Profile page.
5. Click **Custom**.
6. Select an existing Load Profile from the Custom/Saved Load Profiles menu.

   Note: You must use an existing Load Profile as a template for creating a Load Profile. If you want to build your Load Profile from scratch, we recommend that you use BreakingPoint Default, which comes with three phases. When you are done customizing the Load Profile, you can save the Load Profile as a new Load Profile.

7. Click **Select**. The Phase Settings section will be displayed.
8. Do one of the following:

- If you are modifying the Ramp Up phase (Phase 0), do any of the following:
  - Click the Behavior drop-down box and select one of the following:
    - **Full Open** – If you want the full TCP handshake to be performed when sessions are opened.
    - **Full Open + Data** – If you want the full TCP handshake to be performed when sessions are opened and want data to be sent once the sessions are opened.
    - **Full Open + Data + Close** – If you want the full TCP handshake to be performed when sessions are opened; data to be sent once the sessions are opened; and sessions to be closed as soon they have finished sending data.
    - **Half Open** – If you want the full TCP handshake to be performed when the sessions are opened, but you want to omit the final ACK.
    - **SYN Only** – If you only want to send SYN packets.
  - Enter an integer between 0 – 1,000,000 in the Duration(s) field. This represents the duration of the ramp up phase.
  - Enter an integer between 1 – 750,000 (10Gb) or 500,000 (1Gb) in the **Sessions/s** field. This represents the rate at which connections are being opened and closed. Please note that the maximum number of sessions supported depends on the blade you are using (1 Gb or 10 Gb BreakingPoint Storm blade, or a BreakingPoint FireStorm blade).
  - Enter an integer between 1 – 20,000,000 (10Gb) or 5,000,000 (1Gb) in the **Max Sessions** field. This represents the maximum number of sessions that can be concurrently open at any given time. Please note that the maximum number of sessions supported depends on the blade you are using.
  - Click the **Data Rate** drop-down box and select one of the following:
    - Mbps (Megabits per Second)
    - Frames/ps (Frames per Second)
  - Enter an integer value between 1 – 10,000 (10Gb) or 1,000 (1Gb) (for Mbps) or 1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) (for fps) in the **Data Rate** field.

- If you are modifying the Steady-State phase (Phase 1), do the following:
  - Click the Behavior drop-down box and select one of the following:
    - **Open and Close Sessions** – If you want existing sessions to close as soon as they finish sending data and new sessions to open in their place.
    - **Hold Sessions Open** – If you want to keep open all sessions that were opened during the ramp up phase.
    - **Open and Close with Reset** - If you want to initiate the TCP close with a RST. This bypasses the TCP close state machine.
    - **Open and Close with Reset Response** – If you want to allow the server to close the session after the session has finished sending data. Once the server has closed the session, the client will send a RST.
  - Enter an integer between 0 – 1,000,000 in the **Duration(s)** field. This represents the duration of the steady-state phase.
  - Enter an integer between 1 – 500,000 (1Gb) or 750,000 (10Gb) in the **Session/s** field. This represents the rate at which connections are being opened and closed. Please note that the maximum number of sessions supported depends on the blade you are using.
  - Enter an integer between 1 – 5,000,000 (1Gb) or 20,000,000 (10Gb) in the **Max Sessions** field. This represents the maximum number of sessions that can be concurrently open at any given time. Please note that the maximum number of sessions supported depends on the blade you are using.
  - Click the **Data Rate** drop-down button and select one of the following:
    - Mbps (Megabits per Second)
    - Frames/ps (Frames per Second)
  - Enter an integer value between 1 – 10,000 (10Gb) or 1,000 (1Gb) (for Mbps) or 1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) (for fps) in the **Data Rate** field.
- If you are adding a steady-state phase, do the following:
  - Select an existing steady-state phase from the **Phase Settings** section.
  - Click the **Add New** button located at the bottom of the Load Profile page.
  - Click the **Behavior** drop-down button and select one of the following:
    - **Open and Close Sessions** – If you want existing sessions to close as soon as they finish sending data and new sessions to open in their place.
    - **Hold Sessions Open** – If you want to keep open all sessions that were opened during the ramp up phase.
    - **Open and Close with Reset** - If you want to initiate the TCP close with a RST. This bypasses the TCP close state machine.
    - **Open and Close with Reset Response** – If you want to allow the server to close the session after the session has finished sending data. Once the server has closed the session, the client will send a RST.
  - Enter an integer between 0 – 1,000,000 in the **Duration(s)** field. This represents the duration of the steady-state phase.
  - Enter an integer between 1 – 500,000 (1Gb) or 750,000 (10Gb) in the **Session/s** field. This represents the rate at which connections are being opened and closed. Please note that the maximum number of sessions supported depends on the blade you are using.

- Enter an integer between 1 – 5,000,000 (1Gb) or 20,000,000 (10Gb) in the **Max Sessions** field. This represents the maximum number of sessions that can be concurrently open at any given time. Please note that the maximum number of sessions supported depends on the blade you are using.
- Click the **Data Rate** drop-down button and select one of the following:
  - Mbps (Megabits per Second)
  - Frames/ps (Frames per Second)
- Enter an integer value between 1 – 10,000 (10Gb) or 1,000 (1Gb) (for Mbps) or 1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) or (for fps) in the **Data Rate** field.

- If you are cloning a steady-state phase, do the following:
  - Select an existing steady-state phase from the **Phase Settings** section.
  - Click the **Duplicate Phase** button.
  - Click the **Steady-State Behavior** drop-down button and select one of the following:
    - **Open and Close Sessions** – If you want existing sessions to close as soon as they finish sending data and new sessions to open in their place.
    - **Hold Sessions Open** – If you want to keep open all sessions that were opened during the ramp up phase.
    - **Open and Close with Reset** - If you want to initiate the TCP close with a RST. This bypasses the TCP close state machine.
    - **Open and Close with Reset Response** – If you want to allow the server to close the session after the session has finished sending data. Once the server has closed the session, the client will send a RST.
  - Enter an integer between 0 – 1,000,000 in the **Duration(s)** field. This represents the duration of the steady-state phase.
  - Enter an integer between 1 – 500,000 (1Gb) or 750,000 (10Gb) in the **Session/s** field. This represents the rate at which connections are being opened and closed. Please note that the maximum number of sessions supported depends on the blade you are using.
  - Enter an integer between 1 – 5,000,000 (1Gb) or 20,000,000 (10Gb) in the **Max Sessions** field. This represents the maximum number of sessions that can be concurrently open at any given time. Please note that the maximum number of sessions supported depends on the blade you are using.
  - Click the **Data Rate** drop-down button and select one of the following:
    - Mbps (Megabits per Second)
    - Frames/ps (Frames per Second)
  - Enter an integer value between 1 – 10,000 (10Gb) or 1,000 (1Gb) (for Mbps) or 1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) (for fps) in the **Data Rate** field.
- If you are modifying the Ramp Down phase, do the following:
  - Click the **Behavior** drop-down button and select one of the following:
    - **Full Close** – If you want the full TCP close to be performed on all sessions.
    - **Half Close** – If you want the full TCP close to be performed on all sessions, but you want to omit the final ACK.
    - **Reset** – If you want to close all sessions by sending TCP RST (reset) packets.

- **Reset Response** – Respond to a FIN with a RST. This bypasses the TCP TIME_WAIT state.

- Enter an integer between 0 – 1,000,000 in the **Duration(s)** field. This represents the duration of the ramp-down phase.
- Enter an integer between 1 – 500,000 (1Gb) or 750,000 (10Gb) in the **Session/s** field. This represents the rate at which connections are being opened and closed. Please note that the maximum number of sessions supported depends on the blade you are using.
- Enter an integer between 1 – 5,000,000 (1Gb) or 20,000,000 (10Gb) in the **Max Sessions** field. This represents the maximum number of sessions that can be concurrently open at any given time. Please note that the maximum number of sessions supported depends on the blade you are using.
- Click the **Data Rate** drop-down button and select one of the following:
  - Mbps (Megabits per Second)
  - Frames/ps (Frames per Second)
- Enter an integer value between 1 – 10,000 (10Gb) or 1,000 (1Gb) (for Mbps) or 1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) (for fps) in the **Data Rate** field.

9. Repeat step 8 until you have modified and/or added all the desired phases.

10. Click the **Save As** button to save the Load Profile as a new one.

Note:

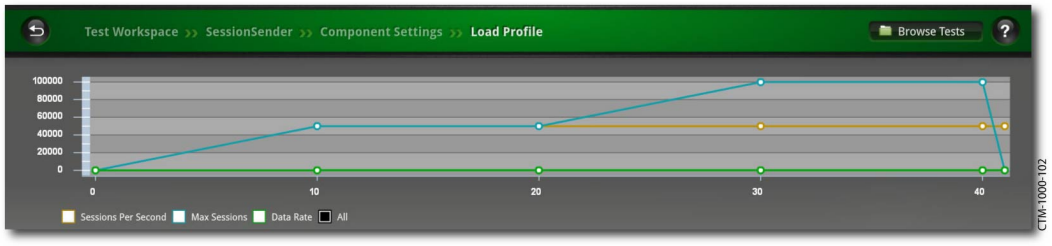
- If you were modifying a non-BreakingPoint Load Profile, you can click the **Save** button; this will override the settings for that particular Load Profile. If you want to save it as a new Load Profile, you should use the **Save As** button.
- To continue editing component parameters, click the Save and Continue button. This will return you to the Parameters section of the Component Settings page.

Load Profile Graph

After the Load Profile has been created, the data configured for each phase will populate the Load Profile graph. You can use this graphical depiction to visualize how the network traffic will appear on the wire – as well as predict what the test results in the reports will look like.

Three statistics are tracked by the Load Profile graph: Max Sessions, Data Rate, and Sessions Per Second. Load Profile Graph below shows the Load Profile graph.

Load Profile Graph

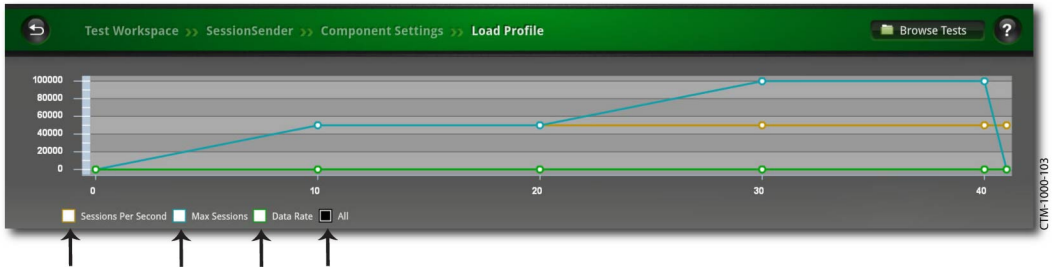


The blue line tracks the maximum number of simultaneous sessions, the orange line tracks the session rate, and the green line tracks the data rate.

If you hover your mouse over data points on the graph, a pop-up window will display, showing the information for that stat at that point in time. This information includes the value for the stat and the point in the test in which the test should reach that value.

To select a specific statistic, select one of the check boxes next to each available statistic (see Load Profile below). There is a set of numbers on the left y-axis and bottom x-axis of the graph. The set of numbers along the left y-axis represents the value of the statistic that you have selected. The set of numbers along the bottom x-axis represents the number of seconds.

Load Profile



This page intentionally left blank.

CHAPTER 16 Packet Buffer

This section covers:

Packet Buffer Overview

BreakingPoint's packet buffer stores all transmitted and received traffic from the last test run. Each port has its own packet buffer with a 2 GB circular buffer limit (4 GB for the BreakingPoint FireStorm). Once the buffer limit is met, the system will overwrite the oldest content on the buffer.

Note: Each time a new test is run, BreakingPoint will overwrite the existing content on the packet buffer with the content from the newest test run.

The traffic capture starts when BreakingPoint begins transmitting traffic. Therefore, slow start packets will not be included in the traffic capture because they are transmitted before BreakingPoint generates traffic. This affects traffic captures for tests running Bit Blaster and Routing Robot.

Manually Stopping the Capture

Since BreakingPoint will automatically start the capture once traffic generation begins, you cannot control when the capture starts; however, you can stop the capture at any time during a test run.

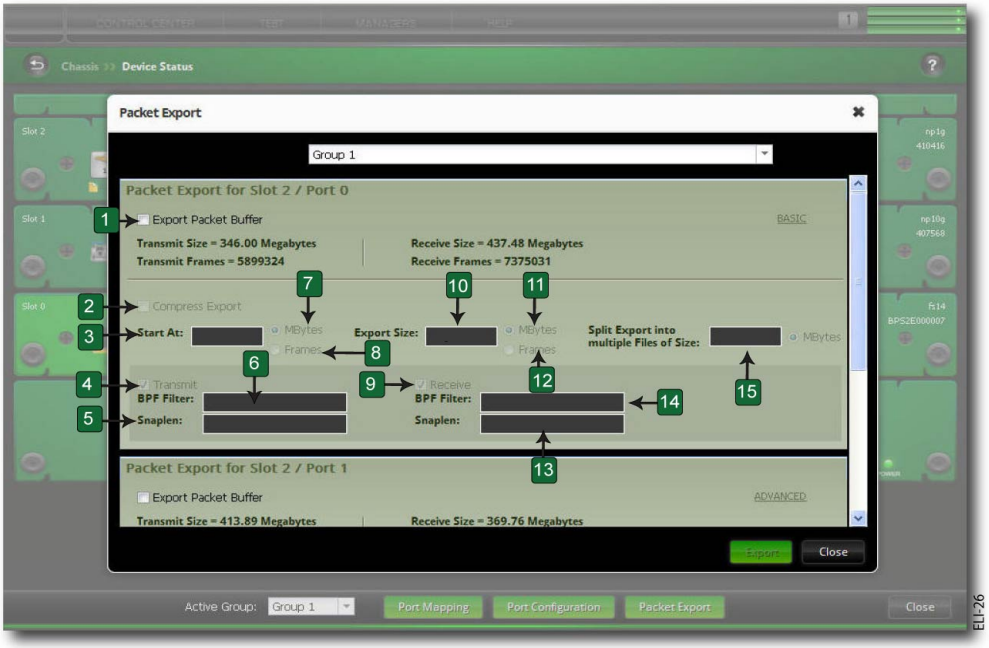
On the Real-Time Statistics screen, there is a **Stop Capture** button; clicking this button will stop the capture. All content stored on the buffer at this point in the test execution can be exported into an external PCAP file. For more information on exporting capture files, see Exporting a Packet Buffer below.

Click **Stop Capture** any point during the test to stop the capture.

Exporting a Packet Buffer

From the Device Status screen, there is a **Packet Export** button that allows you to select the packet buffers you would like to export. Each port has its own packet buffer, so you will need to select the slot (s) and port(s) from which you would like to export content.

Export Packet Buffer



Export Packet Buffer

Callout	Parameter	Description
1	Export Packet Buffer	Select packet buffer to export
2	Compress Export	Select to compress (gzip) the export

Callout	Parameter	Description
3	Start At	Designate a starting point for the export
4	Transmit	Select to export transmitted traffic
5	Snaplen	The amount of data for each frame that is actually exported
6	BPF Filter	Set filtering with BPF syntax
7	Megabytes	Select to start at the size specified
8	Frames	Select to start at the frame specified
9	Receive	Select to export received traffic
10	Export Size	Designate a size for the export
11	Megabytes	Express the desired size of the export in Megabytes
12	Frames	Express the desired size of the export by using the number of frames preferred
13	Snaplen	The amount of data for each frame that is actually exported
14	BPF Filter	Set filtering with BPF syntax
15	Split Export into multiple Files of Size	The total length of the chunk in Megabits

You can do this from the **Export Packet Buffer** window. Additionally from this window, you can select whether you want to export transmitted and/or received traffic.

A capture export is a system process, so only one export can be performed at a time, and you can only export capture files from ports with locked reservations.

Note: The Export Packet Buffer parameters determine the type and number of packets that will be included in your flows. The parameters do not affect any of the characteristics of the original PCAP file. The original PCAP file will still be available for raw playback after it has been exported to your disk.

BPS VE Capture Notes

Minimum memory that must be assigned to support the max packet buffer per port:

Considering the memory allocated for a BPS VE vBlade, the packet buffer limit varies as follows:

- 8 GB RAM (ESXi default allocation) …….. 250 MB*
- 16 GB RAM ……. 500 MB*
- 32 GB RAM …….. 1 GB*

*Please note that the values will be divided between all ports. The values are valid for vBlades deployed on a VMware or KVM hypervisor, as well as for vBlade AMIs deployed on AWS.

Note: Any limitation present in BPS VE will also be present in BPS on AWS.

The maximum packet capture allowed for each vBlade instance type:

Instance memory specifications can be correlated with the memory allocation information provided above to estimate the maximum allowed per vBlade. For example, if Instance type "X" has 8GB of RAM, 250MB can be captured, etc.

Note: Buffer size for packet capture is not dependent on the underlying BPS VE infrastructure (AWS, VMware or KVM).

To export packet buffers:

1. Select **Control Center > Device Status** from the Menu bar.
2. Verify that the ports you would like to export from have locked reservations.
3. To lock a port reservation, simply click on the port. Ports that you have reserved will display a key icon.
4. Click the Packet **Export** button.
5. Select the Export Packet Buffer for Slot check box of each port you would like to export content from.

Only ports with locked reservations will be listed.

6. Click the Advanced arrow and perform any of the following:
   - Select Compress Export if you want to compress the export.
   - Enter a numerical value in the Start At field.
   - Select Megabytes if you want to start exporting at a specific size.
   - Select Frames if you want to start exporting at a specific point in the flow.
   - Enter a numerical value in the Export Size field.
   - Select Megabytes if you want to express the desired size of the export in Megabytes.
   - Select Frames if you want to express the desired size of the export by using the number of frames preferred.

Note: The parameters that you select in the Export Packet Buffer affect the subsequent flows, not the raw PCAP files.

   - Select **Transmit** if you want to export the traffic transmitted by the BreakingPoint system.
   - Set filtering with Berkeley Packet Filtering (BPF) by entering valid BPF syntax in the BPF Filter field.
   - Enter the number of bytes of a given frame you want to export in the Snaplen field.
   - Select **Receive** if you want to export the traffic received by the BreakingPoint system.
   - Set filtering with Berkeley Packet Filtering (BPF) by entering valid BPF syntax in the BPF Filter field.
   - Enter the number of bytes of a given frame you want to export in the Snaplen field.
7. Click **Export**.

The system will display a progress icon over the ports for which you are exporting content. The icon is displayed when the system is placing content into a file. The file will contain PCAPs for each of the ports for which you exported data. Once the process is complete, a Save window will display prompting you to either save or open the file.

Click **Save**.

Navigate to the location where you want to save the exported content.

8. Click **Save**.

Note:

- Packet buffer export operations may be lengthy and export approximately at the rate of between 1.4 MB and 2.3 MB per second.
- When tests that run for three hours or more do not produce enough traffic to rotate the capture buffer, many of the packets at the end of the capture become disordered and display a negative timestamp value. These timestamp values are used to place the buffer into a PCAP file. Negative timestamp values can cause the packet capture to position the packets incorrectly.

Canceling a Packet Buffer Export

Use the **Stop Export** dialog box any time to cancel a running packet buffer export.

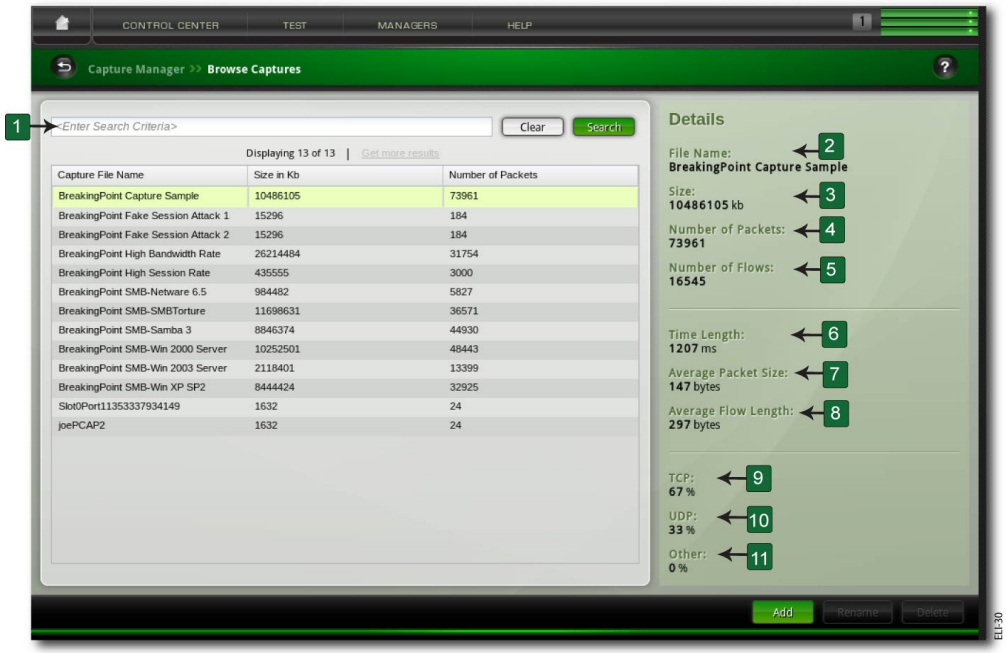
To cancel a packet buffer export:

1. Double-click the Progress icon. The Stop Export dialog box will be displayed.
2. Click Stop. A message asking if you are sure you want to cancel the packet buffer export will be displayed.
3. Click **OK**.

Capture Manager Overview

Use the Capture Manager to add, rename, or delete a capture file. The Capture Manager also has a sort function that allows you to find a PCAP file with a minimum or maximum number of packets or size to build your test. See.

Capture Manager



Capture Manager

Callout	Field	Description
1	Search Field	Allows you to search for available capture files based on `name:`, `protocol:`, or `class:`.
2	File Name	Name of the selected capture file.
3	Size	This value refers to the total number of TCP and UDP payload bytes imported in the capture file. This value does not reflect the size of the original capture file. The maximum size of an imported capture file is currently 700MB. When importing a capture file larger than 700MB, it will still be possible to replay the entire capture file without modifying it.
4	Number of Packets	The total number of packets processed during the capture file import.
5	Number of Flows	The total number of TCP and UDP flows processed during the capture file import.
6	Time Length	The total duration of the capture file in milliseconds.

Callout	Field	Description
7	Average Packet Size	The average size of the packets in the capture file.
8	Average Flow Length	The average length of the flows in the capture file.
9	TCP	The percentage of total packets imported that were TCP.
10	UDP	The percentage of total packets imported that were UDP.
11	Other	The percentage of total packets imported that were fragmented, truncated, or of an invalid size.

Searching for Capture Files

With the Search field of the Capture Manager, you can search for individual capture files based on details such as name, protocol, and class. To perform a search, enter one of the items listed in the following table into the Search field on the Capture Manager page.

contains some of the query strings that can be used to search for specific capture files. Enter these query strings into the search field to narrow your search.

Query Strings For Capture Files

Query Type	Description	Example
`class`	Lists available capture files according to class.	class:exploit
`name`	Lists the names of available capture files containing all or part of the specified criteria.	name:client
`protocol`	Lists available capture files that include the specified protocol.	network:http

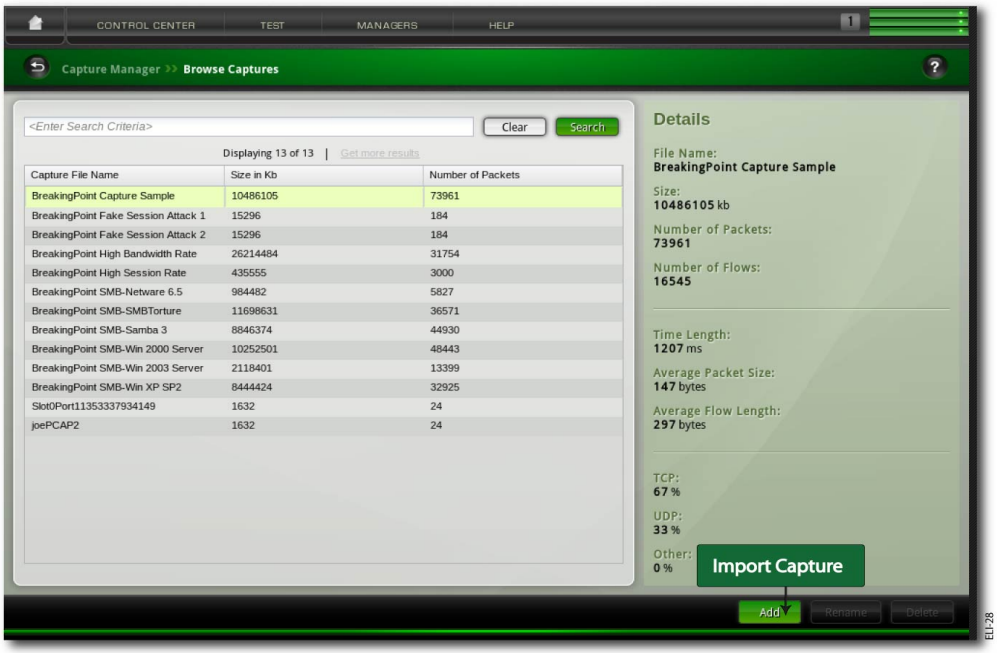
Importing a Capture File

All imported capture files must be a libpcap-compatible PCAP file. Once the capture file has been successfully uploaded, it will be listed under the **Capture File Name** list on the Recreate editor screen and selectable from the **Filename** drop-down menu in the Recreate component's parameters list.

Note:

- There is a 4GB file size limitation on imported PCAPs.
- You can also import a capture file from the **Parameters** section of the Component Settings page of a Recreate Test. From this page, click the Browse button. You can either select a file from the list, or you can click the **Import** button. If you click the Import button, an import window will display, allowing you to browse to and select the capture file you would like to import.

Importing a Capture File



To import a capture file:

1. Select **Managers > Capture Manager** from the Menu bar.
2. Click the **Add** button.

   A new window will display, which will allow you upload a PCAP file.
3. Enter a name in the **Capture Name** field.

   This will be the name displayed for the PCAP in the Capture Manager. Note that capture file names can only contain alphanumeric characters, spaces, and dashes.
4. Click the **Browse** button.
5. Navigate to the location of the PCAP file and select the file.
6. Click the **Open** button.
7. Select the **Allow Overwrite** option if you want to overwrite an existing file with the same name (as defined in the **Capture Name** field).
8. Click the **Upload** button.

Note: Be aware that BreakingPoint pads all frames to 60 bytes. As a result, when you look at a packet capture with Wireshark, a frame that has fewer than 60 bytes will show the original number of bytes as captured while showing 60 bytes on the wire.

The table below lists the settings for the Upload Capture File screen. The Upload Capture File screen is displayed when you click on the Import Capture button.

Upload Capture File Screen Settings

Setting	Description
Capture Name (required)	The name of the Capture file that will be referenced in the test in Parameters/Capture File.
BPF Filter (optional)	A Berkeley Packet Filter expression that will be applied to the capture file during import. Only packets that match the filter will be imported into our internal file format.
Export Size (optional)	Only packets that are under this limit will be imported into our internal file format. The export size can be limited by the number of frames or megabytes.
Allow Overwrite (optional)	Overwrites any existing import with the same name.
Remote URL	Remote location of the capture file in `tcpdump` format (which can also be gzipped) to be imported.
File (required)	The capture file in `tcpdump` format (which can also be gzipped) to be imported.

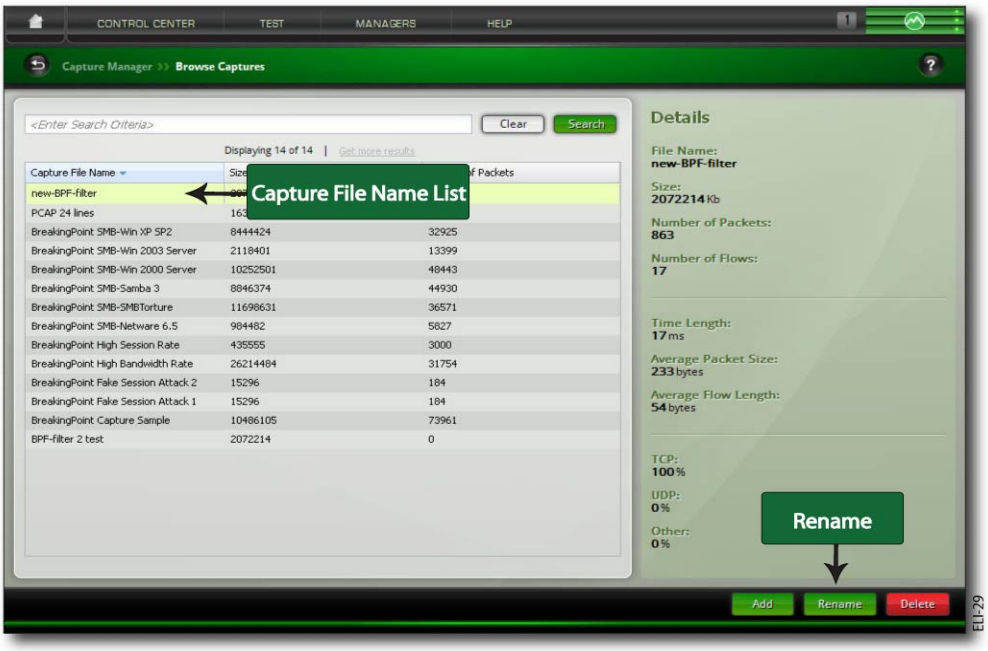
Renaming a Capture File

All imported capture files can be renamed. All capture file names can consist of alphanumeric characters and spaces, and its name cannot exceed 150 characters.

If the capture file is in use by a test, and you rename it, the test will still reference the renamed file. Therefore, you will need to go to the **Parameters** section of the Component Settings page of a Recreate Test and select a new capture file for the test to use.

If you do not select a new capture file and you attempt to run the test, the system will alert you that the file does not exist.

Renaming a Capture File



To rename a capture file:

1. Select **Managers > Capture Manager** from the Menu bar.
2. Select the capture file you would like to rename from the **Capture File Name** list.
3. Click the **Rename** button.
4. Enter the new name in the Capture File Name field.
5. Click the Update button.

Deleting a Capture File

All imported capture files can be deleted from the system. The system will display a warning if the capture file is currently in use by any components. If a capture is in use, and it is deleted, the test will still reference the deleted file. Therefore, you will need to go to the **Parameters** tab of the Recreate component and select a new capture file for the test to use.

If you do not select a new capture file, and you attempt to run the test, the system will alert you that the deleted capture file does not exist.

To delete a capture file:

1. Select **Managers > Capture Manager** from the Menu bar.
2. Select the capture file from the **Capture File Name** list.
3. Click the **Delete** button.
4. Click the **Yes** when the confirmation window displays.

Packet Filter

The Packet Filter feature allows you to selectively export specific packets from the buffer rather than having to export all of the packets in the entire dataset. This feature allows the most efficient usage of the interface card's capture history. Packet Filters are set on a per-port basis, and will process packets as they are received and only capture the packets that you have chosen to capture. For detailed instructions on packet filtering, Packet Filtering on page 190.

Capture Functionality on CloudStorm

The section describes Port Capture user interface functionality on CloudStorm (in FanOut modes: 8x10G, 8x25G) .

Selecting Ports for Capture

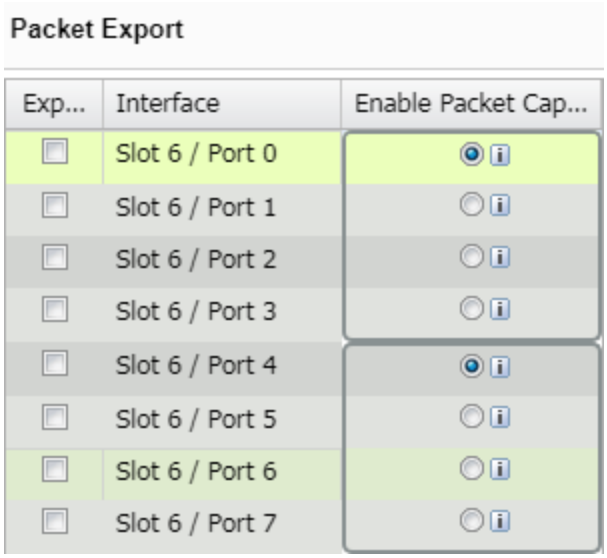
For CloudStorm - FanOut modes: 8x10G, 8x25G:

- Ports are split into 2 Resource Groups
- Each Resource Group contains 4 ports
- 1 port in each Resource Group can be selected for capturing

In the Packet Export window, click **Enable Packet Capture** on the port (see the image below). Select up to 1 port per Resource Group.

Note: If the **Enable Packet Capture** radio button is grayed out/not available, it is because the active port is not currently assigned to the current user.

CloudStorm UI

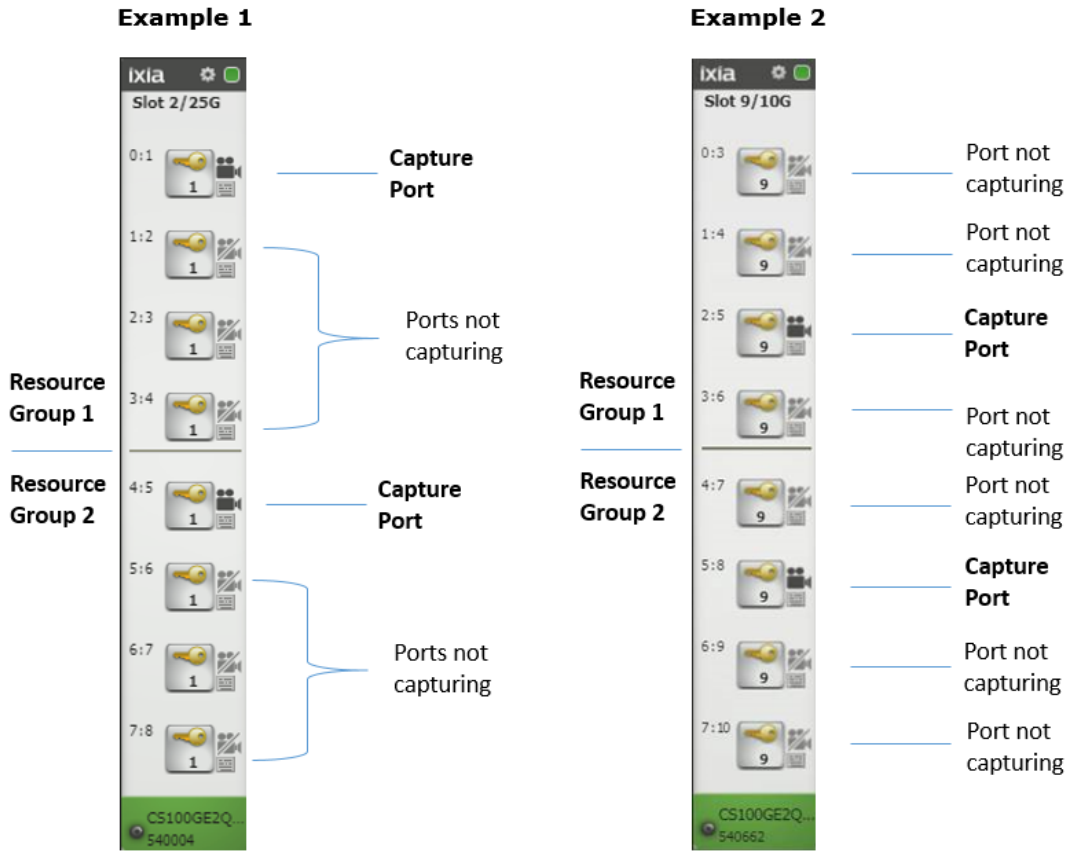


Displayed Port Capture Status

Note:

- In all scenarios other than described below, capture/non-capture icons are not displayed in the Device Status view because all BPS chassis ports allow for capture.
- Capture is not supported on BPS VE at this time.

Device Display when Capturing on CloudStorm (FanOut modes: 8x10G, 8x25G)



- Ports are split into 2 Resource Groups
- Each Resource Group contains 4 ports
- 1 port in each Resource Group can be selected for capturing

CHAPTER 17 Test Components

This section covers:

Test Components Overview

Test components are virtual devices that enable you to test how well your device will operate at different network layers. Each test component comes with a set of parameters, which you can use to create the type of traffic you want.

The following sections will:

- Explain how you can choose the right test component for your testing needs. For more information on choosing test components, see the Choosing Test Components on the next page section.
- Describe the restrictions that must be considered when using test components. For more information on test component restrictions, see the section Restrictions on page 665.
- Provide descriptions of each test component available with the BreakingPoint Storm.

The table below lists the available test components and provides a description of each.

Test Components

Test Component	Description
Application Simulator	The Application Simulator test component allows you to generate application traffic flows. This test component should be used in conjunction with other test components to simulate real-world traffic. The Application Simulator test component uses an App Profile to determine what types of application flows to send to the DUT. The Application Simulator provides a reasonable starting-point for a custom test. The stack parameters are tuned a bit less aggressively than other presets, but still achieves high bandwidth and packet rates.
Bit Blaster	The Bit Blaster test component measures the raw throughput of a device. It analyzes a device's ability to handle high speed traffic by identifying whether or not the DUT receives and sends packets without corrupting or dropping them.

Test Component	Description
Client Simulator	Client Simulator enables you to connect BreakingPoint to a server device under test so the chassis can act as a client generating connections to the server. Client Simulator sends a single Super Flow to the DUT and enables you to search for specific responses from the DUT.
Recreate	This component recreates captured traffic patterns. The Recreate test component recreates traffic in two different modes: Normal and Raw Playback. In Normal mode, the TCP and UDP payloads from the imported capture file are the only portions that will not be modified. The Recreate test component rewrites the data to match the traffic parameters specified for the domain.
Routing Robot	This component measures a device's ability to accurately route IP packets. It sends IP packets with a UDP payload, and measures whether the packets reach their intended destination.
Security	The Security test component can be used to test network security devices – such as IPS, IDS, and firewalls. This component measures the device's ability to protect a host by sending Strikes and verifying whether the device successfully blocks the attacks. It realistically simulates the attacker as well as the responses from the host being attacked.
Session Sender	The Session Sender test component measures a device’s ability to set up and maintain a large number of TCP sessions over a period of time. Each session uses a unique combination of source addresses, destination addresses, source ports, and destination ports; therefore, there must be enough MAC/network address combinations allotted in the domain and enough source/destination port combinations to create that many sessions.
Stack Scrambler	This component measures the device's ability to handle invalid IP, TCP, UDP, ICMP, and Ethernet packets. It sends invalid IP packets produced by fuzzing, and validates whether the device continues to operate.

Choosing Test Components

BreakingPoint allows you to create test scenarios using either a single component or a combination of multiple components. Usage of multiple components allows you to simultaneously test your device using a diverse mix of traffic. Whether you want to send application traffic, mixed with attack traffic, or simply send Layer 2 and 3 traffic, BreakingPoint has a component for every scenario.

To determine which test components to add to your tests, you need to consider the following questions:

What type of network equipment am I testing?

The type of network device you are testing will determine which test components will be best for your test configuration. For example, if you are testing a Layer 2 network switch, you will want to use the Bit Blaster test component to send traffic at line-rate speeds. If you are testing a router, you will want to use the Routing Robot test component to send routable traffic. If you are testing an IPS, you will want to use a combination of test components – such as the Security and Application Simulator test components – to send attacks as well as background application traffic.

Which network layers do I want to test my device at?

The network layer at which you want to test your device will determine which test component(s) you will want to use.

The following test components can be used to generate traffic at different network layers:

- **Layer 2**: Bit Blaster
- **Layer 3**: Routing Robot
- **Layer 4-7**: Session Sender, Application Simulator, Recreate, Client Simulator, and Stack Scrambler

You can use a combination of test components in your tests; however, you must take any restrictions regarding each test component into account. For information on restrictions when using test components, see the section Restrictions below.

What are the goals of testing my device?

Each test component has a set of criteria that determines whether or not the DUT will pass or fail a test. The criteria for each test component will vary depending on the goals of the test component; therefore, it is important that you review each test component's default pass/fail criteria to determine which test component's goals best match your testing needs. You can set your own test criteria if the default pass/fail criteria do not fit your needs. For more information on creating test criteria, see the Test Pass/Fail Criteria on page 821.

Restrictions

This section details the restrictions that can limit the number of test components you can use per test interface. These restrictions are based on bandwidth, hardware resources, and maximum sessions.

Bandwidth

There is a bandwidth limitation for each test interface. The available bandwidth for each interface depends on the DUT's link speed and the type of blade you are using. For example, if you are testing a 10 Gb device and you have a 10 Gb blade, you will have a maximum bandwidth availability of 10,000 Mbps. If you are testing a 1 Gb device and you have a 1 Gb blade, you will have a maximum bandwidth availability of 1,000 Mbps.

The available bandwidth resources are used up by two factors: the data rate set for each test component and the test components you are using. The data rate is the maximum speed at which

traffic can be transmitted to the device. The value set for this parameter will reduce the available bandwidth by that value. For example, if you are testing a 10 Gb device, and you set the data rate to 1,000 Mbps, you will have 9,000 Mbps left to distribute to the other test components on that interface. If you are testing a 1 Gb device, and you set the data rate to 100 Mbps, you will have 900 Mbps left to distribute to the other test components on that interface.

Some test components will only require bandwidth on the transmitting interfaces but will not use up any bandwidth on the receiving interfaces. These components are Bit Blaster, Routing Robot, and Stack Scrambler.

Note:

- If the bandwidth for a test interface is oversubscribed, or using more bandwidth than there is available, check the data rate distribution for each test component on that interface. The sum of the rate distribution values for all test components on the interface should not exceed the bandwidth that is available.
- When components are added to a test or an existing test is opened, BPS will validate that the configured bandwidth settings such as Maximum Data Rate are less than or equal to the achievable aggregate bandwidth of the ports reserved for the test. This validation does not occur if the port group in use has no reserved ports.

BreakingPoint System Hardware Resources

The BreakingPoint system allots a subset of its hardware resources to the test components. Each test component belongs to one of these subsets of resources, which determine the number of components you can add to a test.

Bit Blaster and Routing Robot

Note: To use Bit Blaster or Routing Robot, a load module must be placed in L2/L3 mode.

The following tables describe supported Bit Blaster and Routing Robot component configurations.

Routing Robot 10G Component Support on PerfectStorm

Load Module	Number of Ports	10 G Component Support
10G	8 ports at 1G or 8 ports at 10G	1 x 10G component
40G	2 ports at 40G or (fanout mode) 8 ports at 10G	1 x 40G component or 4 x 10G components
100G	1 port at 80G or 2 ports at 40G or (fanout mode) 8 ports at 10G	1 x 80 G component or 8 x 10G compents

Bit Blaster 10 G Component Support on PerfectStorm

Load Module	Number of Ports	10 G Component Support
10 G	8 ports at 1G or 8 ports at 10G	1 x 10G component
40 G	2 ports at 40G or (fanout mode) 8 ports at 10G	4 x 10G components
100 G	1 port at 80G or 2 ports at 40G or (fanout mode) 8 ports at 10G	8 x 10G components

Maximum Number of Bit Blaster and Routing Robot Components on Storm and FireStorm

Component	Max Number of Components per 1Gb Blade	Max Number of Components per 10Gb Blade
Bit Blaster	8	4
Routing Robot	8	4

When creating your tests, keep in mind that Bit Blaster and Routing Robot can have one or more transmitting (client) interfaces. In addition, each transmitting (client) interface can be used by one or more Bit Blaster or Routing Robot component.

Routing Robot tests that are run with Network Neighborhood configurations have an internal limit of four VLAN tags per test. If you configure your test to run with more than four VLAN tags, only four of the VLAN tags will be recognized by Routing Robot and the results of your test will reflect the packets on those four VLAN tags only.

Note: IMIX mode cannot be run concurrently with Routing Robot using VLAN tags.

Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender

Regardless of the type of network, each pair of ports on the BreakingPoint Storm can accommodate up to 10 components. Each BreakingPoint blade can support up to 20 Session Sender, Application Simulator, Client Simulator, Stack Scrambler, and/or Recreate components. Dynamic-host networks include those configured with either LTE-, GTP-, DHCP- or Multicast-based subnets. All other networks are considered to be non-dynamic-host networks.

The table below lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a dynamic-host network for the BreakingPoint Storm.

Maximum Number of Components In a Dynamic-Host Network

Component	Max Number of Components Per Pair of Ports	Max Number of Components Per Blade
Application Simulator	10	20
Client Simulator	10	20
Recreate	10	20
Session Sender	10	20
Stack Scrambler	10	20

The table below lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a non-dynamic-host network for the BreakingPoint Storm.

Maximum Number of Components In a Non-Dynamic-Host Network

Component	Max Number of Components Per Pair of Ports	Max Number of Components Per Blade
Application Simulator	10	20
Client Simulator	10	20
Recreate	10	20
Session Sender	10	20
Stack Scrambler	10	20

However, please note that the number of these components that can be added to a test is restricted by the maximum number of sessions defined for each component. A 10Gb system allows up to 20,000,000 sessions between these components across all ports on a slot. The number of sessions can be distributed between multiple components as long as they do not exceed 20,000,000 sessions total.

Security

Each pair of ports on the BreakingPoint Storm can accommodate up to 4 Security components. Each BreakingPoint Storm blade can support up to 4 Security components.

The following video link provides a tutorial on how to Add a Security test:
https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

The table below lists the maximum number of Security components for each pair of ports and per each BreakingPoint Storm blade.

Max Number of Security and Stack Scrambler Components

Component	Max Number of Components per Pair of Ports	Max Number of Components per Blade
Security	4	4

Malware

Each pair of ports on the BreakingPoint Storm can accommodate up to 4 Malware components. Each BreakingPoint Storm blade can support up to 4 Malware components.

The following table lists the maximum number of Malware components for each pair of ports and per each BreakingPoint Storm blade.

Max Number of Malware

Component	Max Number of Components per Pair of Ports	Max Number of Components per Blade
Malware	4	4

BreakingPoint FireStorm Hardware Resources

The BreakingPoint FireStorm allots a subset of its hardware resources to the test components. Each test component belongs to one of these subsets of resources, which determine the number of components you can add to a test.

Bit Blaster and Routing Robot

Each BreakingPoint FireStorm 10G blade provides four ports at 10 Gbps each, while each FireStorm 1G blade provides eight ports at 1 Gbps each. Either blade can support BitBlaster and/or Routing Robot components.

Routing Robot tests that are run with Network Neighborhood configurations have an internal limit of four VLAN tags per test. If you configure your test to run with more than four VLAN tags, only four of the VLAN tags will be recognized by Routing Robot and the results of your test will reflect the packets on those four VLAN tags only.

Note: IMIX mode cannot be run concurrently with Routing Robot using VLAN tags.

Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender

Depending on the type of network (dynamic-host or non-dynamic-host) being used in the test, each pair of ports on the BreakingPoint FireStorm can accommodate up to 20 components. Each BreakingPoint FireStorm blade can support up to 40 Session Sender, Application Simulator, Client Simulator, Stack Scrambler, and/or Recreate components. Dynamic-host networks include those configured with either LTE-, GTP-, DHCP- or Multicast-based subnets. All other networks are considered to be non-dynamic-host networks.

The table below lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a dynamic-host network for the BreakingPoint FireStorm.

Maximum Number of Components In a Dynamic-Host Network

Component	Max Number of Components Per Pair of Ports	Max Number of Components Per Blade
Application Simulator	10	40
Client Simulator	10	40
Recreate	10	40
Session Sender	10	40
Stack Scrambler	10	40

The table below lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a non-dynamic-host network for the BreakingPoint FireStorm.

Maximum Number of Components In a Non-Dynamic-Host Network

Component	Max Number of Components Per Pair of Ports	Max Number of Components Per Blade
Application Simulator	20	40
Client Simulator	20	40
Recreate	20	40
Session Sender	20	40
Stack Scrambler	20	40

However, please note that the number of these components that can be added to a test is restricted by the maximum number of sessions defined for each component. The system allows up to 15,000,000 sessions between these components across all ports on a slot. The number of sessions can be distributed between multiple components as long as they do not exceed 15,000,000 sessions total.

Security

Each pair of ports on the BreakingPoint FireStorm can accommodate up to 4 Security components. Each BreakingPoint FireStorm blade can support up to 4 Security components.

lists the maximum number of Security components for each pair of ports and per each BreakingPoint FireStorm blade.

Max Number of Security Components

Component	Max Number of Components Per Pair of Ports	Max Number of Components Per Blade
Security	4	4

Ixia PerfectStorm Fusion Hardware Resources

The PerfectStorm Fusion allots a subset of its hardware resources to the test components. Each test component belongs to one of these subsets of resources, which determine the number of components you can add to a test.

Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender

The PerfectStorm Fusion chassis can accommodate up to 120 components. Each PerfectStorm Fusion load module can support up to 60 Session Sender, Application Simulator, Client Simulator, Stack Scrambler, and/or Recreate components.

lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a dynamic-host network for the PerfectStorm Fusion.

Note: Dynamic-host networks include those configured with either LTE-, GTP-, DHCP- or Multicast-based subnets. All other networks are considered to be non-dynamic-host networks.

Maximum Number of Components In a Dynamic-Host Network

Component	Max Number of Components Per Load Module
Application Simulator	60
Client Simulator	60
Recreate	60
Session Sender	60
Stack Scrambler	60

lists the maximum number of Application Simulator, Client Simulator, Recreate, Stack Scrambler, and Session Sender components in a non-dynamic-host network for the PerfectStorm Fusion.

Maximum Number of Components In a Non-Dynamic-Host Network

Component	Max Number of Components Per Load Module
Application Simulator	30
Client Simulator	30

Component	Max Number of Components Per Load Module
Recreate	30
Session Sender	30
Stack Scrambler	30

However, please note that the number of these components that can be added to a test is restricted by the maximum number of sessions defined for each component. The system allows up to 20,000,000 sessions between these components across all ports on a slot. The number of sessions can be distributed between multiple components as long as they do not exceed 20,000,000 sessions total.

Security

Each pair of ports on the PerfectStorm Fusion can accommodate up to 4 Security components. Each PerfectStorm Fusion load module can support up to 4 Security components.

The table below lists the maximum number of Security components for each pair of ports and per each PerfectStorm Fusion load module.

Max Number of Security and Stack Scrambler Components

Component	Max Number of Components per Load Module
Security	4

Malware

Each pair of ports on the PerfectStorm Fusion can accommodate up to 4 Malware components. Each PerfectStorm Fusion load module can support up to 4 Malware components.

lists the maximum number of Malware components for each pair of ports and per each PerfectStorm Fusion load module.

Max Number of Malware

Component	Max Number of Components per Load Module
Malware	4

Delayed Start

Each test component has a parameter called **Delay Start** that enables you to delay the start of a component by specific amount of time. When the test starts, it will first start the components whose Delay Start values are 0. Then, it will wait for the time defined for Delay Start before running the test component whose Delay Start values are not 0.

Statistic Detail

The Statistic Detail feature allows you to adjust the level of statistics that are collected during your test run. Decreasing the number of statistics collected can allow for an increase in performance. Conversely, when the number of statistics being collected is increased, performance can be adversely affected. Support for the Statistic Detail feature is present in Application Simulator, Client Simulator, Recreate, Session Sender, and Stack Scrambler components.

The Statistic Detail feature has four settings:

- Maximum – Enables all possible statistics.
- Application Only – Enables only Application (L7) statistics.
- Transport Only – Enables only Transport (L4/L3) statistics.
- Minimum – Disables most statistics.

The Real-Time Statistics page will display specific statistics when certain settings of the Statistic Detail feature are enabled.

lists the Real-Time Statistics that are displayed when each Statistic Detail setting is enabled.

Note: These statistics will also be contained in the resulting report when the corresponding settings are enabled.

Statistic Detail Settings

Statistic Detail Setting	Available Real-Time Statistics
Application Only	Application Transaction Rate
	Application Transactions
	Super Flows
	Bandwidth
	Frame Rate
	Cumulative Frames

Statistic Detail Setting	Available Real-Time Statistics
Transport Only	Super Flows
	TCP Flows
	UDP Flows
	SCTP Flows
	TCP Connection Rate
	Cumulative TCP Connections
	Average TCP Time
	Bandwidth
	Frame Rate
	Cumulative Frames
Minimum	Super Flows
	Bandwidth
	Frame Rate
	Cumulative Frames

Bit Blaster

The Bit Blaster test component analyzes a device's ability to handle high speed traffic by identifying whether or not the DUT receives and sends packets without corrupting or dropping them.

The Bit Blaster component only transmits layer 2 frames, which means that it can only be used in a switching environment. If the Bit Blaster component is used in a routing or NAT environment, the component will fail and the resulting report will display an error description stating that the DUT was attempting to execute routable traffic. If you want to generate high-speed, routable traffic, you should use the Routing Robot test component.

Note: The Bit Blaster component will fail in a routing or NAT Network Neighborhood. Use the Routing Robot test component to generate routable traffic.

Note: On Storm and FireStorm there can be up to 4 (8 on 1Gb units) Routing Robot and/or Bit Blaster components per slot. Bit Blaster can have multiple transmitting (client) interfaces. For more information on Bit Blaster component restrictions, see the BreakingPoint System Hardware Resources on page 666.

Payload

The data portion of the payload starts after the Ethernet header. The data portion of the payload can be defined by configuring any of the Payload parameters listed under the Parameters area.

Note: Packets generated by the Bit Blaster test component will reserve the last 16 bytes of the payload for internal use by BreakingPoint Systems. These bytes will not contain the payload value that you have defined, if any.

Slow Start

The Slow Start parameter allows you to specify whether the Bit Blaster test component can send a small amount of traffic to the DUT before reaching the full rate of the test. This ensures that switching devices can identify which port to send traffic on.

If the Slow Start parameter is enabled, It will slow start the total number of MAC/IP pairs that are used during the test. This will enable the Bit Blaster test component to support any number of MAC/IP tuples.

Note: All slow start packets are counted in the Traffic Overview graph.

Note: If the test component measures test duration in frames, or the test component uses a constant data rate and frame size, then the length of the test will be adjusted to account for any slow start packets that were sent.

Incrementing the Frame/Packet Size

The incrementation rate refers to the number of bytes that a frame/packet size is incremented or decremented for a set time increment. For example, you can increment the frame size by 10 bytes every 20 seconds or decrement the packet size 10 bytes every 20 seconds.

The incrementation rate is only applicable if **Size Distribution.Size Distribution Type** is set to **Range** and values have been defined for **Size Distribution.Minimum Frame/Packet Size** and **Size Distribution.Maximum Frame/Packet Size**. The test will start by using the minimum frame/packet size and increment towards the maximum frame/packet size.

The following section will provide an example of an incrementing frame size. For an example of a decrementing frame size, see the Decrementing Frame Size Example on page 685.

Incrementing Frame Size Example

Set the **Size Distribution.Increment N Bytes** parameter to 10 bytes and the **Size Distribution.Every N Seconds** parameter to 20 seconds. This means that the frame size will be incremented by 10 bytes every 20 seconds until the maximum frame size has been met or until the test duration elapses.

The following table lists the values for the parameters used in this example. The minimum frame size to was set to 64 bytes, and the maximum frame size was set to 540 bytes. Every 20 seconds, the frame

size will be incremented by 10 bytes. The frame size will continue to be incremented until it either reaches the maximum frame size of 1,024 bytes or 60 seconds have elapsed.

Incrementing Frame Size Example

Parameter	Value
Size distribution.Size Distribution Type	Range
Size distribution.Minimum Frame/Packet Size	64
Size distribution.Maximum Frame/Packet Size	1,024
Size distribution.Increment N Bytes	10
Size distribution.Every N Seconds	20
Test Duration.Test Duration Measured by a Time Interval	60 Seconds
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.	

The following table lists the results for this example. By the end of the test, the frame size has reached 94 bytes.

Results from the Incrementing Frame Size Example

Time	Frame Size
0	64
20	74
40	84
60	94

Incrementing the Data Rate

The incrementation rate refers to the rate at which the data rate is incremented or decremented over a set period of time. For example, the data rate can be incremented by 50 Mbps every 10 seconds.

The incrementation rate is only applicable if **Data Rate.Data Rate Type** is set to **Range** and values have been defined for **Data Rate.Minimum Data Rate** and **Data Rate.Maximum Data Rate**. The test will start by using the minimum data rate and increment towards the maximum data rate.

Note: Bit Blaster is not restricted to transmit from one port to another port. One component can have one or more transmitting ports and can have one or more receiving ports. From any transmitting port to any receiving port, there is a data path. The data rate configured for the component is the limit on each of the data paths.

Incrementing Data Rate Example

Set the **Data Rate.Increment N Units/Period** parameter to 50 Mbps and the **Data Rate.Every N Seconds** parameter to 10 seconds. This means that the data rate will be incremented by 50 Mbps every 10 seconds until the maximum data rate has been met or until the test duration elapses.

The following table lists the values we have defined for the parameters used in this example. We have set the minimum data rate to 100 Mbps bytes and the maximum data rate to 900 Mbps. Every 10 seconds, the rate will be incremented by 50 Mbps. The data will continue to be incremented until it either reaches the maximum data rate of 900 Mbps or 60 seconds have elapsed.

Incrementing Data Rate Example

Parameter	Value
Data Rate.Data Rate Type	Range
Data Rate.Minimum Data Rate	100
Data Rate.Maximum Data Rate	900
Data Rate.Increment N Units/Period	50
Data Rate.Every N Seconds	10
Test duration.Test Duration Measured by a Time Interval	60 Seconds
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.	

The following table lists the results for this example. By the end of the test, the frame size has reached 400 Mbps.

Results for the Incrementing Data Rate Example

Time	Data Rate
0	100
10	150

Time	Data Rate
20	200
30	250
40	300
50	350
60	400

Bit Blaster Parameters

The following table lists the parameters for the Bit Blaster test component.

Bit Blaster Parameters

Parameter	Description	Valid Values
Test Duration.Test Duration Measured by a Time Interval	Specify the test duration in time.	hh:mm:ss
Test Duration.Test Duration Measured in Frames *	Specify the test duration in number of frames.	1 – 1,000,000,000
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use to send its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.

Parameter	Description	Valid Values
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data rate type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Increment N/Period	Sets the rate at which the data rate will increase or decrease. This parameter is used in conjunction with **Data Rate. Every N Seconds**.	(10Gb): -10,000 to 10,000 (1Gb):-1,000 to 1,000
Data Rate.Every N Seconds	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with **Data Rate. Increment N units**.	1 – 30
Data Rate.Data Rate Ramp	Indicates what to do when reaching the maximum or minimum range value.	Wrap – Resets the data rate to the initial rate value at the start of the test Limit – Holds the terminating data rate until the end of the test
Size Distribution.Size Distribution Unit	Sets whether Bit Blaster uses frame or packets.	Packet or Frame

Parameter	Description	Valid Values
Size Distribution.Size Distribution Type	Sets how the component determines the frame/packet sizes it will use in its traffic.	Constant – Uses **Size distribution.Minimum value** for the frame/packet size. Mix Distribution – Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size. Random – Selects a random value between **Size distribution.Minimum value** and **Size distribution.Maximum value** for the frame/packet size. The size of the packet/frame will be randomly selected for every second of the test. Range – Starts at **Size distribution.Minimum value** and increments until it reaches **Size distribution.Maximum value**. Once the maximum value is met, the packet/frame size will restart at the minimum value.
Size Distribution.Minimum Frame/Packet Size	Sets the minimum frame/packet size, if **Size distribution.Size distribution type** is set to **Constant**. Otherwise, this is the minimum value used if **Size distribution.Size distribution type** is set to **Range** or **Random**.	64 – 9216 bytes (frames) 46 – 9198 bytes (packets)
Size Distribution.Maximum Frame/Packet Size	Sets the maximum frame/packet size; this parameter is used only if **Size distribution.Size distribution type** is set to **Range**.	64 – 9216 bytes (frames) 46 – 9198 bytes (packets)
Size Distribution.Increment N Bytes	Sets the number of bytes to increase or decrease the packet size by; this parameter is used in conjunction with **Size distribution.Every N Seconds**.	-128 to 128

Parameter	Description	Valid Values
Size Distribution.Every N Seconds	Sets the time increment (in seconds) for increasing or decreasing the packet size; this parameter is used in conjunction with **Size Distribution.Increment N units**.	1 – 30
Size Distribution.Length of Mix Distribution	Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size.	64 – 9216 (frames) 46 – 9198 (packets)
Size Distribution.Weight of Mix Distribution	Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size.	64 – 9216 (frames) 46 – 9198 (packets)
Payload.Type	Sets how the component determines the payload it will use for its traffic.	0 – Payload is 0s. 1 – Payload is all 1s. Random – Payload is defined using random Hex values. Increment – Payload is defined using ascending values starting at 0. Decrement – Payload is defined using descending values starting at 0xff. User-Defined – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
Payload.Data Width	Defines the width of the data (in bits) being inserted into the payload.	8, 16, or 32
Payload.User Defined Data	Defines the payload; this parameter is defined only if **Payload.Type** is set to **User Defined**. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)

Parameter	Description	Valid Values
Advanced Options - Payload.UDF Mode	Sets how the component overwrites the existing payload.	Disabled – No data or counter is inserted.Counter – Inserts a 1-to-4 byte counter that increments every frame. The counter uses the value defined for UDF length. The parameters Payload.UDF offset and Payload.UDF length must be defined to use this option. Random – Inserts a 1-to-end-of-payload sequence of random values. The parameters Payload.UDF data width, Payload.UDF length, and Payload.UDF offset must be defined to use this option. Increment – Increments the payload starting at 0. Inserts a 1-to-end-of-payload sequence of incrementing values using an 8, 16, or 32 bit width. The parameters Payload.UDF data width, Payload.UDF length, and Payload.UDF offset must be defined to use this option. Decrement – Decrements the payload starting at 0xff. Inserts a 1-to-end-of-payload sequence of decrementing values using an 8, 16, or 32 bit width. The parameters Payload.UDF data width, Payload.UDF length, and Payload.UDF offset must be defined to use this option.
Advanced Options - Payload.UDF Offset	Defines the number of bytes from the beginning of the payload to place the UDF data.	0 – 9,201
Advanced Options - Payload.UDF Length	Defines the UDF length (in bytes).	1 – 9,202
Advanced Options - Payload.UDF Data Width	Defines the width of the data (in bits) being incremented or decremented.	8, 16, or 32

Parameter	Description	Valid Values
Advanced Options.Ethernet Type Field	Sets how the component will define the Ethernet Length Type field.	Constant – Uses the value defined for **Ethernet type value** in the Ethernet Length Type field. Length – Uses the packet length in the Ethernet length/type field. Bit Blaster will only substitute the Ethernet length/type field with the packet's length if the packet is less than or equal to 1,500 bytes in length and VLAN tagging is not used.
Advanced Options.Ethernet Type Value	Determines what value will be placed in the Ethernet Length Type field. This is defined only if the **Ethernet type field** is set to **Length**. You must only used valid Hex values; do not input VIDs or invalid values, or the system will encounter received frames error counts.	2E – FFFF Values less than 2E will be replaced with 2E.
Advanced Options.Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
Advanced Options.Bidirectional	Originate traffic from both the client and server interfaces.	Check for Yes, uncheck for No
Advanced Options.Slow Start	Specifies whether the component can send a small amount of traffic to the DUT before ramping up to the full rate of the test. This allows switching devices to identify which port to send test traffic.	Check for Yes, uncheck for No
Advanced Options.Slow Start Rate	Sets the rate of the slow rate traffic. The rate specified represents the number of frames to be generated per second.	0 – 1,000,000

Parameter	Description	Valid Values
Advanced Options.Maximum Stream Count	This override parameter sets the minimum and maximum number of streams to use for this component. If requested MAC/IP addresses are not symmetric, the number of streams can exceed the Maximum Stream Count.	1 – 16,777,216
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.		

Routing Robot

The Routing Robot test component determines if a DUT routes traffic properly by sending routable traffic from one interface and monitoring the receiving interface to see if the traffic is successfully received.

Note: There can be up to 4 (8 on 1Gb units) Routing Robot and/or Bit Blaster components used on each slot. Routing Robot can have multiple transmitting (client) interfaces. For more information on Routing Robot component restrictions, see BreakingPoint System Hardware Resources on page 666

Payload

The Routing Robot test component sends packets with a UDP payload. The data portion of the payload can be defined by configuring any of the Payload parameters listed under the Parameters section.

Note: Packets generated by the Routing Robot test component will reserve the last 16 bytes of the payload for internal use by BreakingPoint Systems. These bytes will not contain the payload value that you have defined, if any.

Slow Start

The Slow Start parameter allows you to specify whether the Routing Robot test component can send a small amount of traffic to the DUT before reaching the full rate of the test. This ensures that routing devices can identify which port to send traffic on.

If the Slow Start parameter is enabled, and the Routing Robot test component will generate more than 64 packets, the test component will send 64 slow start packets prior to the actual test. This will enable the Routing Robot test component to support 64 MAC/IP tuples.

If the Slow Start parameter is enabled, and the Routing Robot test component will generate less than 64 packets, then the test component will not send slow start packets at the beginning of the test.

Note:
- All slow start packets are counted in the Traffic Overview graph.
- If the test component measures the test duration in frames, or the test component uses a constant data rate and frame size, then the length of the test will be adjusted to account for any slow start packets that were sent.

Decrementing the Frame/Packet Size

You can decrement the frame or packet size for a set time increment. For example, you can decrement the frame size by 10 bytes every 20 seconds.

To do this, you must set **Size Distribution.Size Distribution Type** to **Range** and define frame/packet sizes using **Size Distribution.Minimum Frame/Packet Size** and **Size Distribution.Maximum Frame/Packet Size**. The test will start by using the maximum frame size and decrement towards the minimum frame size.

Note: Routing Robot is not restricted to transmit from one port to another port. One component can have one or more transmitting ports and can have one or more receiving ports. From any transmitting port to any receiving port, there is a data path. The data rate configured for the component is the limit on each of the data paths.

The following section will provide an example of a decrementing frame size. For an example of an incrementing frame size and incrementing data rate, see Incrementing the Frame/Packet Size on page 675.

Decrementing Frame Size Example

Set the **Size Distribution.Increment N Bytes** parameter to -10 bytes and the **Size Distribution.Every N Seconds** parameter to 20 seconds. This means that the frame size decrement 10 bytes every 20 seconds until the minimum frame size has been met or until the test duration elapses.

The following table lists the values for the parameters used in this example. The minimum frame size was set to 64 bytes, and the maximum frame size was set to 1,024 bytes. Every 20 seconds, the frame size will decrement by 10 bytes. The frame size will continue to decrement until it either reaches the minimum frame size of 64 bytes or 60 seconds have elapsed.

Decrementing Frame Size Example

Parameter	Value
Size Distribution.Size Distribution Type	Range
Size Distribution.Minimum Frame/Packet Size	64
Size Distribution.Maximum Frame/Packet Size	1,024
Size Distribution.Increment N Bytes	-10
Size Distribution.Every N Seconds	20

Parameter	Value
Test Duration.Test Duration Measured by a Time Interval	00:00:60
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.	

The following table lists the results for this example. By the end of the test, the frame size has reached 34 bytes.

Results from the Decrementing Frame Size Example

Time	Frame Size
0	64
20	54
40	44
60	34

Decrementing the Data Rate

The incrementation rate refers to the rate at which the data rate is incremented or decremented over a set period of time. For example, the data rate can be decremented by 50 Mbps every 10 seconds.

To do this, you must set **Data Rate.Data Rate Type** to **Range** and define frame/packet sizes using **Data Rate.Minimum Data Rate** and **Data Rate.Maximum Data Rate**. The test will start by using the maximum data rate and decrement towards the minimum data rate.

When using IMIX mode, you can enter up to 10 different packet sizes. The sizes are specified as packet length of frame length, just like in other modes.

Note: When the Size Distribution Type is set to Mix, the Data Rate Unit must be set to Megabits/Second. If Frames/Second is selected, the rate distribution will be ignored and interpreted as Megabits/Second.

Each packet size that is specified has an associated weight value. The weight determines the percentage of the overall traffic that will be generated at that specified size. You can specify up to ten different combinations of packet sizes and percentages.

The following section will provide an example of a decrementing data rate. For an example of an incrementing data rate, see Incrementing the Data Rate on page 676.

Decrementing Data Rate Example

Set **Data Rate.Increment N Units/Period** to 50 Mbps and **Data Rate.Every N Seconds** to 10 seconds. This means that the data rate will be incremented by 50 Mbps every 10 seconds until the maximum data rate has been met or until the test duration elapses.

The following table lists the values we have defined for the parameters used in this example. We have set the minimum data rate to 100 Mbps bytes and the maximum data rate to 900 Mbps. Every 10 seconds, the rate will decrement by 50 Mbps. The data will continue to decrement until it either reaches the maximum data rate of 100 Mbps or 60 seconds have elapsed.

Decrementing Data Rate Example

Parameter	Value
Data Rate.Data Rate Type	Range
Data Rate.Minimum Data Rate	100
Data Rate.Maximum Data Rate	900
Data Rate.Increment N Units/Period	-50
Data Rate.Every N Seconds	10
Test Duration.Test Duration Measured by a Time Interval	00:00:60
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.	

The following table lists the results for this example. By the end of the test, the frame size has reached 200 Mbps.

Results for the Decrementing Data Rate Example

Time	Data Rate
0	500
10	450
20	400
30	350
40	300
50	250

Time	Data Rate
60	200

Routing Robot Parameters

The following table lists the parameters for the Routing Robot test component.

Routing Robot Parameters

Parameter	Description	Valid Values
Test Duration.Test Duration Measured by a Time Interval	Specifies the test duration in time.	hh:mm:ss
Test Duration.Test Duration Measured in Frames	Specifies the test duration in frames.	1 – 1,000,000,000
Packet Templates.Delay Start	Delays the start of a test component by the time specified.	hh:mm:ss
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/Second Megabits/Second
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use to send its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.

Parameter	Description	Valid Values
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data rate type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,880,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,880,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Increment N Units/Period	Sets the rate at which the data rate will increase or decrease. This parameter is used in conjunction with **Data Rate. Every N Seconds**.	(10Gb): -10,000 to 10,000 (1Gb): -1,000 to 1,000
Data Rate.Every N Seconds	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with **Data Rate. Increment N units**.	1 – 30
Data Rate.Data Rate Ramp	When Data Rate Type is 'Range', this value indicates what to do when reaching the maximum or minimum range value.	Wrap or Limit
Size Distribution.Size Distribution Unit	Sets whether Routing Robot uses frame or packets.	Packet or Frame Note: Selecting the Packet option will add 20 bytes to the header. Selecting the Frame option will add 14 bytes to the header. Each VLAN added to the network neighborhood will add an additional 2 bytes to the values described above.

Parameter	Description	Valid Values
Size Distribution.Size Distribution Type	Sets how the component determines the frame/packet sizes it will use for its traffic.	**Constant** – Uses **Size distribution.Minimum value** for the frame/packet size. Mix – Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size. **Random** – Selects a random value between **Size distribution.Minimum value** and **Size distribution.Maximum value** for the frame/packet size. The size of the packet/frame will be randomly selected for every second of the test. **Range** – Starts at **Size distribution.Minimum value** and increments until it reaches **Size distribution.Maximum value**. Once the maximum value is met, the packet/frame size will restart at the minimum value. **Mix** - Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size.
Size Distribution.Minimum Frame/Packet Size	Sets the minimum frame/packet size, if **Size distribution.Size distribution type** is set to **Constant**. Otherwise, this is the minimum value used if **Size distribution.Size distribution type** is set to **Range** or **Random**.	64 – 9216 bytes (frames) 46 – 9198 bytes (packets)
Size Distribution.Maximum Frame/Packet Size	Sets the maximum frame/packet size; this parameter is used only if **Size distribution.Size distribution type** is set to **Range**.	64 – 9216 bytes (frames) 46 – 9198 bytes (packets)

Parameter	Description	Valid Values
Size Distribution.Increment N Bytes	Sets the number of bytes to increase or decrease the packet size by; this parameter is used in conjunction with **Size distribution.Every N Seconds**.	-128 to 128
Size Distribution.Every N Seconds	Sets the time increment (in seconds) for increasing or decreasing the packet size; this parameter is used in conjunction with **Size Distribution.Increment N units**.	1 – 3600
Length of Mix Distribution	Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size.	64 – 9216 (frames) 46 – 9198 (packets)
Width of Mix Distribution	Indicates that up to 10 different frame/packet sizes will be used with a weighting factor per size.	64 – 9216 (frames) 46 – 9198 (packets)
Payload.Type	Sets how the component determines the payload it will use for its traffic.	**0** – Payload is 0s. **>1** – Payload is all 1s. **Random** – Payload is defined using random Hex values. **Increment** – Payload is defined using ascending values starting at 0. **Decrement** – Payload is defined using descending values starting at 0xff. **User Defined** – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
Payload.Data Width	Defines the width of the data (in bits) being inserted into the payload.	8, 16, or 32

Parameter	Description	Valid Values
Payload.User Defined Data	Defines the payload; this parameter is defined only if **Payload.Type** is set to **User Defined**. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)
Advanced Options - Payload.UDF mode	Sets how the component will overwrite the existing payload.	**Disabled**– No data or counter is inserted. **Counter**– Inserts a 1-to-4 byte counter that increments every frame. The counter uses the value defined for UDF length. The parameters **Payload.UDF offset** and **Payload.UDF length** must be defined to use this option. **Random** – Inserts a 1-to-end-of-payload sequence of random values. The parameters **Payload.UDF data width**, **Payload.UDF length**, and **Payload.UDF offset** must be defined to use this option. **Increment** – Increments the payload starting at 0. Inserts a 1-to-end-of-payload sequence of incrementing values using an 8, 16, or 32 bit width. The parameters **Payload.UDF data width**, **Payload.UDF length**, and **Payload.UDF offset** must be defined to use this option. **Decrement** – Decrements the payload starting at 0xff. Inserts a 1-to-end-of-payload sequence of decrementing values using an 8, 16, or 32 bit width. The parameters **Payload.UDF data width**, **Payload.UDF length**, and **Payload.UDF offset** must be defined to use this option.
Advanced Options - Payload.UDF offset	Defines the number of bytes from the beginning of the payload to place the UDF data.	0 – 9,173

Parameter	Description	Valid Values
Advanced Options - Payload.UDF length	Defines the UDF length (in bytes).	1 – 9,174
Advanced Options - Payload.UDF data width	Defines the width of the data (in bits) being incremented or decremented.	8, 16, or 32
Advanced Options - IPv4.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
Advanced Options - IPv4.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff
Advanced Options - IPv4.Length Field	Sets how the component determines Total Length field that will be used in the IP header.	**Actual** – Uses the correct IP datagram length in the Total Length field of the IP header. **Constant** – Uses **IPv4.Length value** in the Total Length field of the IP header.
Advanced Options - IPv4.Length Value	Defines the Total Length field of the IP header when **IPv4.Length field** is **Constant**.	0 – 255
Advanced Options - IPv4.Checksum Field	Sets how the component determines the Checksum field that is used in the IP header.	**Actual** – Uses the correct checksum in the Checksum field of the IP header. **Constant** – Uses **IPv4.Checksum value** in the Checksum field of the IP header.
Advanced Options - IPv4.Checksum Value	Defines the Total Length field of the IP header when **IPv4.Checksum field** is **Constant**.	0 – FFFF
Advanced Options - IPv4.Option Header Field	Allows up to 56 bytes of IP option data to be specified. If this parameter is disabled, the UDP header will follow the IPv4 header.	Enabled or Disabled

Parameter	Description	Valid Values
Advanced Options - IPv4.Option Header Data	Defines the IPv4 option data, if **IPv4.Option header field** is **Enabled**.	Hexadecimal value (up to 56 bytes of data)
Advanced Options - IPv6.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
Advanced Options - IPv6.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF
Advanced Options - IPv6.Flow Label	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
Advanced Options - IPv6.Length Field	Sets how the component determines the Payload Length field that is used in the IP header.	**Actual** – Uses the correct IP datagram length in the Packet Length field of the IP header. **Constant** – Uses **IPv6.Length value** in the Packet Length field of the IP header.
Advanced Options - IPv6.Length Value	Defines the Packet Length field of the IP header when **IPv6.Length field** is **Constant**.	0 – 65,535
Advanced Options - IPv6.Extension Header Field	Allows up to 56 bytes to be specified for the IPv6 extension header(s). If this parameter is enabled, IPv6.Next header and IPv6.Extension header data must be defined. If this parameter is disabled, the UDP header will follow the IPv6 header.	Enabled or Disabled

Parameter	Description	Valid Values
Advanced Options - IPv6.Next Header	Defines the Next header in the IPv6 header if **IPv6.Extension header field** is **Enabled**. This is the extension header that will appear first in the Extension header data. Configure this value to 11 to indicate a UDP payload.	0 – ff
Advanced Options - IPv6.Extension Header Data	Defines the IPv6 extension header (s), if **IPv6.Extension header field** is **Enabled**.	Hexadecimal value (up to 56 bytes of data)
Advanced Options - UDP.Length Field	Determines the UDP datagram length that is placed in the length field of the UDP header.	**Actual** – Uses the correct UDP datagram length in the length field of the UDP header. **Constant** – Uses the value defined for **UDP. Length value** in the length field of the UDP header.
Advanced Options - UDP.Length Value	Defines the UDP datagram length that is placed in the length field of the UDP header. This parameter is defined only if **UDP.Length field** is set to **Constant**.	0 – 65,535
Advanced Options - UDP.Checksum Field	Determines the value that is placed into the checksum field of the UDP header.	**Actual** – Uses the correct UDP checksum in the checksum field of the UDP header. **Constant** – Uses the value defined for **UDP.Checksum value** in the checksum field of the UDP header. Using a constant UDP checksum may cause the test results to report invalid IP checksums.
Advanced Options - UDP.Checksum Value	Defines the value that is used in the checksum field of the UDP header. This parameter is defined only if **UDP.Checksum field** is set to **Constant**.	0 – FFFF

Parameter	Description	Valid Values
Packet Templates.Type	Provides alternative packet definitions to the standard UDP packet that is used by the Routing Robot component. They provide a quick method to generate rate based traffic with several different packet types.	ICMP Echo Reply ICMP Echo Request TCP TCP Syn Flood UDP
Packet Templates.Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
Packet Templates.Source Port	Establishes the UDP port from which packets are addressed.	1 – 65,535
Packet Templates.Destination Port	Establishes the UDP port to which packets are addressed.	1 – 65,535
Packet Templates.Source Port Mask Length	Defines how the bits will be masked for each packet. This mask is right-justified and only applies to UDP source ports.	1 – 16
Packet Templates.Destination Port Mask Length	Defines how the bits will be masked for each packet. This mask is right-justified and only applies to UDP destination ports.	1 – 16
Packet Templates.Source Port Modifier	Determines how the UDP source port is modified.	**Constant** – Uses the port value defined for **Source Port**. **Random** – Selects a random port value between 1 and 65,535. **Increment** – Starts at the **Source Port** value and increments the port value by 1. **Decrement** – Starts at the **Source Port** value and decrements the port value by 1.

Parameter	Description	Valid Values
Packet Templates.Destination Port Modifier	Determines how the UDP destination port is modified.	**Constant** – Uses the port value defined for **Source Port**. **Random** – Selects a random port value between 1 and 65,535. **Increment** – Starts at the **Destination Port** value and increments the port value by 1. **Decrement** – Starts at the **Destination Port** value and decrements the port value by 1.
Packet Templates.Bidirectional	Originate traffic from both the client and server interfaces.	True or False
Advanced Options - Enable TCP	Specifies that pre-defined TCP packets will be generated (instead of UDP packets) during the test.	True or False
Packet Templates.Slow Start	Specifies whether the component can send a small amount of traffic to the DUT before ramping up to the full rate of the test. This allows switching devices to identify which port to send test traffic.	Check for Yes, uncheck for No
Packet Templates.Slow Start Rate	Sets the rate of the slow rate traffic. The rate specified represents the number of frames to be generated per second.	0 – 1,000,000
Packet Templates.Maximum Stream Count	This override parameter sets the minimum and maximum number of streams to use for this component. If requested MAC/IP addresses are not symmetric, the number of streams can exceed the Maximum Stream Count.	1 – 16,777,216
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.		

Advanced Routing Robot

The Advanced Routing Robot test component determines if a DUT routes traffic properly by sending routable traffic from one interface and monitoring the receiving interface to see if the traffic is successfully received. Most importantly, the component supports an Advanced Test Path functionality that generates realistic random tuples for component parameters (IPv4 Source IP, IPv4 Destination IP, Source Port, and Destination Port).

Note: This feature is only supported on the CloudStorm platform. While it provides similar functionality as Routing Robot, it is not a superset of the Routing Robot test component.

Configure an Advanced Test Path

From a NN select, **Test Paths** at the bottom left corner. Select the **Advanced** tab. The configuration of an Advanced Test Path is similar to the configuration of a Basic Test Path.

Note: Advanced Test Paths can only be used by the Advanced Routing Robot but will not adversely affect a NN that does not contain an Advanced Routing Robot test component.

The following table lists the Advanced Test Path parameters. Parameter can also be imported using the Path Files parameter described below.

Advanced Test Path Parameters

Parameter	Description	Valid Values
Source Endpoint	Drop-down menu with all IPv4 static hosts.	Select the source tag.
Destination Endpoint	Drop-down menu with all IPv4 static hosts.	Select the destination tag from the drop-down menu.
Tag	String identifier of the Advanced RR component.	Any suitable string combination can be entered. This tag will will be displayed in the component tag selection option of the Advanced RR component.
Source Port Base	Integer value for min source port.	1-65535
Source Port Count	Integer value for source port count.	1-65535 (base source port + source port count < 65536)
Source Port Algorithm		Random, Default or Preferred

Parameter	Description	Valid Values
Destination Port Base	Integer value for min destination port.	Select a value greater or equal to "1" based on the requirement of the tuple.
Destination Port Count	Integer value for destination port count.	Destination Port Base + Destination Port count < 65536
Destination Port Algorithm		Constant or Random
Tuple Limit	Integer value for the max number of tuples to be generated for this tuple set. For example, if you enter "1,000", it means that BPS will populate a table with 1,000 tuples based on the ranges specified in the other Advanced test path settings. Test traffic will "loop" so that the 1st 1,000 packets will be identical to the 2nd 1,000 packets.	1-32768 **Note:** Configuring a large number of tuples may cause Slow Start (described in the Routing Robot parameters) to take longer. **Note:** Since the tuples are well defined for Advanced RR, Slow Start is not necessary. We recommend that Slow Start is disabled.
Balance Control	Used to generate an address table according to a proprietary algorithm. It should be left blank in normal use.	Supported values are 1-10. The option supports the control algorithm of "2^1-2^10" different combinations of XOR values of: (Source port, Destination port, Source IP, and Destination IP).
Paths File	Used to import Advanced Path parameters using a .csv file.	See Paths File Field on the next page

Add an Advanced Routing Robot Test Component

1. From within a Test, in the **Test Components** section, click **Add New**.
2. Select **Advanced RoutingRobot** from the list of test components.
3. Click **Select**.

4. In the **Component Tags** section, select 1 or more Advanced Test Paths (the configured traffic flow of all selected Advanced Test Paths must be in the same direction).

Note: The total Tuple Limit for all selected Advanced Test Paths is 32,768.

5. Click the right arrows (>>) to move the Advanced Test Paths to the Path Tags field.
6. Configure the Parameters (unsupported options are grayed out). The Parameters are described in the Routing Robot Parameters on page 1106, except for:
   - **Tuple Generation Seed** - Used to initialize the random number generator. Defining a seed allows the generation of IP and Port values that are repeatable for each test. Setting the random seed to "\0" will generate dynamic values.

Paths File Field

BPS provides the ability to import a Paths File (table of values) into an Advanced Path configuration using the **Paths File** field of the Advanced Path configuration.

Note: Except for the interfaces specified in the Network Neighborhood, the parameters specified in a Paths file (described below) supersedes the parent Network Neighborhood settings. Each packet transmitted will be generated based explicitly on a line in the Paths File, in sequential order. Therefore, this feature gives you granular control of the contents of your test and the order of the packets that are transmitted, but it also gives you the responsibility for ensuring that the content of your test is correct.

Paths File Format

A Paths File must be formatted as described below.

Note: The TPID value used for VLANs is not specified in the Paths File. For single and double VLAN configurations, the TPID (tag protocol identifier) inserted in the packet will be “0x8100”.

- A Paths File must be in .csv format.
- Each row of the file must have the following parameters in the order shown:

Source MAC, Destination MAC, Source Outer VLAN ID, Source Inner VLAN ID, Destination Outer VLAN ID, Destination Inner VLAN ID, Source Router IP, Source Gateway IP, Source Netmask, Destination Router IP, Destination Gateway IP, Destination Netmask, Source IP, Destination IP, Source Port, Destination Port.

- Each parameter must be separated by a comma, if VLANs or virtual routers aren't enabled, leave a non-numeric value between commas (see Example without VLANs below)
- Lines that start with “#” are considered comments and are ignored.
- The maximum number of rows per file is 32,767.
- VLAN tagging must be uniform throughout the file. That is, they must all be, no VLAN, single VLAN, or double VLAN. Note that bad formats will be rejected at runtime.
- Single tagged VLANs must only be "Inner".
- There can only be 1 Virtual Router per VLAN (inner if single tagged, outer/inner based on Network Neighborhood “Interface"->”VLAN Key" if double tagged).

- Only one Virtual Router per VLAN (inner if single tagged, outer/inner based on Network Neighborhood "Interface"->"VLAN Key" if double tagged).
- Only one Virtual Router if not using VLANs.
- If VRs are specified, test ports will ARP the DUT for a gateway MAC address.
- ARP requests to test ports are only functional if a Virutal Router is configured.

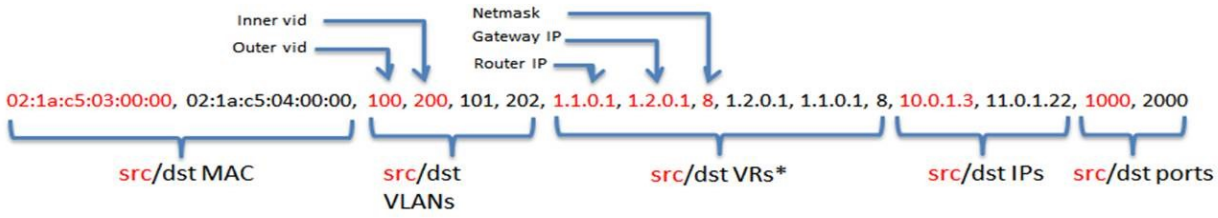
> **Note:** Use slow start to populate the DUT ARP cache.

> **Note:** A DUT cannot ARP test ports for static host MAC addresses.

- MACs must have a common base (top 16 bits). Only the bottom 32 bits can vary per tuple due to hardware limitations.

Paths File Format and Examples

Import Path File Format



*One VR per VLAN per direction (No VLANs counts as One VLAN)

Example with double tagged VLANs, no virtual routers:

02:1a:c5:03:00:00, 02:1a:c5:04:00:00, 100, 200, 101, 202, x, x, x, x, x, x, 10.0.1.3, 11.0.1.22, 1000, 2000

Example with single tagged VLANs, no virtual routers

02:1a:c5:03:00:00, 02:1a:c5:04:00:00, x, 200, x, 202, x, x, x, x, x, x, 10.0.1.3, 11.0.1.22, 1000, 2000

Example with no VLANs, no virtual routers:

02:1a:c5:03:00:00, 02:1a:c5:04:00:00, x, x, x, x, x, x, x, x, x, x, 10.0.1.3, 11.0.1.22, 1000, 2000

Example with virtual routers, no VLANs:

02:1A:C5:01:00:00, 08:5B:0E:E2:DF:7C, x, x, x, x,40.1.0.1,40.1.0.254,16,40.3.0.1,40.3.0.254,16, 1.36.1.0,3.36.1.165,55340,49673

Selecting a Paths File

To select a Paths File, click inside the **Paths File** field of an Advanced Path. You will then have two options for selecting a Paths File.

a. Select a Paths File from a list of previously uploaded files that will display. Then click OK.
b. Click Import to import your own file. Follow the onscreen prompts to upload your custom file. Select your file and then click **OK**.

Session Sender

The Session Sender test component measures a device's ability to set up and maintain a large number of TCP sessions over a period of time. Each session uses a unique combination of source addresses, destination addresses, source ports, and destination ports; therefore, there must be enough MAC/network address combinations allotted and enough source/destination port combinations to create that many sessions.

To ensure that there are enough addresses, verify that the Ethernet and Host Masks allow for enough addresses to be created. For example, the higher the Ethernet and Network Mask, the lower the number of available addresses there will be; therefore, you should set the Ethernet and Network Mask high enough so that there are enough addresses that the system can select from.

With the Session Sender test component, you can control:

- The maximum number of simultaneous TCP sessions
- The rate at which sessions are opened
- The duration of the sessions

Port Number Distribution

The Session Sender test component uses a unique pair of source and destination port numbers for each TCP session. If there are not enough source and destination port pairs, then the system may not reach the desired number of connections.

To prevent this from happening, you should set the Port Distribution Type parameter to Range or Random for both the source and destination ports. Then, you should define the Maximum Port Number and the Minimum Port Number parameters so that there are a large number of port numbers available for both the source and destination ports.

Phases in a Session Sender Test

There are three phases within a Session Sender test: ramp up, steady-state, and ramp down. Each phase dictates the behavior of the TCP sessions.

Ramp Up Phase

During the ramp up phase, the system will attempt to open the maximum number of simultaneous sessions in the time allotted to the ramp up phase. There are six parameters specifically used to create the ramp up phase:

- Ramp Up Behavior
- Ramp Up Duration
- Minimum Connection Rate
- Maximum Connection Rate

- Increment n Connections Per Second
- Fixed Time Interval

The **Ramp Up Behavior** parameter determines how the sessions are opened, and the **Ramp Up Duration** parameter determines how long the ramp up phase lasts.

For example, if **Ramp Up Behavior** is set to **Full Open**, **Ramp Up Duration** is set to **5seconds**, and **Minimum Connection Rate** is set to **50,000**, then Session Sender will attempt to open as many sessions as possible, at the rate of (up to) 50,000 sessions per second for 5 seconds. Since **Ramp Up Behavior** is set to **Full Open**, Session Sender will perform the full TCP handshake when a connection is made.

Once Session Sender finishes the ramp up phase, it will attempt to maintain the total number of sessions that it was able to open.

Note: The value defined for Maximum Simultaneous Super Flows is an upper-bound limit on the number of sessions that can be open at any given time during the test; therefore, the number of sessions that Session Sender maintains may be less than the value defined.

Ramp Up Rate

Session Sender will automatically adjust the ramp up rate so that it works within the test's duration. For example, if you have configured the ramp up duration to 5 seconds and the number of simultaneous sessions to 50 sessions, then the configured ramp up rate will be 10 sessions per second.

In instances where the ramp up rate is a decimal value, such as 10.5 sessions per second, Session Sender will round the ramp up rate down the nearest value. Session Sender will automatically round the rate down unless the rate is less than 1 session per second. In those cases, Session Sender will use the configured rate.

The only time in which the ramp up rate will be less than 1 is when the number of simultaneous sessions is less than the ramp up duration. For example, if you set the ramp up time to 20 seconds and the number of sessions to be opened to 10, then the ramp up rate is 0.5 sessions per second. In this case, instead of rounding the rate to 0 sessions per second, Session Sender will use 0.5 sessions per second as the ramp up rate.

Steady-State Phase

During the steady-state phase, the system will open, transmit data, and close sessions (depending on the steady-state behavior) while maintaining the maximum number of sessions. There are two parameters specifically associated with the steady-state phase: **Steady-State Behavior** and **Steady-State Duration**. The Steady-State Behavior parameter determines the behavior of the open sessions. The **Steady-State Duration** parameter determines how long the steady-state phase lasts.

For example, if **Steady-State Behavior** is set to **Open and Close Sessions**, **Steady-State Duration** is set to **30 seconds**, **Maximum Simultaneous Super Flows** is set to **100,000**, and **Maximum Super Flows Per Second** is set to **50,000**, then the Session Sender test component will maintain up to 100,000 sessions for 30 seconds, at a rate of (up to) 50,000 sessions per second. Since **Steady-State Behavior** is set to **Open and Close Sessions**, the sessions will be closed once they finish sending data, and new sessions will be opened in their place.

Note: The value defined for Maximum Simultaneous Super Flows is an upper-bound limit on the number of Super Flows that can be open at any given time during the test; therefore, the number of sessions that Session Sender maintains may be less than the value defined.

Ramp Down Phase

During the Ramp Down phase, no new sessions will be established. There are two parameters specifically associated with the ramp down phase: **Ramp Down Behavior** and **Ramp Down Duration**. As existing sessions complete during the Ramp Down phase, they will be closed in the manner specified by these settings. The Ramp Down Behavior parameter determines the behavior of the sessions when they are being closed. The Ramp Down Duration parameter determines how long the Ramp Down phase lasts.

For example, if **Ramp Down Behavior** is set to **Full Close** and **Ramp Down Duration** is set to **1 second**, then the Session Sender test component will perform a complete TCP session close on all open sessions.

Note: If your test displays a message that says Failures Due to Ramp Down, make sure that you have configured enough ramp down time for all of the connections to close on their own.

Session Sender Test Example

To tie together the different ramp phases, take a look at the parameters defined in the following table.

Session Sender Test Values

Parameter	Value
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	100,000
Session/Super Flow Configuration.Maximum Super Flows Per Second	50,000
Session Ramp Distribution.Ramp Up Behavior	Full Open
Session Ramp Distribution.Ramp Up Seconds	5 seconds
Session Ramp Distribution.Steady-State Behavior	Open and Close Sessions
Session Ramp Distribution.Steady-State Time	30 seconds
Session Ramp Distribution.Ramp Down Behavior	Full Close
Session Ramp Distribution.Ramp Down Time	1 second
Session/Super Flow Configuration.Target Minimum Simultaneous Sessions	100
Session/Super Flow Configuration.Target Minimum Sessions Per Second	50

Based on the parameters defined in the table above, the test will attempt to open 100,000 sessions with the full TCP handshake within the 5 second ramp up phase. Then, the test will try to maintain the maximum number of sessions that it was able to open for the steady-state period of 30 seconds. During the steady-state phase, the test will open new sessions, send data, and then close them. Finally, after the steady-state phase is over, the test will try to close all open sessions.

The Target Minimum Simultaneous Super Flows and the Target Minimum Super Flows Per Second parameters set the pass/fail criteria for the test. If the test meets these target values, then the device will pass the test – regardless of whether or not it is able to reach the values defined for maximum simultaneous Super Flows and maximum Super Flows per second.

Note: To determine how much time to allot to Ramp Up Seconds to open the maximum number of sessions, use the following equation:

Maximum Simultaneous Super Flows / Target Sessions Per Second = Ramp Up Duration

Single-Session High Throughput

Session Sender is the only test component that enables you to run a single stream at 1 Gbps. In order to create a single-session high throughput stream, you will need to set the Payload Packets Per Session parameter to -1. Normally, this parameter specifies the number of data segments that are sent during each session, but when it is set to -1, Session Sender will send an unlimited amount of data during the stream.

Once one stream finishes, Session Sender will send another; it will continue sending individual streams of traffic for the duration of the test.

In order to set up a single stream of high-throughput, you will need to modify the following parameters:

- Session/Super Flow Configuration.Maximum Simultaneous Super Flows = 1
- Session/Super Flow Configuration.Maximum Super Flows Per Second = 1
- Payload Packets Per Session = -1
- Data Rate.Minimum Data Rate = 1,000

Additionally, you can set the sizes of the segments being sent during each session (i.e., the segment size distribution type, minimum, and maximum parameters). The segment size distribution parameters are applied only to the first 1,000 packets in the stream. If you have the segment size distribution type set to either random or range, then the sizes are repeated again.

Note: Due to the dynamic nature of TCP, you may see the same data segment sent more than once in a session if the device under test drops a packet and the test must retry the send.

Additionally, if you do not want to manually configure the parameters for a high throughput session, you can use of the single session high throughput templates available by clicking the Load a Template button.

To set up a high-throughput single-stream session:

1. Create or open an existing Session Sender test.
2. Click the **Add New** button in the Test Components section.

3. Select a Session Sender component from the list and click the Select button.
4. Enter a name and description for the component. (Optional)
5. Select the **Use Template check box** to select a pre-configured test component. (Optional)
6. Click the **Create** button. The name of the component you selected will be displayed in the Test Components section.
7. Click the edit component icon.
8. Locate the **Payload Packets Per Session** field in the TCP Configuration section.
9. Enter -1 in the **Payload Packets Per Session** field.
10. Enter 1000 in the **Minimum Data Rate** field in the Data Rate section.
11. Verify that the **Data Rate Type** parameter is set to **Constant**.
12. Configure any other parameters or components as desired.
13. Click the **Return** to Test Workspace button.
14. Click the S**ave and Run** button to run the test.

Note: To include TCP latency statistics in the report, the Add Segment Timestamps check box in the TCP Configuration section must be selected.

Session Sender Parameters

The following table lists the parameters for the Session Sender test component.

Session Sender Parameters

Parameter	Description	Valid Values
Payload.Transport	Sets the protocol for Session Sender	TCP UDP ICMP UDP Lossy – A payload type of UDP Lossy indicates that UDP packets that are not received due to packet loss are not counted as errors. All – (Combines TCP, UDP, and ICMP)
Payload.Add Payload Timestamp	Inserts time stamps for UDP and ICMP packets.	true or false

Parameter	Description	Valid Values
Payload.Type	Sets how the component determines the payload it will use for its traffic.	**0** – Payload is 0s. **1** – Payload is all 1s. **Random** – Payload is defined using random Hex values. **HTTP** – Payload consists of a simple HTTP 1.0 GET request for the '/' URL, padded to match the payload size distribution. **User-Defined** – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
Payload.HTTP Request Type		HTTP 1.0 - GET - No Compression HTTP 1.0 - GET - Compression HTTP 1.1 - GET - No Compression HTTP 1.1 - GET - Compression HTTP 1.0 - POST - No Compression HTTP 1.0 - POST - Compression HTTP 1.1 - POST - No Compression HTTP 1.1 - POST - Compression
Payload.User Defined Data	Defines the payload; this parameter is defined only if **Payload.Type** is set to **User Defined**. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)

Parameter	Description	Valid Values
Payload Size Distribution.Distribution Type	Sets how the component will define the size of the TCP segment.	**Constant** – All payloads will use the size defined for **Payload Size Distribution.Minimum (bytes)**. **Range** – All payloads will use the size defined for **Payload Size Distribution.Minimum (bytes)** and will increment to the payload size defined for **Payload Size Distribution. Maximum (bytes)**. The system uses an algorithm that determines the incremental value that will increase **Payload Size Distribution.Minimum (bytes)** until it reaches **Payload Size Distribution.Maximum (bytes)**. **Random** – All payloads will have sizes that are randomly chosen between **Payload Size Distribution. Minimum (bytes)** and **Payload Size Distribution. Maximum (bytes)**.
Payload Size Distribution.Minimum (bytes)	Sets the minimum TCP payload size that will be sent by the client. This value is used as the payload size if **Segment Size Distribution.Distribution type** is defined as **Constant**.	0 – 9,416
Payload Size Distribution. Maximum (bytes)	Sets the maximum TCP payload size that will be SENT by the client. This parameter is defined only if **Payload Size Distribution.Distribution Type** is set to **Range** or **Random**.	0 – 9,416

Parameter	Description	Valid Values
Data Rate.Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable **Note:** Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data Rate.Data Rate Scope	Defines whether the rate distribution number is treated as a per-interface limit or an aggregate limit on the traffic that this component generates.	Limit Per-Interface Throughput Limit Aggregate Throughput
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use to send its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data rate type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) 1,000 (1Gb) Mbps

Parameter	Description	Valid Values
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 148,800,952 (10Gb) or 14,880,095 (1Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Session Ramp Distribution.SYN Only Retry Mode	Defines the behavior of the TCP Retry Mechanism when dealing with the initial SYN packet of a flow.	Continuous – Continue sending SYN packets, even if we have ran out of retries (Retry Count). Continuous with new session – Same as Continuous, except we change the initial sequence number every Retry Count loop(s). Obey Retry Count – Send no more than Retry Count initial SYN packets.
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	Sets the maximum number of concurrent sessions that can be set up by the system at a given time. This value must be greater than **Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Maximum Simultaneous Active Flows	Sets the maximum number of concurrent sessions that can be active in the system at the same time.	1 – 20,000,000 (10Gb) See Maximum Simultaneous Active Flows Details on page 811.
Session/Super Flow Configuration.Maximum Super Flows Per Second	Sets the maximum number of sessions that can occur per second. This value must be greater than **Session/Super Flow Configuration.Target Minimum Super Flows Per Second**.	1 – 1,000,000 (10Gb) or *500,000 (1Gb)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Engine Selection	Select the type of engine with which to run the test component. Select Advanced to enable the default, full-featured engine. Select Simple to enable a simpler, higher-performance, stateless engine.	Advanced (Features) Simple (Performance)
Session/Super Flow Configuration.Performance Emphasis	Adjusts whether the advanced engine's flow scheduler favors opening new sessions, sending on existing sessions, or a mixture of both. Note:Ramp-up phases automatically emphasize Sessions, ramp-down phases emphasize Throughput.	Balanced – Emphasize both opening new sessions and sending data n existing sessions equally (default) Simultaneous Sessions – Emphasize opening new sessions Throughput – Emphasize sending data on existing sessions
Session/Super Flow Configuration.Resource Allocation Override	Allows you to override the amount of CPU and other resources allocated to this test component. The value indicates the amount of resources to use on a single processor.	Automatic – The system will estimate the amount of resources required by a particular component. % – Manually tune the amount (in terms of percentages) of resources allocated to this component
Session/Super Flow Configuration.Statistic Detail	Adjusts the level of statistics to be collected. Decreasing the number of statistics collected can increase performance and allow for targeted reporting.	Maximum – Enable all possible statistics Application Only – Enable only Application statistics (L7) Transport Only – Enable only Transport statistics (L4/L3) Minimum – Disable most statistics

Parameter	Description	Valid Values
Session/Super Flow Configuration.Unlimited Super Flow Open Rate	Disables the limit on the open rate. If this option is set to **false**, then the open rate is limited by the connection establishment rate. If this option is set to **true**, then the system will open the Super Flows as quickly as the bandwidth allows; therefore, the number of closed Super Flows will be closer to the maximum number of simultaneous Super Flows set for the component.	true or false
Session/Super Flow Configuration.Unlimited Super Flow Close Rate	This parameter determines how fast sessions are closed. If set to false, session close rate will mirror the session open rate. If set to true, sessions will be closed as fast as possible.	true or false
Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows	The number of Super Flows that must open to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Target Minimum Super Flows Per Second	The number of sessions per second that must be reached to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Super Flows Per Second**.	1 – 1,000,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Target Number of Successful Matches	Specifies the minimum number of successful matches required to pass in the final results.	N/A
IPv4 Configuration.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255

Parameter	Description	Valid Values
IPv4 Configuration.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff
IPv6 Configuration.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv6 Configuration.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF
IPv6 Configuration.Flowlabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
Source Port.Port Distribution Type	Sets how the component will obtain the source port numbers.	**Constant** – Uses **Source Port.Minimum port number** as the source port. **Random** – Uses random values between **Source Port.Minimum port number** and **Source Port.Maximum port number**. **Range** – Increments **Source Port.Minimum port number** by one until it reaches **Source Port.Maximum port number**. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
Source Port.Minimum Port Number	Sets the minimum source port number, if **Source Port.Port distribution type** is **Range** or **Random**. Otherwise, this will be the value used for the source port.	0 – 65,535

Parameter	Description	Valid Values
Source Port.Maximum Port Number	Sets the maximum source port number, if **Source Port.Port distribution type** is **Range** or **Random**.	0 – 65,535
Destination Port.Port Distribution Type	Sets how the component will obtain destination ports for TCP connections.	**Constant** – Uses **Destination Port.Minimum port number** as the source port. **Random** – Uses random values between **Destination Port.Minimum port number** and **Destination Port.Maximum port number**. **Range** – Increments **Destination Port.Minimum port** by one until it reaches **Destination Port.Maximum port**. Once the port number reaches the maximum destination port number, it will reset to the minimum destination port number.
Destination Port.Minimum Port Number	Sets the minimum destination port number, if **Destination Port.Port distribution type** is **Range** or **Random**. Otherwise, this will be the value used for the destination port.	0 – 65,535
Destination Port.Maximum Port Number	Sets the maximum destination port number, if **Destination Port.Port distribution type** is **Range** or **Random**.	0 – 65,535
TCP Configuration.Maximum Segment Size (MSS)	Sets the maximum segment size (MSS) that is used during the ramp up phase. MSS is the maximum size that a client advertises it can receive.	512 – 9,146

Parameter	Description	Valid Values
TCP Configuration.Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing.	1 – 120
TCP Configuration.Reset at End	Indicates whether or not a test should reset all existing TCP connections at the end. If enabled, all TCP connections will reset if the test either ends naturally or is canceled.	true or false
TCP Configuration.Retry Quantum	Sets the amount of time that elapses before a connection is retried.	1 – 2,000
TCP Configuration.Retry Count	Sets the number of times a connection is attempted before it is canceled.	1 – 3
TCP Configuration.Delay ACKs	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
TCP Configuration.Initial Receive Window	Sets the size of the initial receive window for a new connection.	1 – 65,535
TCP Configuration.TCP Window Size	This value will allow the tcp connection to negotiate the window scale option as per RFC1323. The value set will be the maximum window size that the connection will attempt to negotiate. The window scale factor is the number of bits by which to shift the window size. Send and receive windows are negotiated separately based on the configuration of the equipment at each end.	0-14 (0=disabled)

Parameter	Description	Valid Values
TCP Configuration.Dynamic receive window size	Dynamic receive window size = 1 (box checked - this is the default setting) Note that earlier versions of BPS had this option enabled by default. • The receive window size is increased by MSS with each received window until it reaches the maximum value (65535 * 2^window_scale_factor). • If Delay ACK is enabled, the receive window size is doubled on window full. Dynamic received window size = 0 (box unchecked) • In order to keep the window fixed, the "Dynamic Receive Window Size" needs to be set to off. • The receive window size is not increased. • It remains fixed in all scenarios.	true or false
TCP Configuration.Add Segment Timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
TCP Configuration.Piggy-back Data on 3-way Handshake ACK	Determines whether to add Data to the client ACK packet of the TCP handshake.	true or false
TCP Configuration.Piggy-back Data on Shutdown FIN	Determines whether to add Data to the client FIN packet of the TCP shutdown	true or false

Parameter	Description	Valid Values
TCP Configuration.Initial Congestion Window	Determines the size of the initial congestion window.	1 – 16
TCP Configuration.Explicit Congestion Notification	Determines the support level for ECN.	Disable – Disables all support for ECN Support ECN – ECN will be supported if the remote host initiates it first. Use ECN – ECN will be initiated for new connections.
TCP Configuration.Raw Flags	Allows the specification of the TCP flags as bits.	-1 – 255
TCP Configuration.Connect Delay	Adds two delays (one between the TCP SYN-ACK packet and the final ACK, and one between the final ACK and the first data packet) in the TCP state machine when the delay is set to a value greater than 0.	0 – 10
TCP Configuration.Payload Packets Per Session	Specifies how many data packets are sent during an open session.	1 – 1,000
TCP Configuration.Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	0 – 4,294,967,295

Note: * 1 Gb blades will only support up to 5,000,000 simultaneous sessions at a rate of 500,000 sessions per second.

Security

The Security test component can be used to test network security devices – such as IPS, IDS, and firewalls. It measures a device's ability to protect a host by sending Strikes and verifying that the device successfully blocks the attacks.

To create a Security test, you will need to select a Strike List and an Evasion Setting. BreakingPoint offers several default options for Strike Lists and Evasion Settings. For more information on creating Strike Lists, see the Strike List Security Group on page 223 section. For more information on the configuration of the default Evasion Settings, see the Evasion Settings on page 725section.

Additionally, the Security component allows you to edit Evasion Profiles. This allows you to override any evasion options that are configured for an Evasion Setting or configured within the Strike List.

One-Arm Security

One-arm security testing allows you to test the authenticity of the attack traffic generated by the Security component. It targets a specific destination address (or range of addresses) through the test port of the chassis.

It is designed to trigger the vulnerabilities in your device, rather than exploit them for access; therefore, this mode will put your device in a crash condition, and it will not result in code execution on the device under test.

One-Arm Security Test

To set up a one-arm Security test, you will need to enable the External interface for the Security component. Therefore, this requires that you create a Network Neighborhood that has addressing information configured for the specific destination address, or range of addresses, that you are targeting.

To create a one-arm Security test:

1. Select **Test > New Test** from the Menu bar.
2. Click the browse available network neighborhoods icon.
3. Select a Network Neighborhood and click the **Select** button.
4. Click the edit network neighborhood icon.
5. Configure any of the parameters for each interface to be used in your test. For more information the external interface, see the External Interface Addressing on page 157 section.
6. Click the go back arrow to return to the Test Workspace page.
7. Click the **Add New** button to add a component to your test.
8. Select the **Security** component and click the **Select** button.
9. Enter a name and description for the component. (Optional)
10. Click the **Create** button.
11. Click the **Edit Component** button.
12. Assign Client and/or Server tags to the component.
13. Configure any of the parameters for the Security component as desired. For more information on Security parameters, see the Security Parameters on the facing page section.
    a. Adjust any parameters for the test component.
    b. Edit the Evasion Profile settings. (Optional)
    c. Edit the Concurrent Strikes settings from the Parameter Label section (Optional). The Concurrent Strikes parameter allows you to choose between Single Strike and Default modes. Single Strike mode runs only one strike at a time, while Default mode runs up to five strikes simultaneously.
14. Click the **Return to Test Workspace** button when done.

15. Click the **Save As** button to save the test without running it, or click **Save and Run** to save and run the test.

Security Test Results

The table below lists the definitions of the terms found in the Security Test results.

Security Test Results Terms

Term	Definition
Blocked Client	A packet that was sent from the client was blocked.
Blocked Close	A strike that is blocked during the FIN - FIN/ACK on close.
Blocked Open	A strike that is blocked in a 3-way handshake opening connection.
Blocked Server	A packet that was sent from the server was blocked.

Security Templates

BreakingPoint offers five default levels of security testing:

- Security Level 1 – Targets high-risk vulnerabilities in services often exposed to the Internet. This includes approximately 170 Strikes.
- Security Level 2 – Targets all high-risk vulnerabilities. This includes approximately 450 Strikes.
- Security Level 3 – Targets all high-risk vulnerabilities, worms, and backdoors. This includes approximately 500 Strikes.
- Security Level 4 – Targets all vulnerabilities, worms, and backdoors. This includes approximately 750 strikes.
- Security Level 5 – Targets all vulnerabilities, worms, backdoors, probes, and denial of service flaws. This includes approximately 2,800 non-fuzzing Strikes.

Note: If you want to run all Strikes that are available on the system, then you can use the All Strikes Strike List. Running this Strike List can take up to 33 hours.

Security Parameters

The following table lists the parameters for the Security test component.

Security Parameters

Parameter	Description	Valid Values
Concurrent Strikes	Sets the maximum number of Strikes that will run simultaneously.	Single Strike – Runs one Strike at a time. Default – Runs up to five Strikes concurrently.
Maximum Attacks Per Second	Sets the maximum number of attacks sent every second.	0 – 1,000
Maximum Packets Per Second	Sets the maximum number of packets sent per second.	0 – 100,000
Attack Timeout Seconds	Sets the amount of time the system will wait for a packet to arrive at its destination before resending the attack or determining that the DUT successfully blocked the attack.	0 – 3,600
Attack Retries	Sets the number of times to attempt an attack before determining that the DUT successfully blocked the attack.	0 – 100
Random Seed	Determines whether the test will generate static or dynamic attacks. '0' will randomize the content of each strike in the strike series. Any other value defined here will keep the strike content static.	0 – 4,294,967,295
Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	0 – 4,294,967,295
Strike List	Sets the Strike List the Security component will use to derive its attacks.	A Strike List
Strike List Iterations	The number of times the Security component will send the configured set of attacks.	1 – 1000
Strike List Interation Delay	If Strike List Iterations is set to be 2 or more, this setting determines the number of seconds the Security Component will sleep before starting the next Strike List Iteration.	0 – 4,294,967,295
Evasion Profile	Sets the default evasion options for the Strikes.	An Evasion Profile

Malware

The Malware test component can be used to test network security devices – such as IPS, IDS, and firewalls. It measures a device's ability to protect a host by sending Strikes and verifying that the device successfully blocks the attacks.

To create a Malware test, you will need to select a Strike List and an Evasion Setting. BreakingPoint offers several default options for Strike Lists and Evasion Settings. For more information on creating Strike Lists, see the Strike List Security Group on page 223 section. For more information on the configuration of the default Evasion Settings, see the Evasion Settings on page 725 section.

Please be aware that the Malware test component exclusively runs malware strikes. Any non-malware strikes must be removed before your test will start. In order to use the Malware test component, your system must have the malware packages installed. Malware packages are available at the Ixia Support Website with your ATI Update service.

Additionally, the Malware component allows you to edit Evasion Profiles. This allows you to override any evasion options that are configured for an Evasion Setting or configured within the Strike List.

One-Arm Security

One-arm security testing allows you to test the authenticity of the attack traffic generated by the Malware component. It targets a specific destination address (or range of addresses) through the test port of the chassis.

It is designed to trigger the vulnerabilities in your device, rather than exploit them for access; therefore, this mode will put your device in a crash condition, and it will not result in code execution on the device under test.

One-Arm Security Test

To set up a one-arm Malware test, you will need to enable the External interface for the Malware component. Therefore, this requires that you create a Network Neighborhood that has addressing information configured for the specific destination address, or range of addresses, that you are targeting.

Please be aware that the Malware test component exclusively runs malware strikes. Any non-malware strikes must be removed before your test will start. In order to use the Malware test component, your system must have the malware packages installed.

Note: Malware packages are available at the Ixia Support website with your ATI Update service.

To create a one-arm Security test:

1. Select **Test > New Test** from the Menu bar.
2. Click the browse available network neighborhoods icon.
3. Select a Network Neighborhood and click the Select button.
4. Click the edit network neighborhood icon.

5. Configure any of the parameters for each interface to be used in your test. For more information the external interface, see the External Interface Addressing on page 157 section.
6. Click the go back arrow to return to the Test Workspace page.
7. Click the **Add New** button to add a component to your test.
8. Select the **Malware** component and click the **Select** button.
9. Enter a name and description for the component. (Optional)
10. Click the **Create** button.
11. Click the **Edit Component** button.
12. Assign Client and/or Server tags to the component.
13. Configure any of the parameters for the Malware component as desired. For more information on Malware parameters, see the section Malware Parameters on the facing page.
    a. Adjust any parameters for the test component.
    b. Edit the Evasion Profile settings. (Optional)
14. Edit the Concurrent Strikes settings from the Parameter Label section (Optional). The Concurrent Strikes parameter allows you to choose between Single Strike and Default modes. Single Strike mode runs only one strike at a time, while Default mode runs up to five strikes simultaneously.
15. Click the **Return to Test Workspace** button when done.
16. Click the **Save As** button to save the test without running it, or click **Save and Run** to save and run the test.

Malware Test Results

The following table lists the definitions of the terms found in the Malware Test results.

Malware Test Results Terms

Term	Definition
Blocked Client	A packet that was sent from the client was blocked.
Blocked Close	A strike that is blocked during the FIN - FIN/ACK on close.
Blocked Open	A strike that is blocked in a 3-way handshake opening connection.
Blocked Server	A packet that was sent from the server was blocked.

Malware Template

BreakingPoint offers five default levels of security testing:

- Security Level 1 – Targets high-risk vulnerabilities in services often exposed to the Internet. This includes approximately 170 Strikes.
- Security Level 2 – Targets all high-risk vulnerabilities. This includes approximately 450 Strikes.
- Security Level 3 – Targets all high-risk vulnerabilities, worms, and backdoors. This includes approximately 500 Strikes.

- Security Level 4 – Targets all vulnerabilities, worms, and backdoors. This includes approximately 750 strikes.
- Security Level 5 – Targets all vulnerabilities, worms, backdoors, probes, and denial of service flaws. This includes approximately 2,800 non-fuzzing Strikes.

Note: If you want to run all Strikes that are available on the system, then you can use the All Strikes Strike List. Running this Strike List can take up to 33 hours.

Malware Parameters

The following table lists the parameters for the Malware test component.

Parameter	Description	Valid Values
Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable **Note:** Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data Rate Scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	Limit Aggregate Throughput or Limit Per-Interface Throughput
Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/Second or Megabits/Second
Data Rate Type	Sets how the component determines the data rate it will use to send its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.

Parameter	Description	Valid Values
Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if Data Rate Type is set to Constant. Otherwise, this is the minimum value used by the test if Data Rate Type is set to Range or Random.	Frames per second: 1 – 1,488,095 (1Gb ports) 1 – 14,880,952 (10Gb ports)
		Megabits per second: 1– 1000 (1Gb ports) 1 – 10,000 (10Gb ports)
Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if Data Rate Type is set to Range or Random.	Frames per second: 1 – 1,488,095 (1Gb ports) 1 – 14,880,952 (10Gb ports)
		Megabits per second: 1– 1000 (1Gb ports) 1 – 10,000 (10Gb ports)
Maximum Simultaneous Attacks	Defines the maximum simultaneous strikes that will exist concurrently during the test duration.	1 – 20,000,000
Maximum Attacks Per Second	Sets the maximum number of attacks sent every second.	0 – 1,000,000
Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
Attack Retries	Sets the number of times to attempt an attack before determining that the DUT successfully blocked the attack.	0 – 100

Parameter	Description	Valid Values
Random Seed	Determines whether the test will generate static or dynamic attacks. '0' will randomize the content of each strike in the strike series. Any other value defined here will keep the strike content static.	0 – 4,294,967,296
Strike List	Sets the Malware strikes to use in the test.	A Strike List
Strike List Iterations	The number of times to run the Strike List.	1 – 1000
Strike List Interation Delay	The number of seconds between Strike List Iterations.	0 – 4,294,967,295
Evasion Profile	Sets the default evasion options for the Strikes.	An Evasion Profile

Evasion Settings

The following table lists the Evasion Settings and their default configurations.

Evasion Settings

Evasion Setting	Configuration
Default evasion settings	No evasion options are applied.
IP: Ordered 16 byte, overlapping (new)	Splits every IP packet into 16 byte fragments; each fragment is overlapped, and new data is given priority. Includes the following evasion option(s) and value(s): • IP:MaxFragSize – 16 • IP:FragPolicy – Last • IP:FragEvasion – Overlap-all-new
IP: Ordered 16 byte, overlapping (old)	Splits every IP packet into 16 byte fragments; each fragment is overlapped, and old data is given priority. Includes the following evasion option(s) and value(s): • IP:MaxFragSize – 16 • IP:FragPolicy – first • IP:FragEvasion – Overlap-all-old

Evasion Setting	Configuration
IP: Ordered 24 byte fragments	Splits every IP packet into 24 byte fragments and sends the fragments in order. Includes the following evasion option(s) and value(s): IP:MaxFragSize – 24
IP: Ordered 8 byte fragments	Splits every IP packet into 8 byte fragments and sends the fragments in order. Includes the following evasion option(s) and value(s): IP:MaxFragSize – 8
IP: Out-of-order 8 byte fragments	Splits every IP packet into 8 byte fragments and sends the fragments in a random order. Includes the following evasion option(s) and value(s): • IP:MaxFragSize – 8 • IP:FragOrder – random
IP: Reverse order 8 byte fragments	Splits every IP packet into 8 byte fragments and sends the fragments in reverse order. Includes the following evasion option(s) and value(s): • IP:MaxFragSize – 8 • IP:FragOrder – reverse
TCP: Ordered 1 byte segments	Configures the Strikes to segment RPC Strikes into 2-byte TCP segments. Includes the following evasion option(s) and value (s): • DCERPC:MaxFragmentSize – 2 • SUNRPC:TCPFragmentSize – 2
TCP: Ordered 1 byte segments, duplicate last packet	Splits every TCP packet into 1 byte segments and resends the last packet. Includes the following evasion option(s) and value (s): • TCP:MaxSegmentSize – 1 • TCP:DuplicateLastSegment – true
TCP: Ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums	Splits every TCP packet into 1 byte segments with invalid TCP checksums. Includes the following evasion option(s) and value (s): • TCP:MaxSegmentSize – 1 • TCP:DuplicateBadChecksum – true

Evasion Setting	Configuration
TCP: Ordered 1 byte segments, interleaved duplicate segments with null TCP control flags	Splits every TCP packet into 1 byte segments and sends duplicate segments with null TCP control flags. Includes the following evasion option(s) and value(s): • TCP:MaxSegmentSize – 1 • TCP:DuplicateNullFlags – true
TCP: Ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence numbers mid-stream	Splits every TCP packet into 1 byte segments. Includes the following evasion option(s) and value(s): • TCP:MaxSegmentSize – 1 • TCP:DuplicateBadSyn – true
TCP: Ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers	Splits every TCP packet into 1 byte segments, and sends 1 packet with an out-of-window sequence number for each real packet. Includes the following evasion option(s) and value(s): • TCP:MaxSegmentSize – 1 • TCP:DuplicateBadSeq – true
TCP: Out of order 1 byte segments	Splits every TCP packet into 1 byte segments, and sends them in a random order. Includes the following evasion option(s) and value(s): • TCP:MaxSegmentSize – 1 • TCP:SegmentOrder – random
Browser: High Evasion	Performs evasion attacks against Web browsers using Unicode UTF-7 character encoding, Gzip encoding, and chunked encoding with very small chunk sizes. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_7 • HTML:HTMLUnicodeUTF7EncodingMode – standard • HTTP:ServerChunkedTransfer – true • HTTP:ServerChunkedTransferSize – 3 • HTTP:ServerCompression – gzip
Browser: Low Evasion	Performs evasion attacks against Web browsers using Unicode UTF-7 character encoding. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_7 • HTML:HTMLUnicodeUTFEncodingMode – standard

Evasion Setting	Configuration
Browser: Medium Evasion	Performs evasion attacks against Web browsers using Unicode UTF-7 character encoding and chunked encoding. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_7 • HTML:HTMLUnicodeUTF7EncodingMode – standard • HTTP:ServerChunkedTransfer – true • HTTP:ServerChunkedTransferSize – 32
DCERPC: High Evasion	Configures Strikes to perform high levels of evasion. Includes the following evasion option(s) and value(s): • DCERPC:MaxFragmentSize – 2 • DCERPC:MultiContextBind – true • DCERPC:MultiContextHead – 20 • DCERPC:MultiContextTail – 20 • SMB:MaxWriteSize – 4 • SMB:MaxReadSize – 4 • SMBLRandomPipeOffset – true
DCERPC: Low Evasion	Configures Strikes to perform low levels of evasion. Includes the following evasion option(s) and value(s): • DCERPC:MaxFragmentSize – 256 • DCERPC:MultiContextBind – true • SMB:MaxWriteSize – 512 • SMB:MaxReadSize – 512
DCERPC: Medium Evasion	Configures Strikes to perform medium levels of evasion. Includes the following evasion option(s) and value(s): • DCERPC:MaxFragmentSize – 128 • DCERPC:MultiContextBind – true • SMB:MaxWriteSize – 128 • SMB:MaxReadSize – 128 • SMBLRandomPipeOffset – true
FTP: Multiple telnet opcodes, beginning of command	Inserts multiple telnet opcodes at the beginning of each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 3

Evasion Setting	Configuration
FTP: Multiple telnet opcodes, randomly placed	Inserts multiple telnet opcodes at a random location in each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 4
FTP: One telnet opcode per character	Inserts a single telnet opcode between each character in each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 6
FTP: One telnet opcode per word	Inserts a single telnet opcode between each word in each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 6
FTP: Single telnet opcode, beginning of command	Inserts a single telnet opcode at the beginning of each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 1
FTP: SIngle telnet opcode, randomly placed	Inserts a single telnet opcode at a random location in each client FTP command. Includes the following evasion option(s) and value(s): • FTP:PadCommandWhitespace – true • FTP:FTPEvasionLevel – 2
HTML: Unicode UTF16 (Big Endian)	Encodes HTML content using Unicode UTF-16 big-endian character encoding. Includes the following evasion option(s) and value(s): HTML:HTMLUnicodeEncoding – UTF_16BE
HTML: Unicode UTF16 (Little Endian)	Encodes HTML content using Unicode UTF-16 little-endian character encoding. Includes the following evasion option(s) and value(s): HTML:HTMLUnicodeEncoding – UTF_16LE

Evasion Setting	Configuration
HTML: Unicode UTF32 (Big Endian)	Encodes HTML content using Unicode UTF-32 big-endian character encoding. Includes the following evasion option(s) and value(s): HTML:HTMLUnicodeEncoding – UTF_32BE
HTML: Unicode UTF32 (Little Endian)	Encodes HTML content using Unicode UTF-32 little-endian character encoding. Includes the following evasion option(s) and value(s): HTML:HTMLUnicodeEncoding – UTF_32LE
HTML: Unicode UTF7 All	Encodes HTML content using Unicode UTF-7 character encoding. All characters, including alphanumeric characters, are encoded. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_7 • HTML:HTMLUnicodeUTF7EncodingMode – all
HTML: Unicode UTF7 Standard	Encodes HTML content using Unicode UTF-7 character encoding. Alphanumeric characters are not encoded. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_7 • HTML:HTMLUnicodeUTF7EncodingMode – standard
HTML: Unicode UTF8 Overlong Invalid Maximum Size	Encodes HTML content using Unicode UTF-8 invalid character encoding, with a UTF-8 encoding size of 7. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_8 HTML: • HTMLUnicodeUTF7EncodingMode – invalid • HTML:HTMLUnicodeUTF8EncodingSize – 7
HTML: Unicode UTF8 Overlong Invalid Minimum Size	Encodes HTML content using Unicode UTF-8 overlong invalid character encoding, with a UTF-8 encoding size of 2. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_8 • HTML:HTMLUnicodeUTF8EncodingMode – invalid • HTML:HTMLUnicodeUTF8EncodingSize – 2

Evasion Setting	Configuration
HTML: Unicode UTF8 Overlong Maximum Size	Encodes HTML content using Unicode UTF-8 overlong character encoding, with a UTF-8 encoding size of 7. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_8 • HTML:HTMLUnicodeUTF7EncodingMode – overlong • HTML:HTMLUnicodeUTF8EncodingSize – 7
HTML: Unicode UTF8 Overlong Minimum Size	Encodes HTML content using Unicode UTF-8 character encoding, with a UTF-8 encoding size of 2. Includes the following evasion option(s) and value(s): • HTML:HTMLUnicodeEncoding – UTF_8 • HTML:HTMLUnicodeUTF7EncodingMode – overlong • HTML:HTMLUnicodeUTF8EncodingSize – 2
HTTP: Apache High Evasion	Configures the Strikes to run as if the target Web server is running Apache, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:EncodeHexAll – true • HTTP:EndRequestFakeHTTPHeader – true • HTTP:DirectoryFakeRelative – true • HTTP:DirectorySelfReference – true • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true • HTTP:MethodURITabs – true • HTTP:VersionRandomizeCase – true • HTTP:MethodRandomizeCase – true • HTTP:MethodRandomInvalid – true • HTTP:URIPrependAltSpaces – true • HTTP:URIPrependAltSpacesSize – 1 • HTTP:URIAppendAltSpaces – 1 • HTTP:URIAppendAltSpacesSize – 1
HTTP: Apache Low Evasion	Configures the Strikes to run as if the target Web server is running Apache, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:EncodeHexRandom – true • HTTP:DirectorySelfReferece – true • HTTP:VersionRandomizeCase – true

Evasion Setting	Configuration
HTTP: Apache Medium Evasion	Configures the Strikes to run as if the target Web server is running Apache, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:EncodeHexRandom – true • HTTP:DirectoryFakeRelative – true • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true • HTTP:MethodURITabs – true • HTTP:VersionRandomizeCase – true
HTTP: Apache No Evasion	Configures the Strikes to run as if the target Web server is running Apache. No evasion options are applied.
HTTP: Complete hex encoding	Encodes all characters with Hex encoding. Includes the following evasion option(s) and value(s): HTTP:EncodeHexAll – true
HTTP: Complete unicode encoding	Encodes all characters with Unicode encoding. Includes the following evasion option(s) and value(s): • HTTP:HTTP:ServerProfile – iis • HTTP:EncodeUnicodeAll – true
HTTP: Covert forward slash to backslash	Converts all forward slashes in the URL to back slashes. Includes the following evasion option(s) and value(s): • HTTP:HTTPServerProfile – iis • HTTP:ForwardToBackSlashes – true
HTTP: Fake relative directory	Embeds an encoded HTTP header in the URL. Includes the following evasion option(s) and value(s): HTTP:EndRequestFakeHTTPHeader – true
HTTP: GET / POST Parameter Random Prepend	Generates random variables in GET and POST requests. Includes the following evasion option(s) and value(s): • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true

Evasion Setting	Configuration
HTTP: IIS High Evasion	Configures the Strikes to run as if the target Web server is running IIS, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:HTTPServerProfile – iis • HTTP:EncodeUnicodeAll – true • HTTP:EndRequestFakeHTTPHeader – true • HTTP:DirectoryFakeRelative – true • HTTP:DirectorySelfReference – true • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true • HTTP:MethodURITabs – true • HTTP:URIRandomizeCase – true • HTTP:ForwardToBackSlashes – true
HTTP: IIS Low Evasion	Configures the Strikes to run as if the target Web server is running IIS, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:HTTPServerProfile – iis • HTTP:EncodeUnicodeRandom – true • HTTP:DirectorySelfReference – true • HTTP:MethodURITabs – true • HTTP:URIRandomizeCase – true
HTTP: IIS Medium Evasion 1	Configures the Strikes to run as if the target Web server is running IIS, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:HTTPServerProfile – iis • HTTP:EncodeUnicodeRandom – true • HTTP:DirectoryFakeRelative – true • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true • HTTP:MethodURITabs – true • HTTP:URIRandomizeCase – true • HTTP:ForwardToBackSlashes – true

Evasion Setting	Configuration
HTTP: IIS Medium Evasion 2	Configures the Strikes to run as if the target Web server is running IIS, with several HTTP-specific evasion options set. Includes the following evasion option(s) and value(s): • HTTP:HTTPServerProfile – iis • HTTP:EndRequestFakeHTTPHeader – true • HTTP:EncodeUnicodeRandom – true • HTTP:HTTP:DirectoryFakeRelative – true • HTTP:DirectorySelfReference – true • HTTP:GetParameterRandomPrepend – true • HTTP:PostParameterRandomPrepend – true • HTTP:MethodURITabs – true • HTTP:URIRandomizeCase – true • HTTP:ForwardToBackSlashes – true
HTTP: No Evasion	Configures the Strikes to run as if the target Web server is running IIS. No evasion options are applied.
HTTP: Random hex encoding	Encodes random characters with hex encoding. Includes the following evasion option(s) and value(s): HTTP:EncodeHexRandom – true
HTTP: Request fake HTTP header	Embeds an encoded HTTP header in the URL. Includes the following evasion option(s) and value(s): HTTP:EndRequestFakeHTTPHeader – true
HTTP: Self-referential directory	Embeds pathnames that reference the current directory. Includes the following evasion option(s) and value(s): HTTP:DirectorySelfReference – true
HTTP: Self-referential directory and Fake relative	Embeds fake pathnames and uses parent paths to go backup the tree and pathnames that reference the current directory. Includes the following evasion option(s) and value(s): • HTTP:DirectorySelfReference – true • HTTP:DirectoryFakeRelative – true

Evasion Setting	Configuration
RPC: 1-byte TCP segments	Configures the Strikes to segment RPC Strikes into 1-byte TCP segments. Includes the following evasion option(s) and value (s): • DCERPC:MaxFragmentSize – 1 • SUNRPC:TCPFragmentSize – 1
RPC: 2-byte TCP segments	Configures the Strikes to segment RPC Strikes into 2-byte TCP segments. Includes the following evasion option(s) and value (s): • DCERPC:MaxFragmentSize – 2 • SUNRPC:TCPFragmentSize – 2

Stack Scrambler

The Stack Scrambler test component tests the integrity of different protocol stacks by sending malformed IP, TCP, UDP, ICMP, and Ethernet packets (produced by a fuzzing technique) to the device under test. The fuzzing technique modifies a part of the packet (checksum, protocol options, etc.) to generate the corrupt data.

Stack Scrambler Parameters

The following table lists the parameters for the Stack Scrambler test component.

Stack Scrambler Parameters

Parameter	Description	Valid Values
IPv4 Configuration.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv4 Configuration.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff
IPv6 Configuration.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv6 Configuration.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF

Parameter	Description	Valid Values
IPv6 Configuration.Flowlabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
TCP Configuration.Maximum Segment Size (MSS)	Sets the maximum segment size (MSS) that is used during the ramp up phase. MSS is the maximum size that a client advertises it can receive.	512 – 9,146
TCP Configuration.Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing.	1 – 120
TCP Configuration.Reset at End	Indicates whether or not a test should reset all existing TCP connections at the end. If enabled, all TCP connections will reset if the test either ends naturally or is canceled.	true or false
TCP Configuration.Retry Quantum	Sets the amount of time that elapses before a connection is retried.	1 – 2,000
TCP Configuration.Retry Count	Sets the number of times a connection is attempted before it is canceled.	1 – 3
TCP Configuration.Delay ACKs	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
TCP Configuration.Initial Receive Window	Sets the size of the initial receive window for a new connection.	1 – 65,535

Parameter	Description	Valid Values
TCP Configuration.TCP Window Size	This value will allow the tcp connection to negotiate the window scale option as per RFC1323. The value set will be the maximum window size that the connection will attempt to negotiate. The window scale factor is the number of bits by which to shift the window size. Send and receive windows are negotiated separately based on the configuration of the equipment at each end.	0-14 (0=disabled)
TCP Configuration.Dynamic receive window size	Dynamic receive window size = 1 (box checked - this is the default setting) Note that earlier versions of BPS had this option enabled by default. • The receive window size is increased by MSS with each received window until it reaches the maximum value (65535 * 2^window_scale_ factor). • If Delay ACK is enabled, the receive window size is doubled on window full. Dynamic received window size = 0 (box unchecked) • In order to keep the window fixed, the "Dynamic Receive Window Size" needs to be set to off. • The receive window size is not increased. • It remains fixed in all scenarios.	true or false

Parameter	Description	Valid Values
TCP Configuration.Add Segment Timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
TCP Configuration.Piggy-back Data on 3-way Handshake ACK	Determines whether to add Data to the client ACK packet of the TCP handshake.	true or false
TCP Configuration.Piggy-back Data on Shutdown FIN	Determines whether to add Data to the client FIN packet of the TCP shutdown	true or false
TCP Configuration.Initial Congestion Window	Determines the size of the initial congestion window.	1 – 16
TCP Configuration.Explicit Congestion Notification	Determines the support level for ECN.	Disable – Disables all support for ECN Support ECN – ECN will be supported if the remote host initiates it first. Use ECN – ECN will be initiated for new connections.
TCP Configuration.Raw Flags	Allows the specification of the TCP flags as bits.	-1 – 255
TCP Configuration.Connect Delay	Adds two delays (one between the TCP SYN-ACK packet and the final ACK, and one between the final ACK and the first data packet) in the TCP state machine when the delay is set to a value greater than 0.	0 – 10

Parameter	Description	Valid Values
Source Port.Port Distribution Type	Sets how the component will obtain the source port numbers.	Constant – Uses Source Port.Minimum port number as the source port.
		Random – Uses random values between Source Port.Minimum port number and Source Port.Maximum port number.
		Range – Increments Source Port.Minimum port number by one until it reaches Source Port.Maximum port number. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
Source Port.Minimum Port Number	Sets the minimum source port number, if Source Port.Port distribution type is Range or Random. Otherwise, this will be the value used for the source port.	0 – 65,535
Source Port.Maximum Port Number	Sets the maximum source port number, if Source Port.Port distribution type is Range or Random.	0 – 65,535

Parameter	Description	Valid Values
Destination Port.Port Distribution Type	Sets how the component will obtain destination ports for TCP connections.	Constant – Uses Destination Port.Minimum port number as the source port.
		Random – Uses random values between Destination Port.Minimum port number and Destination Port.Maximum port number.
		Range – Increments Destination Port.Minimum port by one until it reaches Destination Port.Maximum port. Once the port number reaches the maximum destination port number, it will reset to the minimum destination port number.
Destination Port.Minimum Port Number	Sets the minimum destination port number, if Destination Port.Port distribution type is Range or Random. Otherwise, this will be the value used for the destination port.	0 – 65,535
Destination Port.Maximum Port Number	Sets the maximum destination port number, if Destination Port.Port distribution type is Range or Random.	0 – 65,535
Payload.Transport	Sets the protocol stack to target.	TCP
		UDP
		ICMP
		UDP Lossy – A payload type of UDP LOSSY indicates that UDP packets that are not received due to packet loss are not counted as errors.
		ALL – (Combines TCP, UDP, and ICMP)

Parameter	Description	Valid Values
Payload.Type	Sets how the component determines the payload it will use for its traffic.	0 – Payload is 0s.
		1 – Payload is all 1s.
		Random – Payload is defined using random Hex values.
		User-Defined – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
Payload.User Defined Data	Defines the payload; this parameter is defined only if Payload.Type is set to User-Defined. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)

Parameter	Description	Valid Values
Payload Size Distribution.Distribution Type	Sets how the component will define the size of the TCP segment.	**Constant** – All payloads will use the size defined for **Payload Size Distribution.Minimum (bytes)**. **Range** – All payloads will use the size defined for **Payload Size Distribution.Minimum (bytes)** and will increment to the payload size defined for **Payload Size Distribution. Maximum (bytes)**. The system uses an algorithm that determines the incremental value that will increase **Payload Size Distribution.Minimum (bytes)** until it reaches **Payload Size Distribution.Maximum (bytes)**. **Random** – All payloads will have sizes that are randomly chosen between **Payload Size Distribution. Minimum (bytes)** and **Payload Size Distribution. Maximum (bytes)**.
Payload Size Distribution.Minimum (bytes)	Sets the minimum TCP payload size that will be sent by the client. This value is used as the payload size if **Segment Size Distribution.Distribution type** is defined as **Constant**.	0 – 9,416
Payload Size Distribution. Maximum (bytes)	Sets the maximum TCP payload size that will be SENT by the client. This parameter is defined only if **Payload Size Distribution.Distribution Type** is set to **Range** or **Random**.	0 – 9,416

Parameter	Description	Valid Values
Data Rate.Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable Note:Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data Rate.Data Rate Scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	**Limit Per-Interface Throughput** – Uses the data rate as the limit for the transmitting and receiving interfaces. **Limit Aggregate Throughput** – Uses the data rate as an aggregate limit for the test component.
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use to send its traffic.	**Constant** – Uses Data Rate.Minimum value as the date rate. **Random** – Selects a random value between the Data Rate.Minimum value and Data Rate.Maximum value as the data rate. **Range** – Starts at the Data Rate.Minimum value and increments until it reaches the Data Rate.Maximum value. The system uses an algorithm that determines the incremental value that will increase **Data Rate.Minimum value** until it reaches **Data Rate.Maximum value**.

Parameter	Description	Valid Values
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data rate type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data rate type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Session Ramp Distribution.SYN Only Retry Mode	Defines the behavior of the TCP Retry Mechanism when dealing with the initial SYN packet of a flow.	Continuous – Continue sending SYN packets, even if we have ran out of retries (Retry Count). Continuous with new session – Same as Continuous, except we change the initial sequence number every Retry Count loop(s). Obey Retry Count – Send no more than Retry Count initial SYN packets.
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	Sets the maximum number of concurrent sessions that can be set up by the system at a given time. This value must be greater than **Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Maximum Super Flows Per Second	Sets the maximum number of sessions that can occur per second. This value must be greater than **Session/Super Flow Configuration.Target Minimum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Maximum Simultaneous Active Flows	Sets the maximum number of concurrent sessions that can be active in the system at the same time.	1 – 20,000,000 See Maximum Simultaneous Active Flows Details on page 811.
Session/Super Flow Configuration.Engine Selection	Select the type of engine with which to run the test component. Select Advanced to enable the default, full-featured engine. Select Simple to enable a simpler, higher-performance, stateless engine.	Advanced (Features) Simple (Performance)
Session/Super Flow Configuration.Performance Emphasis	Adjusts whether the advanced engine's flow scheduler favors opening new sessions, sending on existing sessions, or a mixture of both. Note:Ramp-up phases automatically emphasize Sessions, ramp-down phases emphasize Throughput.	Balanced – Emphasize both opening new sessions and sending data n existing sessions equally (default) Simultaneous Sessions – Emphasize opening new sessions Throughput – Emphasize sending data on existing sessions
Session/Super Flow Configuration.Resource Allocation Override	Allows you to override the amount of CPU and other resources allocated to this test component. The value indicates the amount of resources to use on a single processor.	Automatic – The system will estimate the amount of resources required by a particular component. % – Manually tune the amount (in terms of percentages) of resources allocated to this component

Parameter	Description	Valid Values
Session/Super Flow Configuration.Statistic Detail	Adjusts the level of statistics to be collected. Decreasing the number of statistics collected can increase performance and allow for targeted reporting.	Maximum – Enable all possible statistics Application Only – Enable only Application statistics (L7) Transport Only – Enable only Transport statistics (L4/L3) Minimum – Disable most statistics
Session/Super Flow Configuration.Unlimited Super Flow Open Rate	Disables the limit on the open rate. If this option is set to **false**, then the open rate is limited by the connection establishment rate. If this option is set to **true**, then the system will open the Super Flows as quickly as the bandwidth allows; therefore, the number of closed Super Flows will be closer to the maximum number of simultaneous Super Flows set for the component.	Enable or Disable
Session/Super Flow Configuration.Unlimited Super Flow Close Rate	This parameter determines how fast sessions are closed. If set to false, session close rate will mirror the session open rate. If set to true, sessions will be closed as fast as possible.	true or false
Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows	The number of Super Flows that must open to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Target Minimum Super Flows Per Second	The number of sessions per second that must be reached to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Target Number of Successful Matches	Specifies the minimum number of successful matches required to pass in the final results.	N/A
Pseudo-random Number Generator Options.Seed for the generator	Sets a value for the seed generator. This value enables the ability to resend the same data to the device. Setting the random seed to '0' will generate dynamic content.	0 – 4,294,967,295
Pseudo-random Number Generator Options.Offset into the Seed	Deprecated.	
StackScrambler.Maximum number of simultaneous corruptions	Sets the maximum number of corruptions per packet.	1 – 5
StackScrambler.Bad Ethernet Type	Sets the maximum possible percentage of Ethernet packets that will have malformed Ethernet Types. The actual percentage could be lower.	0 – 100
StackScrambler.Bad IP Version	Sets the maximum possible percentage of IP packets that will have a malformed IP version. The actual percentage could be lower.	0 – 100
Bad IPv4 TTL or Bad IPv6 Hop Limit	The maximum percentage of packets that will have a randomized TTL set. This value is 8 bits in length and specifies the number of hops to traverse before being discarded. In the IPv4 header, this value is known as TTL. In the IPv6 Header, this value is known as Hop Limit.	0 – 100

Parameter	Description	Valid Values
Bad IPv4 Header Length	Sets the maximum possible percentage of IP packets that will have a malformed IP Length set. The actual percentage could be lower. This value is 4 bits in length and specifies the number of 32-bit words in the header.	0 – 100
Bad IP Differentiated Services Field (TOS)	Sets the maximum possible percentage of IP packets that will have malformed IP Differentiated Services Field. The actual percentage could be lower.	0 – 100
Bad IPv4 or IPv6 Total Length	Sets the percentage of IP packets that will have malformed IP total length	0 – 100
Bad IPv4 Flags	Sets the maximum possible percentage of IPv4 packets that will have malformed IP flags. The actual percentage could be lower.	0 – 100
Bad IPv4 Fragment Offset	Sets the maximum possible percentage of IPv4 packets that will have a malformed Fragment Offset. The actual percentage could be lower.	0 – 100
Bad IP Protocol	Sets the maximum possible percentage of packets that will have malformed IP protocol set in the IP header. The actual percentage could be lower. This value is 8 bits in length and is labeled Protocol in the IPv4 header and Next Header in the IPv6 header.	0 – 100

Parameter	Description	Valid Values
Bad IPv4 Checksum	Sets the maximum percentage of packets that will have a randomized IPv4 checksum set in the IP header. This value is 16 bits in length and is used for error-checking of the header.	0 – 100
Bad IPv4 Options	Sets the maximum possible percentage of TCP packets that will have malformed IP options set in the IP header. The actual percentage could be lower.	0 – 100
Bad IPv6 Flow Label	The maximum percentage of packets transmitted that will have a randomized Flow label. This value is 20 bits in length and was created for giving real-time applications special service.	0 – 100
Bad TCP Flags	Sets the maximum possible percentage of packets that will have a malformed TCP Flags. The actual percentage could be lower.	0 – 100
Bad TCP Options	Sets the maximum possible percentage of packets transmitted that will have randomized TCP options set in the TCP header. The actual percentage could be lower. TCP Options are variable in length.	0 – 100
Bad TCP or UDP Header Length	Sets the maximum possible percentage of packets that will have a malformed TCP Data Offset. The actual percentage could be lower.	0 – 100
Bad TCP Urgent Pointer	Sets the maximum possible percentage of packets that will have a malformed TCP Urgent Pointer. The actual percentage could be lower.	0 – 100

Parameter	Description	Valid Values
Bad L4 Checksum	Sets the maximum possible percentage of packets that will have a malformed TCP, UDP, or ICMP checksum. The actual percentage could be lower.	0 – 100
Bad ICMP Type	Sets the maximum possible percentage of packets that will have a malformed ICMP Type (ICMP or All Payload Transport must be chosen for this value to take effect). The actual percentage could be lower.	0 – 100
Bad ICMP Code	Sets the maximum possible percentage of packets that will have a malformed ICMP Code (ICMP or All Payload Transport must be chosen for this value to take effect). The actual percentage could be lower.	0 – 100
Bad GTP Flags	The maximum percentage of packets transmitted that will have randomized GTP-U Flags. This value is 8 bits in length.	0 – 100
Bad GTP Type	The maximum percentage of packets transmitted that will have randomized a GTP-U Type. This value is 8 bits in length.	0 – 100
Bad GTP Length	The maximum percentage of packets transmitted that will have randomized a GTP-U Length. This value is 16 bits in length.	0 – 100
Bad GTP Sequence Number	The maximum percentage of packets transmitted that will have randomized a GTP-U Sequence Number. This value is 16 bits in length.	0 – 100

Parameter	Description	Valid Values
Bad GTP N-PDU	The maximum percentage of packets transmitted that will have randomized a GTP-U N-PDU. This value is 8 bits in length.	0 – 100
Bad GTP Next	The maximum percentage of packets transmitted that will have randomized a GTP-U Next. This value is 8 bits in length.	0 – 100
Establish TCP Sessions	Whether to send valid handshake packets to establish TCP sessions before starting fuzzing.	0 – 100
Establish TCP Sessions	Determines whether the system sends valid handshake packets to establish TCP sessions before fuzzing.	true or false
Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
* When specifying durations in frames for Bit Blaster and Routing Robot, the minimum number of frames requested will be honored. At times, however, a small number of frames above the requested value may be sent. In most cases, the number of frames sent will be rounded up to a multiple of four.		

Application Simulator

The Application Simulator test component allows you to generate application traffic flows. This test component should be used in conjunction with other test components to simulate real world traffic.

The Application Simulator test component uses an App Profile to determine what types of application flows to send to the DUT. The App Profile contains a set of flow specifications that defines the protocol, client-type, and server-type the traffic will use. For more information on App Profiles, see the Each Super Flow will be assigned a weight that determines its frequency in the application traffic and a seed that determines whether the Super Flow generates static or dynamic application flows. Super Flows with higher weights will make up larger portion of the test traffic. For more information on Super Flow weight distribution, see the section Super Flow Weight Distribution below. on page 268 section.

The following video link provides a tutorial on how to Build and Run an Application Simulator test: https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

Application Simulator Test Phases

There are three phases within an Application Simulator test: ramp up, steady-state, and ramp down. Each phase dictates the behavior of the TCP flows.

Ramp Up Phase

During the ramp up phase, the system will attempt to open as many TCP flows as possible in the time allotted to the ramp up phase; however, no data will be sent during the ramp up phase. Any traffic that is sent during this period that is non-TCP (i.e., UDP, ARP, ICMP) will not be affected by the ramp up phase and will send data as usual.

The Application Simulator test component will use the value defined for **Session/Super Flow Configuration.Maximum Simultaneous Super Flows** as an upper-bound limit on the number of flows that can be open during the ramp up phase. The duration of the ramp up phase is determined by the value defined for **Ramp Up Duration**.

Steady-State Phase

During the steady-state phase, the system will attempt to maintain the number of TCP flows opened during the ramp up phase. It will open flows, send data, and then close the flows for the duration of the steady-state phase. The duration of the steady-state phase is determined by the value defined for **Steady-State Distribution**.

Ramp Down Phase

During the ramp down phase, no new sessions will be opened, but the Application Simulator test component will finish running all open flows. The duration of the ramp down phase is determined by the value defined for **Ramp Down Duration**.

Application Simulator Parameters

The following table lists the parameters for the Application Simulator test component.

Application Simulator Parameters

Parameter	Description	Valid Values
Include in Report	Select the Include in Report checkbox to include detailed statistics from the test in the report. Deselect the checkbox to disable detailed statistics from the test in the report.	Enable or disable

Parameter	Description	Valid Values
Data Rate.Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable **Note:** Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data Rate.Data Rate Scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	Limit Per-Interface Throughput: Uses the data rate as the limit for the transmitting and receiving interfaces. Limit Aggregate Throughput: Uses the data rate as an aggregate limit for the test component.
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second

Parameter	Description	Valid Values
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use for its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data Rate Type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps

Parameter	Description	Valid Values
Session Ramp Distribution.SYN Only Retry Mode	Defines the behavior of the TCP Retry Mechanism when dealing with the initial SYN packet of a flow.	Continuous – Continue sending SYN packets, even if we have ran out of retries (Retry Count). Continuous with new session – Same as Continuous, except we change the initial sequence number every Retry Count loop(s). Obey Retry Count – Send no more than Retry Count initial SYN packets.
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	Sets the maximum number of concurrent sessions that can be set up by the system at a given time. This value must be greater than **Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Maximum Simultaneous Active Flows	Sets the maximum number of concurrent sessions that can be active in the system at the same time.	1 – 20,000,000 See Maximum Simultaneous Active Flows Details on page 811.
Session/Super Flow Configuration.Maximum Super Flows Per Second	Sets the maximum number of sessions that can occur per second. This value must be greater than **Session/Super Flow Configuration.Target Minimum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Engine Selection	Select the type of engine with which to run the test component. Select Advanced to enable the default, full-featured engine. Select Simple to enable a simpler, higher-performance, stateless engine.	Advanced (Features) Simple (Performance)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Performance Emphasis	Adjusts whether the advanced engine's flow scheduler favors opening new sessions, sending on existing sessions, or a mixture of both. **Note:** Ramp-up phases automatically emphasize Sessions, ramp-down phases emphasize Throughput.	Balanced – Emphasize both opening new sessions and sending data n existing sessions equally (default) Simultaneous Sessions – Emphasize opening new sessions Throughput – Emphasize sending data on existing sessions
Session/Super Flow Configuration.Resource Allocation Override	Allows you to override the amount of CPU and other resources allocated to this test component. The value indicates the amount of resources to use on a single processor.	Automatic – The system will estimate the amount of resources required by a particular component. % – Manually tune the amount (in terms of percentages) of resources allocated to this component
Session/Super Flow Configuration.Statistic Detail	Adjusts the level of statistics to be collected. Decreasing the number of statistics collected can increase performance and allow for targeted reporting.	Maximum – Enable all possible statistics Application Only – Enable only Application statistics (L7) Transport Only – Enable only Transport statistics (L4/L3) Minimum – Disable most statistics

Parameter	Description	Valid Values
Session/Super Flow Configuration.Unlimited Super Flow Open Rate	Disables the limit on the open rate. If this option is set to **false**, then the open rate is limited by the connection establishment rate. If this option is set to **true**, then the system will open the Super Flows as quickly as the bandwidth allows; therefore, the number of closed Super Flows will be closer to the maximum number of simultaneous Super Flows set for the component.	true or false
Session/Super Flow Configuration.Unlimited Super Flow Close Rate	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
Session/Super Flow Configuration.Target Minimum Simultaneous Sessions	The number of sessions that must open to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Target Minimum Super Flows Per Second	The number of sessions per second that must be reached to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Target Number of Successful Matches	Specifies the minimum number of successful matches required to pass in the final results.	N/A
IPv4 Configuration.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv4 Configuration.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff

Parameter	Description	Valid Values
IPv6 Configuration.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv6 Configuration.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF
IPv6 Configuration.Flowlabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
Source Port.Port Distribution Type	Sets how the component will obtain the source port numbers.	Constant – Uses Source Port.Minimum port number as the source port.
		Random – Uses random values between Source Port.Minimum port number and Source Port.Maximum port number.
		Range – Increments Source Port.Minimum port number by one until it reaches Source Port.Maximum port number. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
Source Port.Minimum Port Number	Sets the minimum source port number, if Source Port.Port distribution type is Range or Random. Otherwise, this will be the value used for the source port.	0 – 65,535
Source Port.Maximum Port Number	Sets the maximum source port number, if Source Port.Port distribution type is Range or Random.	0 – 65,535

Parameter	Description	Valid Values
TCP Configuration.Maximum Segment Size (MSS)	Sets the maximum segment size (MSS) that is used during the ramp up phase. MSS is the maximum size that a client advertises it can receive.	512 – 9,146
TCP Configuration.Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing.	1 – 120
TCP Configuration.Reset at End	Indicates whether or not a test should reset all existing TCP connections at the end. If enabled, all TCP connections will reset if the test either ends naturally or is canceled.	true or false
TCP Configuration.Retry Quantum	Sets the amount of time that elapses before a connection is retried.	1 – 2,000
TCP Configuration.Retry Count	Sets the number of times a connection is attempted before it is canceled.	1 – 3
TCP Configuration.Delay ACKs	When enabled, the first unacknowledged data packet received arms the Delayed ACK timer. The delayed ack timer value can be either static (Delay ACKs ms parameter) or dynamically calculated based on the following algorithm: 40 + retry_quantum_ms/10 * minimum(rcv_windows, 65535)/65535 **Note:** • DelayACK with SSL is only supported on Two Arm tests on PerfectStorm and CloudStorm. • A Piggyback ACK on data is sent whenever possible. • Regarding the data rate limiter - With user defined data rates, if BPS is not able to Piggyback the ACK on a data packet, an explicit ACK packet is sent. This does not affect scenarios where the data rate is unlimited.	true or false

Parameter	Description	Valid Values
TCP Configuration.Delay ACKs ms	Sets the amount of time that elapses before an explicit ACK is sent. This value should be smaller than the value of TCP Configuration.Retry Quantum in order to avoid unwanted retries.	0-500
TCP Configuration.Initial Receive Window	Sets the size of the initial receive window for a new connection.	1 – 65,535
TCP Configuration.TCP Window Size	This value will allow the tcp connection to negotiate the window scale option as per RFC1323. The value set will be the maximum window size that the connection will attempt to negotiate. The window scale factor is the number of bits by which to shift the window size. Send and receive windows are negotiated separately based on the configuration of the equipment at each end.	0-14 (0=disabled)
TCP Configuration.Dynamic receive window size	Dynamic receive window size = 1 (box checked - this is the default setting) Note that earlier versions of BPS had this option enabled by default. • The receive window size is increased by MSS with each received window until it reaches the maximum value (65535 * 2^window_scale_factor). • If Delay ACK is enabled, the receive window size is doubled on window full. Dynamic received window size = 0 (box unchecked) • In order to keep the window fixed, the "Dynamic Receive Window Size" needs to be set to off. • The receive window size is not increased. • It remains fixed in all scenarios.	true or false
TCP Configuration.Add Segment Timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false

Parameter	Description	Valid Values
TCP Configuration.Piggy-back Data on 3-way Handshake ACK	Determines whether to add Data to the client ACK packet of the TCP handshake.	true or false
TCP Configuration.Piggy-back Data on Shutdown FIN	Determines whether to add Data to the client FIN packet of the TCP shutdown.	true or false
TCP Configuration.Initial Congestion Window	Determines the size of the initial congestion window.	1 – 16
TCP Configuration.Explicit Congestion Notification	Determines the support level for ECN.	Disable – Disables all support for ECN Support ECN – ECN will be supported if the remote host initiates it first. Use ECN – ECN will be initiated for new connections.
TCP Configuration.Raw Flags	Allows the specification of the TCP flags as bits.	-1 – 255
TCP Configuration.Connect Delay	Adds two delays (one between the TCP SYN-ACK packet and the final ACK, and one between the final ACK and the first data packet) in the TCP state machine when the delay is set to a value greater than 0.	0 – 10
App Configuration.Remove all DNS actions	The Remove all DNS Actions option removes all DNS specific operations inside the super flows (that is, if the super flow contains DNS protocol, no DNS packets will be sent on the wire if this option is enabled).	

Parameter	Description	Valid Values
App Configuration.Streams Per Super Flow	The maximum number of streams that will be instantiated for an individual Super Flow at one time. The effective number may be limited by the number of Super Flows in the test. Setting this parameter to a lower number will cause tests to initialize faster and provide less random application traffic. Setting the parameter to a higher number will allow tests to initialize slower with greater randomization in application traffic, especially for static flows.	1 – 256
App Configuration.Content Fidelity	Indicates whether applications will generate more complex, dynamic traffic, or generate simpler, possibly more performant traffic. This setting may not have an effect for all application protocols and profiles.	High – for more complex/dynamic traffic Normal – for simpler traffic
App Configuration.Replace Streams at Runtime	Enabled by default, this feature ensures that any Super Flow that is represented as a stream that has been active for at least 5 to 15 seconds, and no longer has active references could be removed, rebuilt (via AppManager and StreamServer), and then filled again and made active. Weighting will be honored in the case of an AppProfile. When this setting is disabled, BP will never replace the stream(s) associated with any SuperFlow, and weights are not honored.	true or false
App Configuration.Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
App Configuration.Application Profile	Determines the mix of applications simulated, as well as the specifics of what the traffic looks like for those applications.	Available Application Profile

Note: * 1 Gb blades will only support up to 5,000,000 simultaneous sessions at a rate of 500,000 sessions per second.

Notes on Session Performance

Total Session Dependencies

The results for a Session Sender test may not reflect the total number of TCP sessions you would expect to see based on the values you have defined for Maximum Sessions Per Second and test

duration. For example, if you have Maximum Sessions Per Second set to 5,000 and a total test duration of 30 seconds, you would expect to see a total of 150,000 sessions. However, there are several factors that impact the total number of sessions generated by the system.

The following factors will affect the total number of TCP sessions that the system generates:

- The amount of bandwidth each session consumes
- Whether the component use aggregate or per-interface bandwidth
- The packet sizes generated by the component
- The number of segments in a session (TCP Session Duration)
- The number many sessions were retried
- The amount of time allotted to Retry Quantum
- Whether delayed ACKs have been enabled
- The send/receive window sizes
- The maximum segment size (MSS)
- Ramp up/ramp down settings

Concurrent Session Statistics

Be aware that the TCP Concurrent statistics displayed in real-time for Session Sender reports the maximum value per interval while the TCP Concurrent statistics displayed in Reporting reports sampling interval edge values. For example, the real-time statistics for Session Sender may display 1000 concurrent TCP sessions while a Report for the same time period displays 850 concurrent TCP sessions.

SSL/TLS Testing

Note: TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE

Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are protocols that provide secure communication between a client and a server. Secure Sockets Layer is the name used for versions 3.0 and earlier, while Transport Layer Security is the name used for versions 3.1 and later. Both SSL and TLS provide confidentiality, message integrity, and endpoint authentication.

Typically, only the server endpoint is authenticated, but the protocol also provides mechanisms for client authentication.

BreakingPoint SSL/TLS performance highlights:

- Supports 380,000 one-arm sessions, or 190,000 client/server sessions.
- Supports up to 380,000 endpoints.
- Supports up to 43,000 client handshakes per second (with resume enabled).
- Supports bulk encryption rates up to 1.25 Gbps.

BreakingPoint SSL/TLS features highlights:

- Supports cipher suites.
- Supports one-arm clients, one-arm servers, or two-arm clients and servers.

- Any TCP flow can be tunneled through SSL/TLS simply by adding a few SSL/TLS Super Flow actions to the flow.
- Support for Close Notify.
- Support for resumed sessions on both client and server.
- Support for client authentication.
- Support up to 4096-bit keys.

SSL Templates

BreakingPoint contains five templates for initiating SSL testing:

- SSL HTTPS 1.0
  - Example: SSL/HTTPS 1.0 session with one GET/RESPONSE transaction.
- SSL HTTPS 1.1
  - Example: SSL/HTTPS 1.1 session with multiple GET/RESPONSE transactions.
- SSL HTTPS Bulk Encryption Performance
  - Example: SSL/HTTPS session with a small request and a 100KB response.
- SSL HTTPS One-arm Client Handshake Performance
  - Example: one-arm client performance test for measuring SSL/HTTPS handshakes per second.
- SSL HTTPS One-arm Client Request Performance
  - Example: one-arm client performance test for measuring SSL requests per second.

Supported Cipher Suites

The following table lists the supported Cipher Suites.

RSA_WITH_RC4_128_MD5	SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA	TLS_RSA_WITH_AES_128_GCM_SHA256	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
RSA_WITH_RC4_128_SHA	SSL_DH_anon_WITH_RC4_128_MD5	TLS_RSA_WITH_AES_256_GCM_SHA384	TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
RSA_WITH_3DES_EDE_CBC_SHA	SSL_DH_anon_WITH_3DES_EDE_CBC_SHA	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256	TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
RSA_WITH_AES_128_CBC_SHA	TLS_DHE_RSA_WITH_AES_128_CBC_SHA	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
RSA_WITH_AES_256_CBC_SHA	TLS_DHE_RSA_WITH_AES_256_CBC_SHA	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

SSL_RSA_ WITH_ NULL_MD5	TLS_DH_anon_ WITH_AES_256_ CBC_SHA	TLS_ECDH_RSA_ WITH_AES_128_ GCM_SHA256	TLS_ECDHE_RSA_ WITH_AES_128_ GCM_SHA256	This field is intentionally blank
SSL_RSA_ WITH_ NULL_SHA	TLS_RSA_WITH_ AES_128_CBC_ SHA256	TLS_ECDH_RSA_ WITH_AES_256_ GCM_SHA384	TLS_ECDHE_RSA_ WITH_AES_256_ GCM_SHA384	This field is intentionally blank
TLS_RSA_ WITH_IDEA_ CBC_SHA	TLS_RSA_WITH_ AES_256_CBC_ SHA256	TLS_ECDH_ECDSA_ WITH_AES_128_ CBC_SHA256	TLS_ECDHE_ ECDSA_WITH_AES_ 128_CBC_SHA256	This field is intentionally blank

Creating an SSL/TLS Test

You can create an SSL/TLS test with the Application Simulator or Client Simulator test components. The following section will provide an overview for creating an SSL test. You can use the following instructions as a model for creating an SSL test, and then use the import option to bring in your own certificates and keys.

To create a client/server SSL/TLS test:

1. Select **Test** > **New Test** from the Menu bar.
2. Click the browse available network neighborhoods icon and make a selection for your test. You can also select the default Network Neighborhood.
3. Click the Add New button in the Test Component section.
4. Select the Application Simulator component and click the Select button.
5. Enter a name for the component and click the Create button.
6. Click the edit component icon.
7. Click the **Load a Template** button in the Parameters section.
8. Select one of the following SSL templates:
   a. SSL HTTPS 1.0
   b. SSL HTTPS 1.1
   c. SSL HTTPS Bulk Encryption Performance
9. Click **Load** and **Continue**.
10. Configure any of the other parameters for the component as desired.
11. Click the **Return to Test Workspace** button.
12. Click **Save As**.

At this point, you have created a template for a client/server SSL test where BreakingPoint will act as both the client and server. You can run this test through a loopback cable. It can be customized as needed to fit your testing needs.

To create a one-arm SSL/TLS test:

1. Select **Test > New Test** from the Menu bar.
2. Click the **Browse** available network neighborhoods icon and make a selection for your test. You can also select the default Network Neigbhorhood.
3. Click **Add New** in the Test Component section.
4. Select the Application Simulator component and click the **Select** button.
5. Enter a name for the component and click **Create**.
6. Click the edit component icon.
7. Click the **Load a Template** button in the **Parameters** section.
8. Select one of the following SSL templates:
   a. SSL HTTPS One-arm Client Handshake Performance
   b. SSL HTTPS One-arm Client Request Performance
9. Click **Load** and **Continue**.
10. Configure any of the other parameters for the component as desired.
11. Click the **Return to Test Workspace** button.
12. Click **Save As.**

At this point, you have created a template for a one-arm SSL test where BreakingPoint will act only as the client. You will need an external server to run this test.

Customizing an SSL/TLS Test

To customize your test, edit the SSL-related actions in the Super Flow. The SSL-related actions are Accept TLS, Start TLS, and TLS Close Notify.

Note: For a client/server test, the Accept TLS action must appear in the Super Flow before the Start TLS action.

Converting an Encrypted Flow Into an Unencrypted Flow

This example describes how to convert from HTTPS port 443 to HTTP port 80. You can use the same steps to convert other protocols from an encrypted flow into an unencrypted flow.

To convert an encrypted flow into an unencrypted flow:

1. Select **Managers > Application Manager** from the menu bar.
2. Select the **Super Flows** tab.
3. Locate and select the Super Flow that you want to edit.
4. Select the appropriate flow in the Flows section.
5. Click the **Edit** the selected flow protocol parameters button.
6. Select the **Server Port** check box if it is not already checked.
7. Change the value in the Server Port field from 443 to 80.
8. Click **Apply Changes**.

Edit the Accept TLS Action

To edit the accept TLS Action:

1. Select the flow in the Flows section.
2. Click the **Add Action** button.
3. Select **Accept TLS** from the Action dropdown menu in the Create a New Action area.
4. Select the **Accept TLS** action in the Actions section.
5. Click the **Edit** the selected action parameters button.
    a. Make sure that the Enabled check box is selected.
    b. Make sure that the Enabled field is set to false.
6. Click **Apply Changes**.

Edit the Start TLS Action

To edit the start TLS Action:

1. Select **Start TLS** from the Action dropdown menu in the Create a New Action area.
2. Click the **Add Action** button.
3. Select the Start TLS action in the Actions section.
4. Click the **Edit** the selected action parameters button.
    a. Make sure that the Enabled check box is selected.
    b. Make sure that the Enabled field is set to false.
5. Click **Apply Changes**.
6. Click the S**ave Super Flow** button.

Converting an Unencrypted Flow Into an Encrypted Flow

This example describes how to convert from HTTP port 80 to HTTPS port 443. You can use the same steps to convert other protocols from an unencrypted flow to an encrypted flow.

To convert an unencrypted flow into an encrypted flow:

1. Select **Managers > Application Manager** from the menu bar.
2. Select the Super Flows tab.
3. Locate and select the Super Flow that you want to edit.
4. Select the appropriate flow in the Flows section.
5. Click the **Edit** the selected flow protocol parameters button.
6. Select the Server Port check box if it is not already checked.
7. Change the value in the Server Port field from 80 to 443.
8. Click **Apply Changes**.

Add or Edit the Accept TLS Action

If an Accept TLS action already exists on the flow you have selected, use the following steps to edit the action.

1. Select the flow in the Flows section.
2. Click the **Add Action** button.
3. Select Accept TLS from the Action dropdown menu in the Create a New Action area.
4. Select the Accept TLS action in the Actions section.
5. Click the **Edit** the selected action parameters button and make the desired edits to the action.
    a. Make sure that the Enabled check box is selected.
    b. Make sure that the Enabled field is set to true.
6. Click **Apply Changes**.

Note: If no Accept TLS action exists in the flow, add the action and adjust the parameters as needed.

Add or Edit the Start TLS Action

If an Start TLS action already exists on the flow you have selected, use the following steps to edit the action.

a. Select **Start TLS** from the Action dropdown menu in the Create a New Action area.
b. Click the **Add Action** button.
c. Select the Start TLS action in the Actions section.
d. Click the **Edit** the selected action parameters button.
    a. Make sure that the Enabled check box is selected.
    b. Make sure that the Enabled field is set to false.
e. Click **Apply Changes**.

Note: If no Start TLS action exists in the flow, add the action and adjust the parameters as needed.

f. Click the Save Super Flow button.

Creating a One-Arm SSL/TLS Test With Conditional Request Actions

This example demonstrates how to create a one-arm (client only) SSL/TLS test with Conditional Request actions. While this example uses the HTTP and HTTPS protocols, you can use the same steps to create one-arm SSL/TLS tests with Conditional Request actions using other protocols.

To create a one-arm SSL/TLS test with conditional request actions:

1. Create an HTTP (unencrypted) version of your test.
2. Verify your test with external HTTP servers.
3. Convert the flows from HTTP to HTTPS. Follow the instructions in the section titled to complete this step.

SSL/TLS Parameters

The following table lists the parameters for Accept TLS action.

Accept TLS Action Parameters

Parameter	Description	Valid Values
Enabled	If disabled, then skip the handshake and do not tunnel data through SSL. Note, if you disable this, you may need to change the TCP port number.	true or false
Min Version	Sets the minimum protocol version that will be negotiated during the handshake. If the client does not support this version, the session will be terminated.	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
Max Version	Sets the maximum protocol version that will be negotiated during the handshake. If the client does not request this version or lower, the session will be terminated.	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
Cipher	Sets the ciphersuite to use for the encrypted session.	See Supported Cipher Suites.
Resume Max Reuse	Sets an approximate upper limit on the number of times the server will allow a client to resume an SSL session.	
Resume Expire	Sets an approximate upper limit on the length of time (in seconds) an SSL session will be cached and available for resumption.	
Handshake Timeout	Sets the maximum time period (in milliseconds) between the time the TCP connection is established and the time the SSL handshake is completed. If the handshake does not complete within the specified period, the session is terminated.	

Parameter	Description	Valid Values
Client Authentication Enabled	Enable or Disable SSL/TLS client authentication. For client authentication to succeed, the following parameters must be set in the servers's Accept TLS action: Client Authentication (set to Enabled) Client Common Name (example: clientA_512.client.int) Client CA Certificate (example: BreakingPoint_cacert_client.crt) Client Cert Verify Mode (controls strictness of client authentication) In addition, the following parameters must be set in the client's Start TLS action: Client Authentication (set to Enabled) Client Certificate (example: BreakingPoint_clientA_512.crt) Client Private Key (example: BreakingPoint_clientA_512.key)	true or false
Client Common Name	The client's common name (CN) as it appears in the client's certificate.	
Client Cert Verify Mode	Controls how the server handles client certificates. It does not control or enforce the receipt of certificates. It only controls how verification is performed if a certificate is received.	Do Not Check Cert Allow Untrusted Cert Require Trusted Cert
Decryption Mode	Controls how encrypted application data received by the server side of the SSL connection is handled.	Decrypt – Decrypt all incoming application data. Discard/Count – Discard and update bulk decryption statistics. Discard/Nocount – Discard without updating bulk decryption statistics. Auto – Use Decrypt mode if there is action in this flow that processes the application data (e.g., a Conditional Request action). Otherwise use Discard / Count mode.

Parameter	Description	Valid Values
Server Certificate	A file in PEM format containing the server's certificate. This is required since server authentication is always performed during SSL handshakes.	BreakingPoint_clientA_1024.crt BreakingPoint_clientA_512.crt BreakingPoint_clientB_1024.crt BreakingPoint_clientB_512.crt BreakingPoint_serverA_1024.crt BreakingPoint_serverA_512.crt BreakingPoint_serverB_1024.crt BreakingPoint_serverB_512.crt
Server Private Key	A file in PEM format containing the server's private key.	BreakingPoint_clientA_1024.key BreakingPoint_clientA_512.key BreakingPoint_clientB_1024.key BreakingPoint_clientB_512.key BreakingPoint_serverA_1024.key BreakingPoint_serverA_512.key BreakingPoint_serverB_1024.key BreakingPoint_serverB_512.key
Client CA Certificate	A file in PEM format containing the certificate of the Certificate Authority that was used to sign the client's certificate. This is only used when client authentication is performed as part of the handshake.	BreakingPoint_cacert_client.crt BreakingPoint_cacert_server.crt

Start TLS Action Parameters

Parameter	Description	Valid Values
Enabled	If disabled, then skip the handshake and do not tunnel data through SSL. Note, if you disable this, you may need to change the TCP port number.	true or false
Min Version	Sets the minimum protocol version that will be negotiated during the handshake. If the server does not support this version, the session will be terminated.	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.

Parameter	Description	Valid Values
Max Version	Sets the maximum protocol version that will be negotiated during the handshake. If the server does not support this version or lower, the session will be terminated.	SSLv3 or TLS TLSv1 is supported on all BPS platforms. TLSv1.1 and TLSv1.2 are only supported on PS, PSOne and BPS-VE.
Cipher	Sets the ciphersuite to use for the encrypted session. Ciphersuites are not currently supported.	See Supported Cipher Suites.
Resume Max Reuse	Sets an approximate upper limit on the number of times a client will try to resume an SSL session.	
Resume Expire	Sets an approximate upper limit on the length of time (in seconds) a client will cache the information needed to resume a particular SSL session.	
Handshake Timeout	Sets the maximum time period (in milliseconds) between the time the TCP connection is established and the time the SSL handshake is completed. If the handshake is not completed within the specified period, the session is terminated.	
Client Authentication Enabled	Enable or Disable SSL/TLS client authentication. For client authentication to succeed, the following parameters must be set in the servers's Accept TLS action: Client Authentication (set to Enabled) Client Common Name (example: clientA_512.client.int) Client CA Certificate (example: BreakingPoint_cacert_client.crt) Client Cert Verify Mode (controls strictness of client authentication) In addition, the following parameters must be set in the client's Start TLS action: Client Authentication (set to Enabled) Client Certificate (example: BreakingPoint_clientA_512.crt) Client Private Key (example: BreakingPoint_clientA_512.key)	true or false

Parameter	Description	Valid Values
Server Common Name	The server's common name (CN) as it appears in the server's certificate. The server's certificate is provided to the client during the handshake.	
Server Name Indication	Can be used to indicate the hostname of the server to which the client is attempting to connect, in the ClientHello packet, during the SSL handshake process.	
Server Cert Verify Mode	Controls how the client handles server certificates. Server authentication, a required part of every SSL handshake, depends on the server certificate presented to the client during the SSL/TLS handshake.	Do Not Check Allow Untrusted Require Trusted
Decryption Mode	Controls how encrypted application data received by the server side of the SSL connection is handled.	Decrypt – Decrypt all incoming application data. Discard/Count – Discard and update bulk decryption statistics. Discard/Nocount – Discard without updating bulk decryption statistics. Auto – Use Decrypt mode if there is action in this flow that processes the application data (e.g., a Conditional Request action). Otherwise use Discard / Count mode.
Client Certificate	A file in PEM format containing the client's certificate. This is only used when client authentication is performed as part of the handshake.	BreakingPoint_clientA_1024.crt BreakingPoint_clientA_512.crt BreakingPoint_clientB_1024.crt BreakingPoint_clientB_512.crt BreakingPoint_serverA_1024.crt BreakingPoint_serverA_512.crt BreakingPoint_serverB_1024.crt BreakingPoint_serverB_512.crt

Parameter	Description	Valid Values
Client Private Key	A file in PEM format containing the client's private key.	BreakingPoint_clientA_1024.key BreakingPoint_clientA_512.key BreakingPoint_clientB_1024.key BreakingPoint_clientB_512.key BreakingPoint_serverA_1024.key BreakingPoint_serverA_512.key BreakingPoint_serverB_1024.key BreakingPoint_serverB_512.key
Server CA Cert	A file in PEM format containing the certificate of the CA Certificate Authority (a.k.a., the CA Cert) used to sign the server's certificate.	BreakingPoint_cacert_client.crt BreakingPoint_cacert_server.crt

The table below lists the parameters for the TLS Close Notify action.

TLS Close Notify Action Parameters

Parameter	Description	Valid Values
Enabled	An SSL Alert message with a Warning alert level and an alert code of `0x00`. If set to true, a TLS Close Notify message is sent to the peer. Terminating encrypted sessions with Close Notify messages prevents truncation attacks by informing the peer that no more encrypted data will be sent.	true or false

The table below lists the parameters for the TLS Discard Encrypted Data action. The TLS Discard Encrypted Data action can be inserted at any point in an SSL/TLS flow to discard encrypted data from that point forward instead of decrypting it.

TLS Discard Encrypted Data Action Parameters

Parameter	Description	Valid Values
Count Discarded Data	If set to true, update bulk decryption statistics, otherwise do not update statistics.	true or false

Live AppSim

The Live AppSim component is used to regenerate the dynamics of production network traffic characteristics that have been captured by the Ixia TrafficREWIND™ application.

TrafficREWIND Overview

TrafficREWIND is a virtual appliance that translates production network traffic insights into real traffic stimulus for test environments and provides the following:

- Faster fault analysis and reproduction capabilities.
- Reference architectures and pre-deployment validation with production-like application mixes.
- Relevant what-if scenarios by combining real production traffic with other test traffic including security strikes, incremental applications, or even fuzzing.

TrafficREWIND offers a scalable real-time architecture to record and synthesize traffic characteristics over extended periods of time (up to 7 days). Subsequently, up to 1 day of traffic summary test configuration data can be imported into BreakingPoint.

Recorded network traffic dynamics include:

- application type and behavior
- bandwidth distribution
- temporal nature

BreakingPoint allows you to not only replicate the traffic profile with the associated real-world applications, but also adds a new test dimension of dynamically changing traffic composition over time to model the temporal nature of networks and applications.

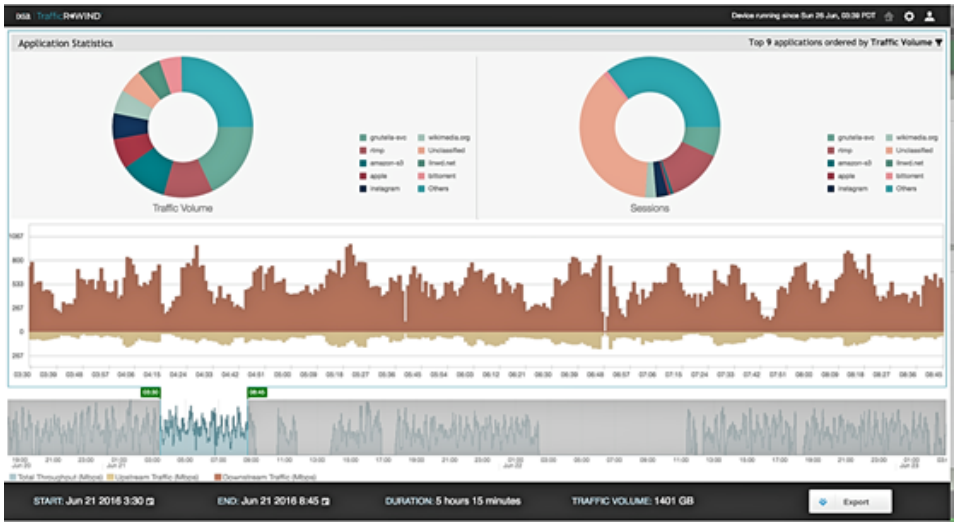
Note: See the TrafficREWIND User Guide for detailed information about the product.

Exporting Data from TrafficREWIND

To transfer the network traffic information from TrafficREWIND to BPS, the network traffic information must first be exported.

To export data from TrafficREWIND:

1. In the TrafficREWIND user interface select a range of time using the left and right interval selectors. For example, see the timeline displayed towards the bottom of image shown below where the selected range is: 3:30 – 8:45.



2. Click **Export**.
3. Name and **Save** the file (xxxx.bptx).

> **Note:** Several aspects of the traffic information saved in the exported file can be modified in BreakingPoint before it is replayed (these options are described later in this section).

> **Note:** When traffic data is exported from a TrafficREWIND controller in a distributed setup, the export file contains traffic data collected by all the attached processors in the selected time range.

Importing TrafficREWIND Data into a BPS Live Profile

The following options are available:

Importing traffic data which is exported from a TrafficREWIND standalone system

Importing traffic data which is exported from a TrafficREWIND controller in a distributed system

Importing traffic data which is exported from a TrafficREWIND standalone system

1. In BPS select, **Managers** > **Live Profiles**.
2. In the Browse Live Profiles window select **Import**.
3. Click **Choose File**.
4. Select a .bptx file that was exported from a TrafficREWIND standalone system.
5. Click **Upload**.

   After the Live Profile file has been uploaded, a chart displaying the selected range of time selected will appear. The traffic characteristics will be segmented into 15 minute increments which is the smallest configurable unit for Live AppSim traffic.

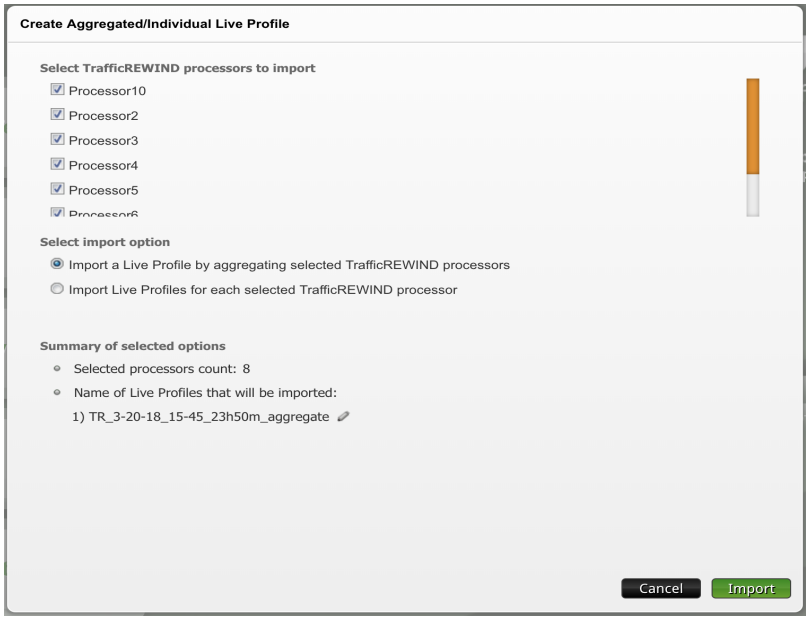
6. At this time, the interval selectors shown in the BPS UI can be adjusted if you want to limit the range of time/data characteristics that will be replicated by BPS. When the interval selectors are configured to select a time range, the throughput Key Performance Indicators displayed (Max throughput, Max sessions per second and Max concurrent sessions) will also change according to the time range selection
7. Click **Apply**.
8. Click **Save** to name and save the Live Profile.

Importing traffic data which is exported from a TrafficREWIND controller in a distributed system

When the export file of a TrafficREWIND distributed setup is imported into BPS, it enables the following options:

- Regeneration of aggregated traffic summary data of all the attached processors
- Regeneration of aggregated traffic summary data of different combinations of processors
- Regeneration of traffic summary data individually for each processor

1. In BPS select **Managers** > **Live Profiles**.
2. In the Browse Live Profiles window select **Import**.
3. Click **Choose File**.
4. Select a .bptx file that was exported from a TrafficREWIND controller in a distributed setup.
5. Click **Upload**. After the Live Profile file has been uploaded, the **Create Aggregated/Individual Live Profile** window appears.



The following import options are available:

- Import a live profile by aggregating selected TrafficREWIND processors
- Import live profiles for each selected TrafficREWIND processors

Import a live profile by aggregating selected TrafficREWIND processors

This option imports an aggregated live traffic summary of all of the selected TrafficREWIND prodcessors.

a. Select the live traffic summary of one or multiple TrafficREWIND processors.
b. Select the first option below **Select import options**: "Import a live profile by aggregating selected TrafficREWIND processors".

Tip: You can look into the summary of the selected options before clicking on **Import**. You can also rename the aggregated profile by clicking the pencil icon.

c. When you click **Import**, the aggregated live traffic summary of all selected processors will be imported.

After the aggregated live profile is imported, a chart displaying the selected range of time selected will appear. The traffic characteristics will be segmented into 15 minute increments which is the smallest configurable unit for Live AppSim traffic.

d. Click **Apply**.
e. Click **Save** to name and save the Live Profile.

Import live profiles for each selected TrafficREWIND processors

This option imports individual live traffic summaries of each selected TrafficREWIND processor.

a. Select the live traffic summary of one or multiple TrafficREWIND processors.
b. Select the second option below **Select import options**: "Import live profiles for each selected TrafficREWIND processors".

Tip: You can look into the summary of the selected options before clicking on **Import**. You can also rename the aggregated profile by clicking the pencil icon.

c. When you click **Import**, live traffic summaries of each of the selected processors will be imported.
d. Select a profile and click **Open** to view the chart displaying the selected range of time. The traffic characteristics will be segmented into 15 minute increments (which is the smallest configurable unit for Live AppSim traffic).
e. Click **Apply**.
f. Click **Save** to name and save the Live Profile.

Loading the Live Profile into a Live Appsim Test Component

A Live Profile is similar to an Application Profile.

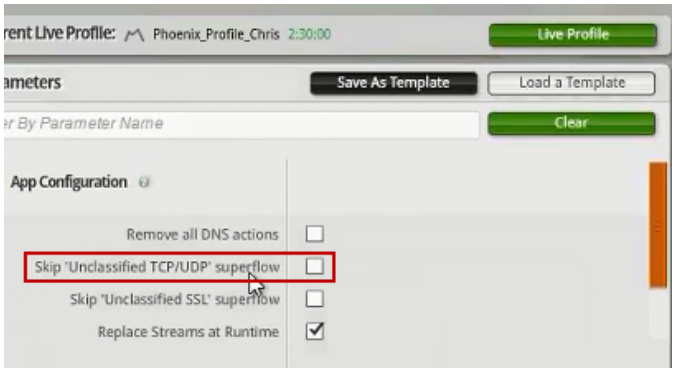
1. Create a new test or open a recent test.
2. Click **Add New** Test Component.
3. Select the Live AppSim component.
4. Click **Live Profile**.
5. Select a Live Profile that was imported into the BreakingPoint System.
6. Optionally configure the Live Profile settings, to modify how the traffic characteristics are replayed.

Live Profile Parameters

Most of the Live Profile parameters are the same as the Application Simulator Parameters on page 752. The information below describes App Configuration and Data Rate settings that are unique to Live Profile.

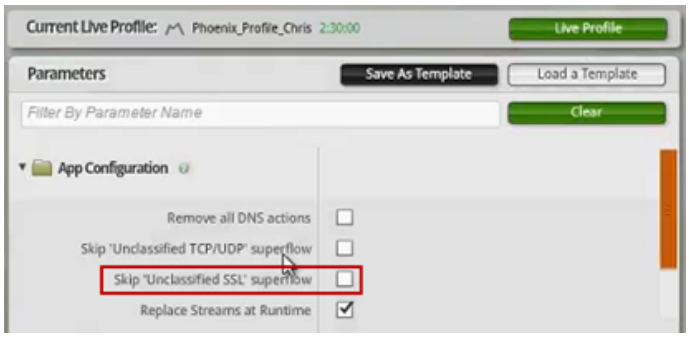
Skip 'Unclassified TCP/UDP'

Selecting the **Skip 'Unclassified TCP/UDP'** superflow option specifies that BPS will not replay unclassified TCP/UDP traffic (TCP/UDP traffic that TrafficREWIND could not classify or associate with an application).



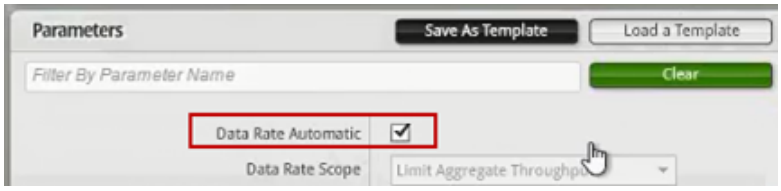
Skip 'Unclassified SSL'

Selecting the **Skip 'Unclassified SSL'** superflow option specifies that BPS will not replay unclassified SSL traffic (SSL traffic that TrafficREWIND could not classify or associate with an application).



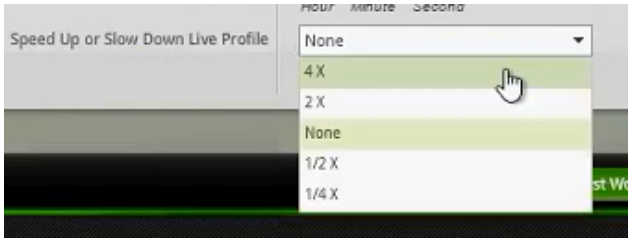
Data Rate Automatic

When the **Data Rate Automatic** option in the Data Rate settings area is selected, Live AppSim will attempt to achieve throughput as observed by TrafficREWIND and captured in the exported .bptx file. Live AppSim will also replicate the live profile as recorded without modification. If this option is deselected, the standard profile options for limiting and adjusting throughput become available.



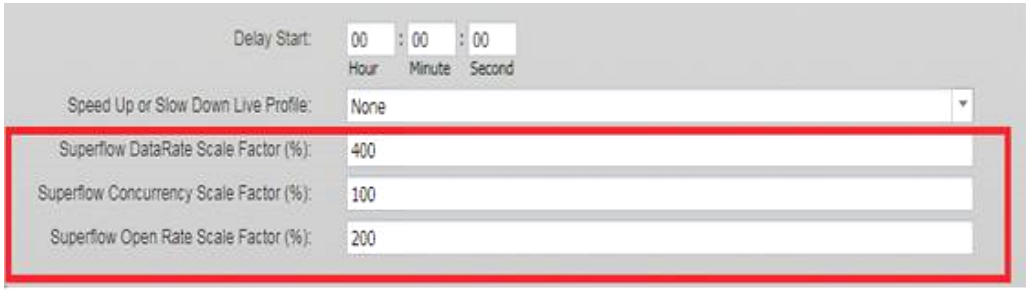
Speed Up or Slow Down Live Profile

You may **Speed Up or Slow Down Live Profile** traffic by using the options available in this feature's drop-down list. For example, selecting the "4X" option will play back a 2 hour Live Profile in 30 minutes.



Data Rate, Concurrency and Open Rate Scaling

The following options provide throughput scaling (up or down) for the Superflow's **Data Rate**, **Concurrency** and **Open Rate**. Concurreny refers to the number of concurrent flows. Open Rate refers to rate for opening connections. These options are configured as a percentage of the original value. The default value for all options is 100.



Note: The result of these settings are displayed in the LiveAppsim tab of Real Time Statiscs and in Report as **Desired data rate**. The Desired data rate is expressed as a percentage (%) of Original data rate.

Client Simulator

Client Simulator enables you to connect the BreakingPoint system to a server device under test so the chassis can act as a client generating connections to the server. Client Simulator sends a single Super Flow to the DUT and enables you to search for specific responses from the DUT.

In order to use Client Simulator, you will need to create Super Flows that use Conditional Requests. Conditional Requests define the specific responses (i.e., strings or patterns) you expect to see from the DUT. The Client Simulator component will track the number of responses from the server that match the string matches defined within the Conditional Requests for the Super Flow. This data will be available in the Response Summary of the test report.

Client Simulator Parameters

The following table lists the parameters for the Client Simulator test component.

Client Simulator Parameters

Parameter	Description	Valid Values
Data Rate.Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable **Note:** Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data Rate.Data Rate Scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	Limit Per-Interface Throughput – Uses the data rate as the limit for the transmitting and receiving interfaces. Limit Aggregate Throughput – Uses the data rate as an aggregate limit for the test component.
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second

Parameter	Description	Valid Values
Data Rate.Data Rate Type	Sets how the component determines the data rate it will use for its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.
Data Rate.Minimum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces, if **Data Rate.Data Rate Type** is set to **Constant**. Otherwise, this is the minimum value used by the test if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps

Parameter	Description	Valid Values
Session Ramp Distribution.SYN Only Retry Mode	Defines the behavior of the TCP Retry Mechanism when dealing with the initial SYN packet of a flow.	Continuous – Continue sending SYN packets, even if we have ran out of retries (Retry Count). Continuous with new session – Same as Continuous, except we change the initial sequence number every Retry Count loop(s). Obey Retry Count – Send no more than Retry Count initial SYN packets.
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	Sets the maximum number of concurrent sessions that can be set up by the system at a given time. This value must be greater than **Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Maximum Simultaneous Active Flows	Sets the maximum number of concurrent sessions that can be active in the system at the same time.	1 – 20,000,000 See Maximum Simultaneous Active Flows Details on page 811.
Session/Super Flow Configuration.Maximum Super Flows Per Second	Sets the maximum number of sessions that can occur per second. This value must be greater than **Session/Super Flow Configuration.Target Minimum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Engine Selection	Select the type of engine with which to run the test component. Select Advanced to enable the default, full-featured engine. Select Simple to enable a simpler, higher-performance, stateless engine.	Advanced (Features) Simple (Performance)
Session/Super Flow Configuration.Performance Emphasis	Adjusts whether the advanced engine's flow scheduler favors opening new sessions, sending on existing sessions, or a mixture of both. Note:Ramp-up phases automatically emphasize Sessions, ramp-down phases emphasize Throughput.	Balanced – Emphasize both opening new sessions and sending data n existing sessions equally (default) Simultaneous Sessions – Emphasize opening new sessions Throughput – Emphasize sending data on existing sessions
Session/Super Flow Configuration.Resource Allocation Override	Allows you to override the amount of CPU and other resources allocated to this test component. The value indicates the amount of resources to use on a single processor.	Automatic – The system will estimate the amount of resources required by a particular component. % – Manually tune the amount (in terms of percentages) of resources allocated to this component

Parameter	Description	Valid Values
Session/Super Flow Configuration.Statistic Detail	Adjusts the level of statistics to be collected. Decreasing the number of statistics collected can increase performance and allow for targeted reporting.	Maximum – Enable all possible statistics Application Only – Enable only Application statistics (L7) Transport Only – Enable only Transport statistics (L4/L3) Minimum – Disable most statistics
Session/Super Flow Configuration.Unlimited Super Flow Open Rate	Disables the limit on the open rate. If this option is set to **false**, then the open rate is limited by the connection establishment rate. If this option is set to **true**, then the system will open the Super Flows as quickly as the bandwidth allows; therefore, the number of closed Super Flows will be closer to the maximum number of simultaneous Super Flows set for the component.	true or false
Session/Super Flow Configuration.Unlimited Super Flow Close Rate	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
Session/Super Flow Configuration.Target Minimum Simultaneous Sessions	The number of sessions that must open to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Target Minimum Super Flows Per Second	The number of sessions per second that must be reached to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Target Number of Successful Matches	Specifies the minimum number of successful matches required to pass in the final results.	N/A
IPv4 Configuration.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv4 Configuration.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff
IPv6 Configuration.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv6 Configuration.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF
IPv6 Configuration.Flowlabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF

Parameter	Description	Valid Values
Source Port. Distribution Type	Sets how the component will obtain the source port numbers.	Constant – Uses Source Port.Minimum port number as the source port.
		Random – Uses random values between Source Port.Minimum port number and Source Port.Maximum port number.
		Range – Increments Source Port.Minimum port number by one until it reaches Source Port.Maximum port number. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
Source Port.Minimum Port Number	Sets the minimum source port number, if Source Port.Port distribution type is Range or Random. Otherwise, this will be the value used for the source port.	0 – 65,535
Source Port.Maximum Port Number	Sets the maximum source port number, if Source Port.Port distribution type is Range or Random.	0 – 65,535
TCP Configuration.Maximum Segment Size (MSS)	Sets the maximum segment size (MSS) that is used during the ramp up phase. MSS is the maximum size that a client advertises it can receive.	512 – 9,146
TCP Configuration.Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing.	1 – 120

Parameter	Description	Valid Values
TCP Configuration.Reset at End	Indicates whether or not a test should reset all existing TCP connections at the end. If enabled, all TCP connections will reset if the test either ends naturally or is canceled.	true or false
TCP Configuration.Retry Quantum	Sets the amount of time that elapses before a connection is retried.	1 – 2,000
TCP Configuration.Retry Count	Sets the number of times a connection is attempted before it is canceled.	1 – 3
TCP Configuration.Delay ACKs	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
TCP Configuration.Initial Receive Window	Sets the size of the initial receive window for a new connection.	1 – 65,535
TCP Configuration.TCP Window Size	This value will allow the tcp connection to negotiate the window scale option as per RFC1323. The value set will be the maximum window size that the connection will attempt to negotiate. The window scale factor is the number of bits by which to shift the window size. Send and receive windows are negotiated separately based on the configuration of the equipment at each end.	0-14 (0=disabled)

Parameter	Description	Valid Values
TCP Configuration.Dynamic receive window size	Dynamic receive window size = 1 (box checked - this is the default setting) Note that earlier versions of BPS had this option enabled by default. • The receive window size is increased by MSS with each received window until it reaches the maximum value (65535 * 2^window_scale_factor). • If Delay ACK is enabled, the receive window size is doubled on window full. Dynamic received window size = 0 (box unchecked) • In order to keep the window fixed, the "Dynamic Receive Window Size" needs to be set to off. • The receive window size is not increased. • It remains fixed in all scenarios.	true or false
TCP Configuration.Add Segment Timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
TCP Configuration.Piggy-back Data on 3-way Handshake ACK	Determines whether to add Data to the client ACK packet of the TCP handshake.	true or false
TCP Configuration.Piggy-back Data on Shutdown FIN	Determines whether to add Data to the client FIN packet of the TCP shutdown.	true or false
TCP Configuration.Initial Congestion Window	Determines the size of the initial congestion window.	1 – 16

Parameter	Description	Valid Values
TCP Configuration.Explicit Congestion Notification	Determines the support level for ECN.	Disable – Disables all support for ECN Support ECN – ECN will be supported if the remote host initiates it first. Use ECN – ECN will be initiated for new connections.
TCP Configuration.Raw Flags	Allows the specification of the TCP flags as bits.	-1 – 255
TCP Configuration.Connect Delay	Adds two delays (one between the TCP SYN-ACK packet and the final ACK, and one between the final ACK and the first data packet) in the TCP state machine when the delay is set to a value greater than 0.	0 – 10
App Configuration.Streams Per Super Flow	The maximum number of streams that will be instantiated for an individual Super Flow at one time. The effective number may be limited by the number of Super Flows in the test. Setting this parameter to a lower number will cause tests to initialize faster and provide less random application traffic. Setting the parameter to a higher number will allow tests to initialize slower with greater randomization in application traffic, especially for static flows.	1 – 256
App Configuration.Content Fidelity	Indicates whether applications will generate more complex, dynamic traffic, or will generate simpler, possibly more performant, traffic. This setting may not have an effect for all application protocols and profiles.	High – for more complex/dynamic traffic Normal – for simpler traffic

Parameter	Description	Valid Values
App Configuration.Content Fidelity	Enabled by default, this feature ensures that any Super Flow that is represented as a stream that has been active for at least 5 to 15 seconds, and no longer has active references could be removed, rebuilt (via AppManager and StreamServer), and then filled again and made active. Weighting will be honored in the case of an AppProfile. When this setting is disabled, BP will never replace the stream(s) associated with any SuperFlow, and weights are not honored.	true or false
App Configuration.Delay Start	Delays the start of a test component by the time specified. Floating values are supported.	hh:mm:ss
App Configuration.Super Flow	Determines the application simulated and settings specific for that application.	Available Super Flow

Note: * 1 Gb blades will only support up to 5,000,000 simultaneous sessions at a rate of 500,000 sessions per second.

SMB and SMB2 Settings for Client Simulator

lists the settings for the SMB and SMB2 protocols.

SMB and SMB2 Settings for Client Simulator

Setting	Description	Valid Options
SMB Authenticate	This is a mandatory action and must be the first action. Supply the SMB or SMB2 Share Name, the local Username and Password for NTLM and LanManager authentication, and select the Protocol Version. SMB2 requires Windows Vista, Windows Server 2008, or Windows 7.	SMB Share Name - The SMB Share Name is a mandatory parameter and it is the SMB or SMB2 Share Name to authenticate against. Ensure that the local account user name has the appropriate permissions (READ, WRITE, DELETE, etc.). If you are using Samba, you can remotely determine available shares using `smbclient -L 192.168.1.2`. If you are using Windows, you can remotely determine available shares using `net view \\192.168.1.2`.
		SMB UserName - The SMB UserName is a local user name on the remote system that has the appropriate permissions to access the SMB or SMB2 Share Name.
		SMB Password - The SMB Password is the password for the local user name on the remote system that has the appropriate permissions to access the SMB or SMB2 Share Name.

Setting	Description	Valid Options
SMB Write File to Share	The SMB Write File to Share is the file name to be written to the SMB or SMB2 Share Name. To ensure a unique file name, use the `%f` parameter to specify the flow and the `%g` parameter to include a goto iteration in the filename. For example, `testfile-%f-%g.txt`.	SMB Minimum Random Filesize - When no resource file is specified, this setting determines the minimum possible size of the randomly-generated file. Setting this parameter will cause the File Contents setting to be ignored.
		SMB Maximum Random Filesize - When no resource file is specified, this setting determines the maximum possible size of the randomly-generated file. Setting this parameter will cause the File Contents setting to be ignored.
		SMB File Contents - The imported contents to write to the file name on the SMB or SMB2 Share Name.
SMB Append file on share	The file name to be appended to the SMB or SMB2 Share Name. To ensure a unique file name, use the `%f` parameter to specify the flow and the `%g` parameter to include a goto iteration in the filename. For example, `testfile-%f-%g.txt`. If the file does not already exist, it is created. If the file already exists, it is appended.	**SMB Minimum Random Filesize** - When no resource file is specified, this setting determines the minimum possible size of the randomly-generated file. Setting this parameter will cause the File Contents setting to be ignored.

Setting	Description	Valid Options
		SMB Maximum Random Filesize - When no resource file is specified, this setting determines the maximum possible size of the randomly-generated file. Setting this parameter will cause the File Contents setting to be ignored.
		SMB File Contents - The imported contents to Append to the File Name on the SMB or SMB2 Share Name.
Verify File from Share	The file name from the SMB or SMB2 Share Name to read and compare against File Contents. If there is a data mismatch between the data read back from the SMB or SMB2 File Name on the Share Name and the File Contents, an exception will be displayed. If a Unique File Name using the `%f` and `%g` options was used for the Write action, it should also be used in the Verify action. For example, `testfile-%f-%g.txt`. The imported File Contents that will be compared against the data read back from the File Name on the remote SMB or SMB2 Share Name. The same source file should be used as the file contents chosen for the write action.	
Read File from Share	The File Name from the SMB or SMB2 Share Name to read. If a Unique File Name using the `%f` and `%g` options was used for the Write action, it should also be used in the Verify action. For example, `testfile-%f-%g.txt`.	
Delete File from Share	The File Name from the SMB or SMB2 Share Name to delete. If a Unique File Name using the `%f` and `%g` options was used for the Write action, it should also be used in the Delete action. For example, `testfile-%f-%g.txt`.	

Setting	Description	Valid Options
Disconnect	No Parameters. Disconnects from Share name and Logs off User Name.	

Recreate

The Recreate test component currently supports importing capture files in standard libpcap (used by tools such as tcpdump) and libpcap files compressed with gzip. Two modes for recreating traffic are supported: Normal and Raw Playback. Capture files up to 4GB can be imported.

In Normal mode, the TCP, UDP, and SCTP payloads from the imported capture file are converted into our internal file format. The Recreate test component rewrites the Layer 2 and Layer 3 data to match the traffic parameters specified for the domain.

Note: When exceedingly large, single-session PCAP files are used in tests with very short durations, the PCAP file will not generate traffic for the test. When this occurs, the test duration must be increased and the test must be rerun.

In Raw mode, a copy of the originally imported capture file is recreated on the wire as-is, without any modifications to any of the L2, L3, L4, or payload information. This mode is useful for testing Layer 2, Layer 3, and Layer 4 headers.

Note: When importing a capture file that contains corrupt Layer 2, 3, or 4 headers like those created by the Stack Scrambler component for replay in the Recreate component, set Replay capture file without modification to true to properly replay the traffic.

BreakingPoint can import a maximum of 4GB of any imported capture file. When this maximum is reached, the BreakingPoint will stop converting the PCAP file into our internal file format. A copy of the original capture file is also imported into the system. This copy of the original capture file is not modified and is not limited by the Export Size, BPF filter options or the 4GB limit. This copy of the original capture file is used when the Recreate Replay capture file without modification option is set to true. The Replay capture file without modification option has a runtime BPF filter parameter that can be used to limit which packets are replayed.

Note: Replay capture file without modification means that the capture file (in libpcap format) will be recreated on the wire as-is, without modifying or changing any of the L2, L3, L4, or payload information.

The Replay Capture File In Single Mode accepts the first 255 flows. Because the first 255 flows of the default pcap have only UDP flows, the Replay Capture File In Single Mode will play only UDP flows and will not play any TCP flows.

General Behavior of Recreate

When a capture file is imported through Capture Manager, it is stored in three different ways, allowing the Recreate Component to replay the traffic in three different modes:

Replay Packet Capture with Modification

- During capture file import, each complete flow's Layer 4 through Layer 7 data is stored in individual files and grouped according to protocol.
- The delay between each packet within each flow is recorded.
- Periodically, the bandwidth in use is recorded.
- Periodically, the total number of currently open sessions is recorded.
- Periodically, the weight, or relationship of currently open flows grouped by protocol, is recorded.
- The duration of the test is recorded.

Note: These individual files grouped by protocol, are only used when Replay Capture File Without Modification is set to false.

When the Recreate Component is run in this mode, flows are randomly chosen according to protocol weight. The Layer 2, Layer 3, and Layer 4 headers honor Network Neighborhood and Recreate Configuration Parameters such as Session/Super Flow Configuration, IPv4 Configuration, IPv6 Configuration, Source Port, TCP Configuration, and Data Rate. Modification Options are ignored, except for General Behavior.

How the General Behavior Settings are Interpreted in this Mode

Use User-Specified Settings

- The Data Rate and Session/Super Flow Configuration Parameters are honored.
- The duration of the test is controlled by the Load Profile.

Use Capture File Settings

- Periodically recorded bandwidth in use is honored during the duration of the test.
- Periodically recorded total number of currently open sessions is honored.
- The duration of the test is set to the duration of the imported capture file.

Replay Packet Capture without Modification

During capture file import, an exact copy of the imported capture file is stored. This mode is useful for testing Layer 2, Layer 3, and Layer 4 anomalies such as fragmentation, invalid packets, retransmissions, and other protocols not properly converted into Layer 4 through Layer 7 payloads.

Note: This file is only used when Replay Capture File Without Modification is set to true.

Note: IPsec is NOT supported when the Replay Packet Capture without Modification option is enabled.

When the Recreate Component is run in this mode, the test duration defined by the Load Profile is not honored and the test ends when all packets are retransmitted. In this mode, Network Neighborhood settings are also ignored. All packets are sent from the client interface. This mode provides the ability to:

- Specify BPF Filter
- Specify start and end packet to be transmitted

- Specify the Number of Times to Loop Capture File

How the General Behavior Settings are Interpreted in this Mode

With the Use User-Specified Settings, the capture file is retransmitted, ignoring the delay between each packet.

With the Use Capture File Settings, the capture file is retransmitted, honoring the delay between each packet.

Replay Packet Capture with Modification (Single Mode)

During capture file import, the Layer 4-Layer 7 data of the first 255 flows is stored in a single file.

Use case: This mode is useful when it is necessary to control the order of the individual flows.

- The delay between each packet is recorded.
- Periodically, the bandwidth in use is recorded.
- Periodically, the total number of currently open sessions is recorded.
- The delay between each packet within each flow is recorded.
- The duration of the test is recorded.

Note: This file is only used when Replay Capture File In Single Mode is set to true.

How the General Behavior Settings are Interpreted in this Mode

Use User-Specified Settings

- The Data Rate and Session/Super Flow Configuration Parameters are honored.
- The duration of the test is controlled by the Load Profile.

Use Capture File Settings

- Periodically recorded bandwidth in use is honored during the duration of the test.
- Periodically recorded total number of currently open sessions is honored.

The duration of the test is set to the duration of the imported capture file.

Playback Settings

There are two ways to play back the PCAP file:

- Use capture file settings
- Use user-specified settings

Using the first setting, **Use capture file settings**, Recreate in Normal Mode will use the data rate, maximum simultaneous Super Flows, sessions per second, test duration, inter-packet delays, application payloads, and destination ports from the PCAP file. All other fields/parameters will be taken from the Parameters tab. The source port will be randomized.

Using the first setting, **Use capture file settings**, Recreate in Raw Playback Mode will IGNORE the data rate, maximum simultaneous Super Flows, sessions per second, test duration, inter-packet

delays, application payloads, and destination ports from the capture file. The capture file will honor the interpacket delay of the original capture file. A BPF filter can be applied to send only packets that match the filter, and TCP and UDP ports can be changed.

Using the first setting, Use capture file settings, Recreate, with the parameter Replay Capture File without Modification set to true, will IGNORE the data rate, maximum simultaneous Super Flows, sessions per second, test duration, inter-packet delays, application payloads, and destination ports from the capture file. The capture file will honor the interpacket delay of the original capture file. A BPF filter can be applied to send only packets that match the filter, and TCP and UDP ports can be changed. The test will end when all the packets are replayed or the time duration of the test expires, whichever comes first.

Note:

- Use capture file settings do not accept TTL/TOS/DSCP values. Also, this setting does not count the number of times a capture is looped or replayed.
- When a file is played back in a Recreate component with Replay capture file without modification set to true and Use capture file settings selected, the inner packet delay between packets can differ from the original packet capture due to additional processing overhead. This can cause the resulting Recreate timestamp to appear to be longer than the original PCAP.

Using the second setting, **Use User-specified settings**, Recreate in Normal Mode will only use the application payload and destination ports from the PCAP file. The source ports will be randomized and all other fields/parameters will be taken from the Parameters tab. The interpacket delays will be set to '0'.

The purpose behind these two settings is to allow you to use the application payload from the PCAP file, but still have some control over how the file is replayed. **Use capture file settings** essentially lets you replay the PCAP as it is, whereas **Use User-specified settings** enables you to control how fast or slow the traffic is replayed. If you want your PCAP file replayed as closely to the original as possible, you should use **Use capture file settings**. If you only want to replay your application payload, and change the speed at which it is replayed, you should use **Use User-specified settings**.

Note: If **Use capture file settings** is selected instead of **Use User-specified settings**, the duration of the capture file overrides the Load Profile durations for the recreate test. The duration of the packet capture can be viewed from Capture Manager and is labeled Time Length.

With Use User-specified settings, Recreate in Raw Playback Mode will retransmit the capture file as fast as possible. The capture file can be looped multiple times, a BPF filter can be applied to send only packets that match the filter, and TCP and UDP ports can be changed.

Recreate Parameters

The following table lists the parameters for the Recreate test component.

Recreate Parameters

Parameter	Description	Valid Values
Session Ramp Distribution.SYN Only Retry Mode	Defines the behavior of the TCP Retry Mechanism when dealing with the initial SYN packet of a flow.	Continuous – Continue sending SYN packets. Continuous with new session – Continue sending SYN packets and change the initial sequence number every "Retry Count" loop(s). Obey Retry Count - Send no more than Retry Count initial SYN packets.
Session/Super Flow Configuration.Maximum Simultaneous Super Flows	Sets the maximum number of concurrent sessions that can be set up by the system at a given time. This value must be greater than **Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
Session/Super Flow Configuration.Maximum Simultaneous Active Flows	Sets the maximum number of concurrent sessions that can be active in the system at the same time.	1 – 20,000,000 See Maximum Simultaneous Active Flows Details on page 811
Session/Super Flow Configuration.Maximum Super Flows Per Second	Sets the maximum number of sessions that can occur per second. This value must be greater than **Session/Super Flow Configuration.Target Minimum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Engine Selection	Select the type of engine with which to run the test component. Select Advanced to enable the default, full-featured engine. Select Simple to enable a simpler, higher-performance, stateless engine.	Advanced (Features) Simple (Performance)

Parameter	Description	Valid Values
Session/Super Flow Configuration.Performance Emphasis	Adjusts whether the advanced engine's flow scheduler favors opening new sessions, sending on existing sessions, or a mixture of both. Note:Ramp-up phases automatically emphasize Sessions, ramp-down phases emphasize Throughput.	Balanced – Emphasize both opening new sessions and sending data n existing sessions equally (default) Simultaneous Sessions – Emphasize opening new sessions Throughput – Emphasize sending data on existing sessions
Session/Super Flow Configuration.Resource Allocation Override	Allows you to override the amount of CPU and other resources allocated to this test component. The value indicates the amount of resources to use on a single processor.	Automatic – The system will estimate the amount of resources required by a particular component. % – Manually tune the amount (in terms of percentages) of resources allocated to this component
Session/Super Flow Configuration.Statistic Detail	Adjusts the level of statistics to be collected. Decreasing the number of statistics collected can increase performance and allow for targeted reporting.	Maximum – Enable all possible statistics Application Only – Enable only Application statistics (L7) Transport Only – Enable only Transport statistics (L4/L3) Minimum – Disable most statistics
Session/Super Flow Configuration.Unlimited Super Flow Open Rate	Disables the limit on the open rate. If this option is set to **false**, then the open rate is limited by the connection establishment rate. If this option is set to **true**, then the system will open the Super Flows as quickly as the bandwidth allows; therefore, the number of closed Super Flows will be closer to the maximum number of simultaneous Super Flows set for the component.	true or false

Parameter	Description	Valid Values
Session/Super Flow Configuration.Unlimited Super Flow Close Rate	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
Session/Super Flow Configuration.Target Minimum Simultaneous Super Flows	The number of sessions that must open to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Simultaneous Super Flows**.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb) 0 – When Modification Options.Replay capture file without modification is set to True
Session/Super Flow Configuration.Target Minimum Super Flows Per Second	The number of sessions per second that must be reached to pass the test. This value must be less than **Session/Super Flow Configuration.Maximum Super Flows Per Second**.	1 – 750,000* (10Gb) or 500,000 (1Gb)
Session/Super Flow Configuration.Target Number of Successful Matches	Specifies the minimum number of successful matches required to pass in the final results.	N/A
IPv4 Configuration.TTL	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv4 Configuration.TOS/DSCP	Configures the TOS field used for all IP packets.	0 – ff

Parameter	Description	Valid Values
IPv6 Configuration.Hop Limit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
IPv6 Configuration.Traffic Class	Defines the Traffic Class field used for all IP packets.	0 – FF
IPv6 Configuration.Flowlabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
Source Port.Port Distribution Type	Sets how the component will obtain the source port numbers.	Constant – Uses Source Port.Minimum port number as the source port.
		Random – Uses random values between Source Port.Minimum port number and Source Port.Maximum port number.
		Range – Increments Source Port.Minimum port number by one until it reaches Source Port.Maximum port number. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
Source Port.Minimum Port Number	Sets the minimum source port number, if Source Port.Port distribution type is Range or Random. Otherwise, this will be the value used for the source port.	0 – 65,535
Source Port.Maximum Port Number	Sets the maximum source port number, if Source Port.Port distribution type is Range or Random.	0 – 65,535

Parameter	Description	Valid Values
TCP Configuration.Maximum Segment Size (MSS)	Sets the maximum segment size that is used during the ramp up phase.	512 – 9,146
TCP Configuration.Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing.	0 – 120
TCP Configuration.Reset at End	Indicates whether a test should reset all existing TCP connections at the end.	true or false
TCP Configuration.Retry Quantum	Sets the amount of time that elapses before a connection is retried.	100 – 2,000
TCP Configuration.Retry Count	Sets the number of times a connection is attempted before it is canceled.	1 – 3
TCP Configuration.Delay ACKs	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
TCP Configuration.Initial Receive Window	Sets the size of the initial receive window for a new connection.	1 – 65,535

Parameter	Description	Valid Values
TCP Configuration.TCP Window Size	This value will allow the tcp connection to negotiate the window scale option as per RFC1323. The value set will be the maximum window size that the connection will attempt to negotiate. The window scale factor is the number of bits by which to shift the window size. Send and receive windows are negotiated separately based on the configuration of the equipment at each end.	0-14 (0=disabled)
TCP Configuration.Dynamic receive window size	Dynamic receive window size = 1 (box checked - this is the default setting) Note that earlier versions of BPS had this option enabled by default. • The receive window size is increased by MSS with each received window until it reaches the maximum value (65535 * 2^window_scale_factor). • If Delay ACK is enabled, the receive window size is doubled on window full. Dynamic received window size = 0 (box unchecked) • In order to keep the window fixed, the "Dynamic Receive Window Size" needs to be set to off. • The receive window size is not increased. • It remains fixed in all scenarios.	true or false

Parameter	Description	Valid Values
TCP Configuration.Add Segment Timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
TCP Configuration.Piggy-back Data on 3-way Handshake ACK	Determines whether to add data to the client ACK packet of the TCP handshake.	true or false
TCP Configuration.Piggy-back Data on Shutdown FIN	Determines whether to add Data to the client FIN packet of the TCP shutdown.	true or false
TCP Configuration.Initial Congestion Window	Determines the size of the initial congestion window.	1 – 16
TCP Configuration.Explicit Congestion Notification	Determines the support level for ECN.	Disable – Disables all support for ECN Support ECN – ECN will be supported if the remote host initiates it first. Use ECN – ECN will be initiated for new connections.
TCP Configuration.Raw Flags	Allows the specification of the TCP flags as bits.	-1 – 255
TCP Configuration.Connect Delay	Adds two delays (one between the TCP SYN-ACK packet and the final ACK, and one between the final ACK and the first data packet) in the TCP state machine when the delay is set to a value greater than 0.	0 – 10

Parameter	Description	Valid Values
Data Rate.Data Rate Unlimited	Defines whether data rate limiting should be enabled or disabled for the test.	Enable or Disable **Note:** Enabling this option also enables a rate check function that can affect performance. Performance will improve if a static value is configured and this option is disabled.
Data rate.Data Rate Scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	Limit Per-Interface Throughput – Uses the data rate as the limit for the transmitting and receiving interfaces. **Limit Aggregate Throughput** – Uses the data rate as an aggregate limit for the test component.
Data Rate.Data Rate Unit	Sets the unit of measurement for the data rate.	Frames/second or Megabits/second
Data Rate.Data Rate Type	Sets how the component determines the data rate it will to send its traffic.	Constant – Uses Minimum Data Rate as the data rate. Random – Selects a random value between Minimum Data Rate and Maximum Data Rate as the data rate. Range – Starts at Minimum Data Rate and increments until it reaches Maximum Data Rate. The system uses an algorithm that determines the incremental value that will increase Minimum Data Rate until it reaches Maximum Data Rate.
Data Rate.Minimum Data Rate	The minimum value used by the test if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 Mbps (1Gb)

Parameter	Description	Valid Values
Data Rate.Maximum Data Rate	Sets the maximum throughput that the test will consume over the transmitting and receiving interfaces; this parameter is used only if **Data Rate.Data Rate Type** is set to **Range** or **Random**.	1 – 14,880,095 (1Gb) or 148,800,952 (10Gb) fps 1 – 10,000 (10Gb) or 1,000 (1Gb) Mbps
Modification Options.Replay Capture File Without Modification	This mode replays libpcap formatted capture files without modifying Layer 2 through Layer 7. Recreate will derive all settings and configurations from the PCAP file. When operating in this mode, Recreate will only track the Interface Stats. Also, the capture file can be replayed at a maximum of 200 Mbps, depending on the type of system you are using.	true or false
Modification Options.Replay Capture File In Single Mode	Replays the L4 Payloads of all of the flows (up to 255) of the imported capture file in order.	true or false
Independent Flow Hosts in Single Mode	This feature is only available if Single Mode is enabled. Causes each flow to have different client/server IP address when in Single Mode. This option is essential to support dual stack with server injection feature.	true or false

Parameter	Description	Valid Values
Replay Capture File With Server IP Intact	Uses the packet capture's server IP and the client IP from the NN. Uses the server mac address from the server subnet in the NN. Unique IP addresses will have unique MAC address. **Note:** DUTs which modify source address will not be supported. Gateway is not supported - the destination addresses in the packet capture may not belong to the same subnet.	true or false
Modification Options.Original Port to be rewritten as New Port	Rewrite source and destination ports that match Original Port to New Port. This option can only be used when Replay capture file without modification is set to false. A value or 0 for Original Port or New Port disables this feature.	0 – 65535
Modification Options.New Port replacing Original Port	Rewrite source and destination ports that match Original Port to New Port. This option can only be used when Replay capture file without modification is set to false. A value or 0 for Original Port or New Port disables this feature.	0 – 65535
Modification Options.BPF filter string	When set, only matching packets will be played back. This option can only be used when Replay capture file without modification is set to true.	tcpdump-style BPF filter string
Modification Options.Number of Times to Loop Capture File	The number of times to replay capture file without modification.	1 – 10000

Parameter	Description	Valid Values
Modification Options.Start Packet	Specify the starting packet number for playback.	
Modification Options.End Packet	Specify the ending packet for playback.	
General Behavior	Determines whether the Recreate test component uses the data in the capture file or the parameters defined for the component.	**Use capture file settings** – Uses the settings within the capture file to recreate traffic. This option will force Recreate to use the data rate, maximum simultaneous Super Flows, sessions per second, test duration, inter-packet delays, application payload(s), and destination ports from the PCAP file. The source ports are randomized and all other settings will be derived from the Parameters tab and Network Neighborhood. **Use user-specified settings** – Uses the Recreate parameters options to recreate traffic. This option will only use the application payload and destination ports from the PCAP file. The source ports and all other parameters and fields will be derived from the Parameters tab and the Network Neighborhood. This option will force Recreate to ignore any inter-packet delays in the PCAP file and set these delays to '0'.
Delay Start	Establishes the amount of time before a test component begins.	hh:mm:ss
Capture File	Sets the capture file that will be used to recreate traffic.	Available capture file.

Templates

The BreakingPoint Storm offers several default templates for each test component. These have been pre-configured by BreakingPoint Systems; they cannot be modified or deleted; however, you can modify them and save them as new templates.

Templates contain predefined component configurations that can be reused in other tests. All parameter definitions, interface selections, and descriptions will be stored in the template. Once you save a component as a template, it will be listed under the Load a Template section of the component type on which it is based. These templates can be treated like any other test component.

Templates are particularly useful if you are using the Tcl Interface to test a device. You can create templates that contain the necessary test configuration parameters built into them. Therefore, when you reference the template from the Tcl interface, you will not need to configure any of the parameters for the component.

Saving Templates

Before saving a template, make sure that you have configured the test component to your specifications. To save the test component, you must be on the Test Workspace page, and the test component you want to save must be listed in the Test Components section.

To save a custom component:

- Right-click on the test component you would like to save as a template.
- Select **Clone Component** from the list of options that display.
- Enter a name for the Template in the **Save As** field.
- Click the **Save** button.

Editing a Template

If you make changes to a template after you have saved it, you must save it as a new template. If you want to reuse the same template name, you must first delete the template from the system. After you have removed the template from the system, you will need to recreate the component and re-save it as a template.

Maximum Simultaneous Active Flows Details

The **Maximum Simultaneous Active Flows** parameter limits the number of TCP active flows. Active TCP flows are defined as either opening or established but have not yet closed. This is almost identical to the number of simultaneous flows. The only time they are different is when you are using the TCP **Aging Time** parameter. By default aging time is 0, which means TCP connections skip the TIME_WAIT state and go straight to closed and then get deleted from memory. If aging time is set to a value > 0, then the TCP connection will sit in TIME_WAIT for the specified amount of time before it gets deleted.

For example:

1. TCP SYN (curr_flows +=1), (active_flows +=1)
2. TCP EST
3. TCP FIN
4. TIME_WAIT (active_flows -=1)
5. (wait for x seconds)
6. CLOSE (curr_flows -=1)

In summary, **Maximum Simultaneous Active Flows** limits the number of TCP sessions that are active.

CHAPTER 18 Testing

This section covers:

Tests Overview

BreakingPoint offers three methods for testing:

- **Quick Tests**: A test based on a single test component that is pre-configured to test industry standard metrics. For more information on Quick Tests, see the Quick Tests belowsection.
- **Tests**: A user-created test configuration made up of one or more test components. For more information on Tests, see the Creating a New Test on page 816 section.
- **Test Series**: A series of one or more tests that execute sequentially. For more information on Test Series, see the Test Series on page 879 section.

Quick Tests

Quick Tests provide you with a quick snapshot of how well a device performs under standard testing metrics. These pre-configured tests can be run without any modifications.

Six Quick Tests are available from the Menu bar:

- **Bit Blaster**: The Bit Blaster Quick Test measures the raw throughput capacity of the device under test. It is comprised of a series of Bit Blaster tests and takes about 30 minutes to run.
- **Routing Robot**: The Routing Robot Quick Test measures a device's ability to route IP packets correctly by sending data out through various interfaces and verifying that the expected interface receives the data.

- **Session Sender**: The Session Sender Quick Test measures the capacity of the device to handle the maximum number of concurrent sessions.
- **Security**: The Security Quick Test uses Security Level 1, which targets high-risk vulnerabilities in services that are often exposed to the Internet.
- **Stack Scrambler**: The Stack Scrambler Quick Test measures a device's ability to handle invalid IP, TCP, UDP, ICMP, and Ethernet packets by fuzzing the protocols and sending the resulting traffic to the device.
- **Application Simulator**: The Application Simulator Quick Test measures the device's ability to handle a realistic mix of application layer traffic flows.

Note: There are several more pre-configured tests available; however, they are all security-based tests.

Running a Quick Test

Each Quick Test is based on a test component and uses a set of pre-defined parameters. You can access Quick Tests by selecting **Test > Quick Test** from the Menu bar. Quick Tests can be run using their default settings, or you can edit a Quick Test to better fit your needs. For more information on editing a Quick Test, see the section Editing a Quick Test below.

To run a Quick Test:

1. Select **Test > Quick Test** from the Menu bar.
2. Select a test component from the **Quick Test** list.
3. Do one of the following when the popup window is displayed:
4. Click the **Run** button if the desired Device Under Test profile and Network Neighborhood are selected.
5. Click the **Browse** button to select another Device Under Test profile and/or Network Neighborhood. Change your selections for the Device Under Test and/or the Network Neighborhood. When you are done making changes, click the **Run** button to run the Quick Test.
6. Click the **X** to exit the Quick Test screen.

Editing a Quick Test

To customize a Quick Test or view its configuration, you will need to select **Test > OpenTest** from the Menu bar, and select the test you want from the list. The following table lists the actual test names of each Quick Test. You can use these names to locate the test from the Test Browse screen.

Quick Tests

Quick Test	Test Name
Bit Blaster	BitBlasterComplete
Routing Robot	RoutingRobot
Session Sender	SessionSender

Quick Test	Test Name
Security	SecurityTestBasic
Stack Scrambler	StackScramblerStd
Application Simulator	AppSim

Additionally, after you run or cancel a Quick Test, you can edit the test from the Real-Time Statistics window by clicking the **Edit** button. The Quick Test will open, and you can edit any of its settings. Once you are done editing the test, you must save the test with a new name. You cannot overwrite any of the settings configured for a Quick Test.

Test Workspace Page

The Test Workspace page consists of the following sections:

- Network Neighborhood – Allows you to add a Network Neighborhood to your test. You can browse and select from a list of Network Neighborhoods that already exist. You can also create and edit your own Network Neighborhood to use in your test.
- Test Components – Allows you to add a Test Component to your test.
- Test Criteria – Allows you to define the test criteria to your test. to be used in your test.
- Device Under Test – Allows you to select and edit the device being tested.
- Shared Component Settings – Aggregates and displays the following settings for all of the components being used in your test:
- Maximum Flow Creation Rate
- Total Bandwidth
- Maximum Concurrent Flows
- Total Attacks
- Total Addresses

You can adjust these settings in your test by changing the value of the Percent Change field. After you have adjusted the setting, you must click the set button for the change to be accepted. Click the reset button to restore the original settings.

Summary Information – Provides a test summary including the test name and description, along with total values for unique Super Flows, unique Strikes, MAC addresses, subnets, and required MTU. You can also enter a value for the Seed Override.

Creating a New Test

The Test Workspace page allows you to define all the elements of your test.

To create a test:

1. Select **Test > New Test** from the Menu bar. You can also access the Test Workspace page by clicking on the **Create a Test** button on the BreakingPoint Home page.

2. Click the **Browse** available network neighborhoods icon in the **Network Neighborhood** section.
3. Select a Network Neighborhood from the list.
4. If you do not see the Network Neighborhood you want to use, enter a portion of the name into the **Browse Network** field.
5. Click **Select**. The name of the Network Neighborhood you selected will be displayed in the Network Neighborhood section. Click the **Edit Network Neighborhood** button to edit the Network Neighborhood.
6. Click the **Add New** button in the Test Components section.
7. Select a component from the list and click the **Select** button.
8. Enter a name and description for the component. (Optional)
9. Select the **Use Template** check box to select a pre-configured test component. (Optional)
10. Click the **Create** button. The name of the component you selected will be displayed in the Test Components section.
11. Click the **Edit Component** icon and do the following:
    a. Select the **Include in Report** check box to include the statistics from the test in the report. Deselect the check box to disable the detailed section of the report for that component.
    b. Enter a name for the test component in the **Component Name** field. (Optional)
    c. Select **Active** from the **State** dropdown box to enable the test component for the test or select **Inactive** from the State drop-down box to disable the test component for the test.
    d. Enter a customized description of the test component in the **Description** field. There is a 500 character limit. (Optional)
    e. Select client and server tags for the component from the **Component Tags** section.
    f. Click **Load Profile** to select a load profile for the test.
    g. Load a template or modify the parameters in the Parameters section of the test component. For more information on test component parameters, see the Test Components Overview on page 663 section.
    h. **Note**: Edit the Evasion Profile settings from the Parameters section. (Optional, for Security component only.) For more information on Evasion Profile settings, see Evasion Profile Settings on page 233.
    i. **Note**: Edit the Concurrent Strikes settings from the Parameters section. (Optional, for Security component only.) The Concurrent Strikes parameter allows you to choose between Single Strike and Default modes. Single Strike mode runs only one strike at a time, while Default mode runs up to five strikes simultaneously.
12. Click the **Return to Test Work**space button.
13. Click the **Edit Test Criteria** icon in the Test Criteria section and create the pass/fail criteria for the test.
14. Click the **Browse Available DUTs** icon in the **Device Under Test** section.
15. Select a DUT from the **Browse DUT** list. If you do not see the DUT you want to use, enter a portion of the name into the **Browse DUT** field.
16. Click **Select**. The name of the DUT you selected will be displayed in the Device Under Test section.

17. Click the **Edit DUT** icon link to modify the DUT Profile. Once you have made your changes, click the **Return** button to return to the Test Workspace screen. For more information on DUT Profiles, see Task 3: Creating a Device Under Test Profile on page 54.
18. Enter a value in the Seed Override field. (Optional)

    Note: Use the Seed Override to modify the seed for Security, Application Simulator, and Stack Scrambler tests. The Seed Override enables you to control whether static or dynamic content will be generated. If you explicitly set the seed, the system will recreate the same application flows each time the Super Flow is run. If you do not explicitly set a seed, the system will automatically randomize a seed for the Super Flow each time it is used.
19. Click the **Save As** button.
20. Enter a name for the test in the **Name** field.
21. Click the **Save and Run** button to run the test.

After clicking **Save and Run**, the test will run and the Real-Time Statistics window will display. For more information on Real-Time Statistics, see Real-Time Statistics on page 865.

When the test finishes, a popup window will display which indicates whether the test passed or failed. Click the **OK** button to close the window. On the Real-Time Statistics screen, you can choose to restart the test, view the results for the test, or edit the test.

Running a Test

Before running any tests, verify that the DUT Profile, Network Neighborhood, and component parameters have been defined to your specifications.

To run a test:

1. Select **Test** > **Open Test** from the Menu bar.
2. Select a test from the list of tests.
3. You can sort the tests by clicking on any of the column headings (Name, Author, Last Run By, etc.) and scroll through the pages by clicking on the page numbers.
4. Click the **Open** button.
5. Click **Save and Run** to run the test.

Note: For GTP-based tests, it may take up to 20 minutes to close all of the tunnels used in the test. The user interface will be unavailable during this time. Once the test is complete and the system closes all of the tunnels used in the test, the system will become available.

Running a Recently Run Test

You can select **Test** > **Run Recent** from the Menu bar to view a list of up to 9 recently run tests. Select any of test from the list to run it.

To run a Quick Test:

1. Select **Test > Run Recent** from the Menu bar.
2. Select a test from the recently run list.

3. Do one of the following when the popup window displays:
   a. Click the **Run** button if the desired Device Under Test profile and Network Neighborhood are selected. This step will start the test.
   b. Click the **Browse** button to select another Device Under Test profile and/or Network Neighborhood. Change your selections for the Device Under Test and/or the Network Neighborhood. Click the **Run** button when you are done making changes.
   a. Click the cancel (**X**) button to exit the window.

Exporting a Test

When a test is exported, the test components and the component parameters are saved in a .bpt file, which can be e-mailed or placed in a central location where other system users can access it to import into their systems.

When you export a test, you have the option of protecting it by encrypting it and requiring a license to run it. To run a protected test, the user importing it must have the required license in place on the system where the test is imported. Otherwise, it cannot be run.

Note: Tests created with a newer version of BreakingPoint will not work on older versions; however, tests created with an older version of the system will migrate to a newer version of the system.

To export a test:

1. Select **Test > Open Test** from the BreakingPoint Control Center Menu bar.
2. Select the test to be exported. Enter a portion of the name of the test in the **Filtered Search** field if the test is not displayed in the list.
3. Click the **Export Test** button.
4. If you want to protect the test, click **Encrypt**, then specify a license and the password for the license. To run the test, another user will have to have the license.
5. If you want to include any associated files used by the test (such as capture files), click **AddAttachments**.
6. Click **Export**. The test is exported to the folder where the browser stores downloaded files.

Importing a Test

BreakingPoint enables you to import tests created on one BreakingPoint System to another. When a test is imported, the test components and their configurations are stored in the system and available to you from the Tests screen. You can open, configure, and run the test – just like any other test stored in the system.

In addition to importing the components and their configurations, the import test tool will import any DUT Profile, Network Neighborhood, PCAP file, Application Profile, Super Flows, and Strike List the system may need to run the test. Existing items that share the same name as the imported items will be overwritten. The system will not provide a warning when this occurs.

If you have the **Allow Overwrite** option enabled, you can assign the imported test the same name as an existing test in the system. The system will overwrite the existing test with the data from the

imported test. If this option is not enabled and you try to assign the imported test the same name as one that exists in the system, the system will alert you that a test of the same name already exists.

If you are importing an encrypted test, you must have the license for it in order to run it. Even if you have the license for an encrypted test, you cannot view or change its configuration - you can only run it. To run an encrypted test, select it in the Test Browse panel and select Run.

To import a test:

1. Select **Test > Import Test** from the Menu bar. The import test screen will open.
2. Enter a name for the test in the **Test Name** field.
3. Click the **Browse** button.
4. Navigate to the location of the test (.bpt file).
5. Select the **Allow Overwrite** check box if you want to overwrite an existing test with the same name.
6. Click the **Upload** button.

Opening a Test

There are two ways to open an existing test: either by using the Open Test feature or the Open Recent Tests feature from the Menu bar. If you select **Test > Open Test** from the Menu bar, the system will display a list of all the available tests on the system – this includes all default and user-created tests. The system will categorize the tests by name, author, interfaces used, last date run, bandwidth, and test result. You can click on any of the column headings to sort the tests.

If you select **Test > Open Recent Tests**, the system will display a list of up to 9 of the most recently saved tests. You can select any test from this list to open it.

Deleting a Test

Deleting a test will remove it completely from the system and from any test series that uses it.

To delete a test:

1. Select **Test > Open Test** from the Menu bar.
2. Select the test you want to delete from the list of tests.
3. Click the **Delete Test** button.
4. Click **Delete** when the confirmation window displays.

Test Status

Before running a test, you should verify that your test has not exceeded the available bandwidth limitations and hardware resources. On the test screen, there is a link called Test Status. Next to it, you will see an icon that automatically updates with the test's status each time modify the test. If the icon is green, then the test is ready to run; however, if the icon is yellow, there is an issue with the test configuration. If this happens, you will need to click on the **Test Status** link to view the issues with the test configuration.

The system authenticates the test by:

- Validating the addressing information provided to the system from the Network Neighborhood.
- Ensuring that the component parameters use valid values.
- Checking to see if the total bandwidth used across all test interfaces is supported by BreakingPoint.
- Verifying the test interfaces being used are connected to a device under test.
- Verifying the parameters for the test work with the speed of the connection.

Bandwidth Limitations

Bandwidth limitations across all test interfaces depend on the link speed that is available for the DUT. The availability of bandwidth resources depend on the data rate and the subset of hardware resources being used by test components. These factors are used to determine whether or not there is enough bandwidth to execute the test.

If the bandwidth for a test interface is oversubscribed, or using more bandwidth than there is available, check the data rate for each test component on that interface. The sum of the data rates for all test components on the interface should not exceed the bandwidth that is available.

To determine how much bandwidth is available for an interface, click the **Test Status** link. You will see all four test interfaces listed on the screen. Each interface will list its maximum bandwidth capacity. Under each interface, you will see which components are utilizing the interface's resources.

Test Pass/Fail Criteria

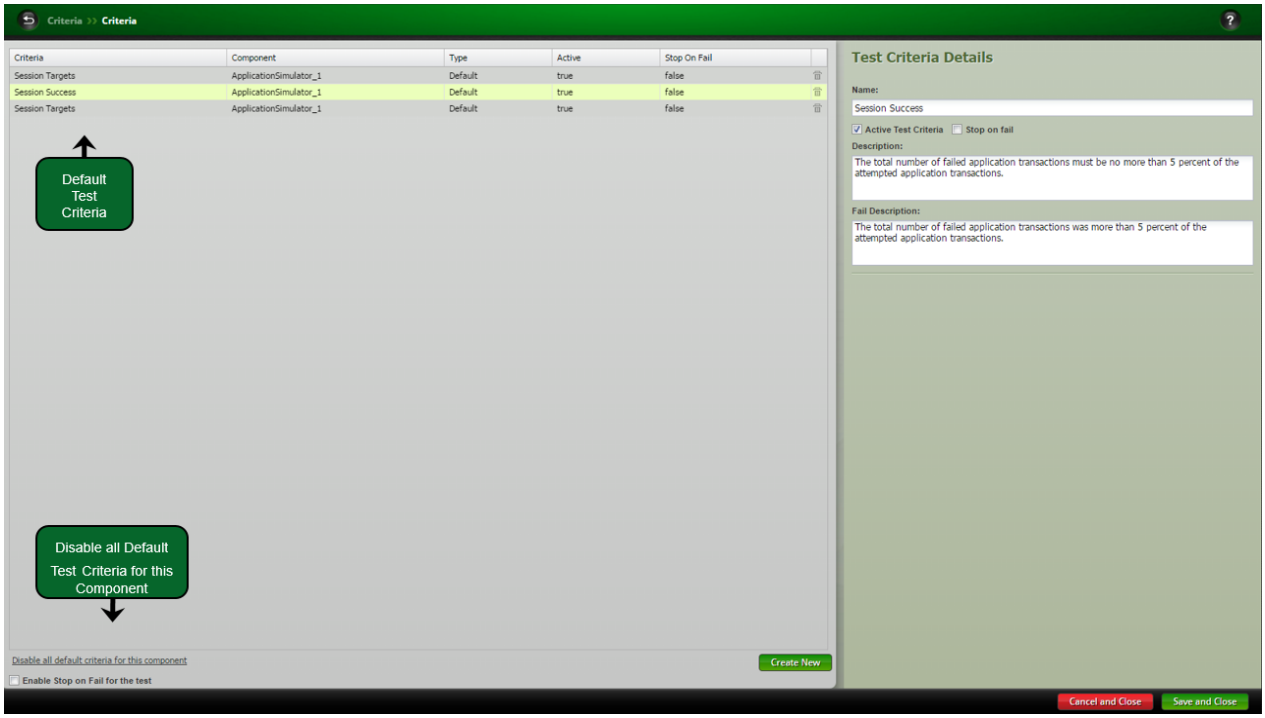
Each test has a set of pass/fail criteria. Each test criteria is a boolean expression that is based on two comparators and an operator (e.g., equals or not equals). The comparators can be selected from a list of stats that are provided by the system, or you can create your own. All criteria have to be met in order for the DUT to pass the test.

If the default pass/fail criteria do not meet your testing needs, you can create your own set of pass/fail conditions.

Default Test Criteria

Each component comes with a set of default test criteria; the system will use these metrics to determine whether the test passes or fails. You can see which test criteria are included with the test component from the test criteria screen. These are denoted as 'Default' in the **Type** field. See Default Test Criteria below.

Default Test Criteria



If you do not want to use the default test criteria for a test component, you can select any of the default criteria for the component and click the **Disable all default criteria for this component** link. This will disable all default test criteria for that particular component.

How the Test Criteria Works

The test will fail if it does not meet the defined test criteria. When test failure occurs, the system will log the failure in the report using the failure description you have provided for the criteria.

On the Test Criteria screen, there is an option called **Active Test Criteria**, which only appears for user-created test criteria. This option determines whether or not the test criteria will be used. This option is useful if you want to create a boolean expression using the results of two pass/fail criterion.

For example, you can create a test criterion called Frames that states: If the number of received frames equals the number of sent frames, then the test passes; and another criterion called Corrupt Frames that states: If the number of corrupted frames equals 0, then the test passes.

Once you have created these two criteria, you can disable the **Active Test Criteria** option for both criteria. This will ensure that the criteria will not be used as individual pass/fail metrics in the test. Instead, you can now create a new criterion that combines both criteria. For example, you can create an expression that states: If Frames and Corrupt Frames are true, then the test passes; or an expression that states: If Frames or Corrupt Frames are true, the test passes.

To create pass/fail test criteria:

1. Select **Test > New Test** from the Menu bar to create a new test or **Test** > **Open Test** from the Menu bar to open an existing test. If you are opening an existing test, skip to Step 8.

2. Click the browse available network neighborhoods icon in the **Network Neighborhood** section. If you are using the default Network Neighborhood, skip this step.
3. Click **Add New** the **Test Components** section.
4. Select a test component to add to the test and click Select.
5. Enter a name for the component in the Add a Component field and click **Create**.
6. Click the browse available **DUTs** icon in the **Device Under Test** section. If you are using the default DUT Profile, skip this step.
7. Repeat steps 3-6 to add additional components to the test.
8. Click the edit test criteria icon in the Test Criteria section.
9. Select the **Enable stop on fail for this test** option to use Stop on Fail as a test criteria for this test (optional).
10. Enter a name for the criterion in the **Name** field.
11. The name can consist of alphanumeric characters, spaces, -, and /.
12. Enter a description for the criteria in the **Description** field.
13. This information will display in the report under the Test Component Criteria section.
14. Enter a description for the test if it fails in the Fail **Description** field.
15. This information will display in the report under the Test Component Criteria section.
16. Select the statistic to be gathered from the Statistic field.
17. Select an operator from the **Operator** field.
18. Click **Create Criteria**.

Comparators

The next few sections will provide you with descriptions of all the comparators that are available in the system. These comparators are based on stats that the system collects while it is running the test.

Each statistic listed in the Comparator list will be tagged with the component's name for which it is associated. For example, if you have an component named Bit Blaster 23, then all stats for that component will be tagged Bit Blaster 23.statName.

You can define a test component's pass/fail criteria by using these stats to create boolean expressions. If the system finds that the boolean expression is true, then the test will pass; if the expression is false, then the test will fail.

When determining the value the system will use for the stat, the system will use the highest value for the comparator found in the test results. For example, let's say you create a criterion that states: the **Frame data transmit rate** must equal 900 Mbps for the test to pass. If the highest data rate that the test ever reaches is 700, then the system will use 700 as the stat's value; therefore, in this case, based on the criterion you created, the test would fail.

If a test fails, the system will log the test criteria that were not satisfied in the test results window that displays when the test completes. It will also be listed in the **Test Synopsis** area of the report as the reason for the test failure.

Stats for Bit Blaster

The following table lists the stats that are available for the Bit Blaster test component.

Bit Blaster Stats

Stat	Description
Gateway ARP Response	The ARP response sent from the gateway
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frame byte transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component
Frame bytes received	The total number of bytes received by the component

Stat	Description
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
Cumulative latency of all frames	The total latency for all frames transmitted and received by the component

Stat	Description
Corrupted frames received	The total numbers of frames that were received by the component; this stat only includes tracks frames that encountered a CRC error.
Out-of-sequence frames received	The total number of Out-of-Sequence frames received by the component
Frames not received on the correct port	The total number of frames that were not received on the correct port
Frames received more than once	The total number of duplicate frames
Frames received that were not test-generated	The total number of frames received by the component that did not come from the system
Slow start frames sent	The total number of slow start frames sent by the component
Dropped frames	The total number of frames dropped by the DUT
Frames received with bad IP checksum	The total number of frames received by the system that had bad IP checksums
Frames received with bad UDP checksum	The total number of frames received by the system that had bad UDP checksums
Slow start frames received	The total number of slow start frames received by the component

Stat	Description
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted by the component
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted by the component
Maximum frame receive rate	The maximum rate (in fps) at which frames are received by the component
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received by the component

Stats for Routing Robot

The following table lists the stats for the Routing Robot test component.

Routing Robot Stats

Stat	Description
Gateway ARP Response	The ARP response sent from the gateway.
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frame byte transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component
Frame bytes received	The total number of bytes received by the component

Stat	Description
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
Cumulative latency of all frames	The total latency for all frames received and transmitted by the component

Stat	Description
Corrupted frames received	The total numbers of frames that were received by the component; this stat only includes tracks frames that encountered a CRC error.
Out-of-sequence frames received	The total number of Out-of-Sequence frames received by the component
Frames not received on the correct port	The total number of frames that were not received on the correct port
Frames received more than once	The total number of duplicate frames
Frames received that were not test-generated	The total number of frames received by the component that did not come from the component
Slow start frames sent	The total number of slow start frames sent by the component
Slow start frames received	The total number of slow start frames received by the component
Dropped frames	The total number of frames dropped by the DUT
Frames received with bad IP checksum	The total number of frames received by the component that had an altered IP checksum
Frames received with bad UDP checksum	The total number of frames received by the component that had an altered UDP checksum

Stat	Description
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted by the component
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted by the component
Maximum frame receive rate	The maximum rate (in fps) at which frames are received by the component
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received by the component

Stats for Session Sender

The following table lists the stats for the Session Sender test component.

Session Sender Stats

Stat	Description
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frame bytes transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component
Frame bytes received	The total number of bytes received by the component
Dropped frames	The total number of frames received by the component but were dropped because they were malformed or misrouted
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes

Stat	Description
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
IP frames transmitted	The total number of IP frames transmitted by the component
IP frame bytes transmitted	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (IP frames only)
IP frames received	The total number of IP frames received by the component
IP frame bytes received	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (IP frames only)

Stat	Description
TCP frames transmitted	The total number of TCP frames transmitted by the component
TCP frame bytes transmitted	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
TCP frames received	The total number of TCP frames received by the component
TCP frame bytes received	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
TCP setup taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
TCP setup taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection
TCP setup taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
TCP setup taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
TCP setup taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection
TCP close taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
TCP close taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
TCP close taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK

Stat	Description
TCP close taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
TCP close taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
Duration 0 - 10 ms	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state
Duration 11 - 100 ms	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
Duration 101 - 1000 ms	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
Duration 1001 - 10000 ms	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
Duration greater than 10000 ms	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
Client established	The total number of TCP connections established by the client
Client closed	The total number of TCP connections closed by the client
Client closed by reset	The total number of times that the server sent a TCP Reset (RST) which caused the session to be disconnected
Client received RST	The total number of TCP Resets sent by the server
Client concurrent	The total number of TCP connections concurrently opened by the client
Client attempted	The total number of TCP connections attempted by the client
Server established	The total number of TCP connections established by the server
Server closed	The total number of TCP connections closed by the server

Stat	Description
Server concurrent	The total number of TCP connections closed by the server
Server closed by reset	The total number of times that the client sent a TCP Reset (RST) which caused the session to be disconnected
Server received RST	The total number of TCP Resets sent by the client
Client State "LISTEN"	The total number of TCP connections on the client's side that were in the LISTEN state
Client State "SYN_SENT"	The total number of TCP connections on the client's side that were in the SYN-SENT state
Client State "SYN_ RECEIVED"	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
Client State "ESTABLISHED"	The total number of TCP connections on the client's side that were in the ESTABLISHED state
Client State "CLOSE_WAIT"	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
Client State "FIN_WAIT_1"	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state
Client State "CLOSING"	The total number of TCP connections on the client's side that were in the CLOSING state
Client State "LAST_ACK"	The total number of TCP connections on the client's side that were in the LAST_ ACK state
Client State "FIN_WAIT_2"	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
Client State "TIME_WAIT"	The total number of TCP connections on the client's side that were in the TIME-WAIT state
Server State "LISTEN"	The total number of TCP connections on the server's side that were in the LISTEN state
Server State "SYN_SENT"	The total number of TCP connections on the server's side that were in the SYN-SENT state

Stat	Description
Server State "SYN_ RECEIVED"	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
Server State "ESTABLISHED"	The total number of TCP connections on the server's side that were in the ESTABLISHED state
Server State "FIN_WAIT_1"	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
Server State "CLOSING"	The total number of TCP connections on the server's side that were in the CLOSING state
Server State "LAST_ACK"	The total number of TCP connections on the server's side that were in the LAST-ACK state
Server State "FIN_WAIT_2"	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
Server State "TIME_WAIT"	The total number of TCP connections on the server's side that were in the TIME-WAIT state
Maximum client concurrent	The maximum number of TCP sessions concurrently opened by the client
Maximum server concurrent	The maximum number of TCP sessions concurrently opened by the server
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component

Stat	Description
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
IP frame transmit rate	The rate (in fps) at which IP frames are transmitted by the component
IP data transmit rate	The rate (in Mbps) at which IP data is transmitted by the component
IP frame receive rate	The rate (in fps) at which IP frames are received by the component
IP data receive rate	The rate (in fps) at which IP frames are received by the component
TCP frame transmit rate	The rate (in fps) at which TCP frames are transmitted by the component
TCP data transmit rate	The rate (in Mbps) at which TCP data is transmitted by the component
TCP frame receive rate	The rate (in fps) at which TCP frames are received by the component
TCP data receive rate	The rate (in Mbps) at which TCP data is received by the component
Average TCP setup time	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average TCP response	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.

Stat	Description
Average TCP time to close	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average duration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Client establish rate	The rate at which TCP sessions are established by the client
Client close rate	The rate at which TCP sessions are closed by the client
Client attempt rate	The rate at which TCP connections are attempted by the client
Server establish rate	The rate at which TCP sessions are established by the server
Server close rate	The rate at which TCP sessions are closed by the server
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted by the component
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted by the component
Maximum frame receive rate	The maximum rate (in fps) at which frames are received by the component
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received by the component
Maximum client establish rate	The maximum rate at which the client establishes TCP connections

Stats for Security

lists the stats for the Security test component.

Security Stats

Stat	Description
Source Gateway ARP Response	The ARP response sent from the source gateway
Destination Gateway ARP Response	The ARP response sent from the destination gateway
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frame byte transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component

Stat	Description
Frame bytes received	The total number of bytes received by the component
Dropped frames	The total number of frames received by the component but were dropped because they were malformed or misrouted
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds

Stat	Description
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
Cumulative latency of all frames	The total latency for all frames transmitted and received by the component
Strikes Passed	The total number of Strikes that were not blocked by the DUT
Total Strikes allowed	The total number of Strikes not blocked by the DUT
Total Strikes blocked	The total number of Strikes blocked by the DUT
Total Strike count	The total number of Strikes sent to the DUT
Blocked Strike count	The total number of Strikes blocked by the DUT
Strike Error Count	The total number of Strikes that encountered an error
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component

Stat	Description
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted
Maximum frame receive rate	The maximum rate (in fps) at which frames are received
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received

Stats for Stack Scrambler

The following table lists the stats for the Stack Scrambler test component.

Stack Scrambler Stats

Stat	Description
Source Gateway ARP Response	The ARP response sent from the source gateway

Stat	Description
Destination Gateway ARP Response	The ARP response sent from the destination gateway
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, ICMP, application, and non-system generated traffic.
Frame bytes transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, ICMP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component
Frame bytes received	The total number of bytes received by the component

Stat	Description
Dropped frames	The total number of frames received by the component but were dropped because they were malformed or misrouted
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds

Stat	Description
Cumulative latency of all frames	The total latency for all frames transmitted and received by the component
Pings sent	The total number of pings sent by the component
Pings received	The total number of pings received by the component
Final pings sent	The number of pings sent at the end of the test by the component
Final pings received	The number of final pings that were sent that are received by the component
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.

Stats for Application Simulator

The following table lists the stats for the Application Simulator test component.

Application Simulator Stats

Stat	Description
Source Gateway ARP Response	The ARP response sent from the source gateway
Destination Gateway ARP Response	The ARP response sent from the destination gateway
Frames transmitted	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, and non-system generated traffic.
Frame byte transmitted	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component

Stat	Description
Frame bytes received	The total number of bytes received by the component
Dropped frames	The total number of frames received by the component but were dropped because they were malformed or misrouted
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
Cumulative latency of all frames	The total latency for all frames transmitted and received by the component

Stat	Description
TCP frames transmitted	The total number of TCP frames transmitted by the component
TCP frame bytes transmitted	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
TCP frames received	The total number of TCP frames received by the component.
TCP frame bytes received	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
TCP setup taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
TCP setup taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection
TCP setup taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
TCP setup taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
TCP setup taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection
TCP response taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to go from the first SYN to SYN-ACK
TCP response taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to go from the first SYN to SYN-ACK
TCP response taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to go from the first SYN to SYN-ACK

Stat	Description
TCP response taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to go from the first SYN to SYN-ACK
TCP response taking greater than 10000 ms	The number of TCP sessions that took over 10000 ms to go from the first SYN to SYN-ACK
TCP close taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
TCP close taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
TCP close taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
TCP close taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
TCP close taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
Duration 0 - 10 ms	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state
Duration 11 - 100 ms	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
Duration 101 - 1000 ms	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
Duration 1001 - 10000 ms	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
Duration greater than 10000 ms	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
Client established	The total number of TCP connections established by the client

Stat	Description
Client closed	The total number of TCP connections closed by the client
Client closed by reset	The total number of times that the server sent a TCP Reset (RST) which caused the session to be disconnected
Client received RST	The total number of TCP Resets sent by the server
Client concurrent	The total number of TCP connections concurrently opened by the client
Client attempted	The total number of TCP connections attempted by the client
Server established	The total number of TCP connections established by the server
Server closed	The total number of TCP connections closed by the server
Server closed by reset	The total number of times that the client sent a TCP Reset (RST) which caused the session to be disconnected
Server received RST	The total number of TCP Resets sent by the client
Server concurrent	The total number of TCP connections closed by the server
Client State "LISTEN"	The total number of TCP connections on the client's side that were in the LISTEN state
Client State "SYN_SENT"	The total number of TCP connections on the client's side that were in the SYN-SENT state
Client State "SYN_ RECEIVED"	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
Client State "ESTABLISHED"	The total number of TCP connections on the client's side that were in the ESTABLISHED state
Client State "CLOSE_WAIT"	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
Client State "FIN_WAIT_1"	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state

Stat	Description
Client State "CLOSING"	The total number of TCP connections on the client's side that were in the CLOSING state
Client State "LAST_ACK"	The total number of TCP connections on the client's side that were in the LAST_ ACK state
Client State "FIN_WAIT_2"	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
Client State "TIME_WAIT"	The total number of TCP connections on the client's side that were in the TIME-WAIT state
Server State "LISTEN"	The total number of TCP connections on the server's side that were in the LISTEN state
Server State "SYN_SENT"	The total number of TCP connections on the server's side that were in the SYN-SENT state
Server State "SYN_ RECEIVED"	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
Server State "ESTABLISHED"	The total number of TCP connections on the server's side that were in the ESTABLISHED state
Server State "CLOSE_WAIT"	The total number of TCP connections on the server's side that were in the CLOSE-WAIT state
Server State "FIN_WAIT_1"	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
Server State "CLOSING"	The total number of TCP connections on the server's side that were in the CLOSING state
Server State "LAST_ACK"	The total number of TCP connections on the server's side that were in the LAST-ACK state
Server State "FIN_WAIT_2"	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
Server State "TIME_WAIT"	The total number of TCP connections on the server's side that were in the TIME-WAIT state
Application frames transmitted	The aggregate total of frames transmitted by the component for all application protocols

Stat	Description
Application frame bytes transmitted	The aggregate total of bytes transmitted by the component for all application protocols
Application frames received	The aggregate total of frames received by the component for all application protocols
Application frame bytes received	The aggregate total of bytes received by the component for all application protocols
App concurrent flows	The maximum number of flows concurrently open at any given time
Aggregate application flows	The total number of flows opened for all application protocols
Application attempted	The total number of application flows attempted
Application successes	The total number of applications flows that were completed
Application failures	The total number of applications flows that did not complete
Application responses taking 0 - 10 ms	The number of transactions that had a response time that lasted between 0 – 10 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
Application responses taking 11 - 100 ms	The number of transactions that had a response time that lasted between 11 – 100 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
Application responses taking 101 - 1000 ms	The number of transactions that had a response time that lasted between 101 – 1000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
Application responses taking 1001 - 10000 ms	The number of transactions that had a response time that lasted between 1001 – 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.

Stat	Description
Application responses taking greater than 10000 ms	The number of transactions that had a response time that lasted longer than 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
Maximum client concurrent	The maximum number of TCP sessions concurrently opened by the client
Maximum server concurrent	The maximum number of TCP sessions concurrently opened by the server
Concurrent Network Flows Max	The maximum number of concurrent application flows reached by the system
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
TCP frame transmit rate	The rate (in fps) at which TCP frames are transmitted by the component
TCP data transmit rate	The rate (in Mbps) at which TCP data is transmitted by the component

Stat	Description
TCP frame receive rate	The rate (in fps) at which TCP frames are received by the component
TCP data receive rate	The rate (in Mbps) at which TCP data is received by the component
Average TCP setup time	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average TCP response	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average TCP time to close	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average duration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Client establish rate	The rate at which TCP sessions are established by the client
Client close rate	The rate at which TCP sessions are closed by the client
Client attempt rate	The rate at which TCP connections are attempted by the client
Server establish rate	The rate at which TCP sessions are established by the server
Server close rate	The rate at which TCP sessions are closed by the server
Application frame transmit rate	The rate (in fps) at which frames are transmitted by the component; this value only includes application traffic.

Stat	Description
Application data transmit rate	The rate (in Mbps) at which layer 7 traffic is transmitted by the component; this value only includes application traffic.
Application frame receive rate	The rate (in fps) at which frames are received by the component; this value only includes application traffic.
Application data receive rate	The rate (in Mbps) at which layer 7 traffic is received by the component; this value only includes application traffic.
Application flow rate	The number of new application flows that are opened per second
Application attempt rate	The number of new application flows that are attempted by the component per second; this value accounts for all flows that have sent a Transaction Start packet.
Application success rate	The number of application flows that are successfully established per second; this value accounts for all flows that have sent a Transaction End packet.
Application failure rate	The rate at which application flows fail; this value accounts for all flows that have sent a Transaction Start packet, but no Transaction End packet.
Application average response time	The average response time for an application flow; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted by the component
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted by the component
Maximum frame receive rate	The maximum rate (in fps) at which frames are received by the component
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received by the component
Maximum establish client rate	The maximum rate at which the client establishes TCP connections

Stat	Description
Network Flow Rate Max	The maximum rate at which application flows were opened

Stats for Recreate

The following table lists the stats for the Recreate test component.

Recreate Stats

Stat	Description
Frames transmitted	The total number of frames transmitted by the component
Frame byte transmitted	The total number of bytes transmitted by the component
Frames transmitted from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames transmitted from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames transmitted from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames transmitted from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames transmitted from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
Frames received	The total number of frames received by the component
Frame bytes received	The total number of bytes received by the component

Stat	Description
Dropped frames	The total number of frames received by the component but were dropped because they were malformed or misrouted
Frames received from 64 - 127 bytes	The total number of transmitted frames that were between 64 – 127 bytes
Frames received from 128 - 255 bytes	The total number of transmitted frames that were between 128 – 255 bytes
Frames received from 256 - 511 bytes	The total number of transmitted frames that were between 256 – 511 bytes
Frames received from 512 - 1023 bytes	The total number of transmitted frames that were between 512 – 1023 bytes
Frames received from 1024 bytes up	The total number of transmitted frames that were larger than 1024 bytes
0 - 10 us latency	The number of frames that had a latency between 0 – 10 microseconds
11 - 100 us latency	The number of frames that had a latency between 11 – 100 microseconds
101 - 1000 us latency	The number of frames that had a latency between 101 – 1000 microseconds
1001 - 10000 us latency	The number of frames that had a latency between 1001 – 10000 microseconds
over 10000 us latency	The number of frames that had a latency of more than 10000 microseconds
TCP frames transmitted	The total number of TCP frames transmitted by the component
TCP frame bytes transmitted	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)

Stat	Description
TCP frames received	The total number of TCP frames received by the component.
TCP frame bytes received	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
TCP setup taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
TCP setup taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection
TCP setup taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
TCP setup taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
TCP setup taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection
TCP response taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to go from the first SYN to SYN-ACK
TCP response taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to go from the first SYN to SYN-ACK
TCP response taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to go from the first SYN to SYN-ACK
TCP response taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to go from the first SYN to SYN-ACK
TCP response taking greater than 10000 ms	The number of TCP sessions that took over 10000 ms to go from the first SYN to SYN-ACK

Stat	Description
TCP close taking 0 - 10 ms	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
TCP close taking 11 - 100 ms	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
TCP close taking 101 - 1000 ms	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
TCP close taking 1001 - 10000 ms	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
TCP close taking greater than 10000 ms	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
Duration 0 - 10 ms	The number of sessions that had a duration of 0 – 10 ms in the ESTABLISHED state
Duration 11 - 100 ms	The number of sessions that had a duration of 11 – 100 ms in the ESTABLISHED state
Duration 101 - 1000 ms	The number of sessions that had a duration of 101 – 1000 ms in the ESTABLISHED state
Duration 1001 - 10000 ms	The number of sessions that had a duration of 1001 – 10000 ms in the ESTABLISHED state
Duration greater than 10000 ms	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
Client established	The total number of TCP sessions established by the client
Client closed	The total number of TCP sessions closed by the client
Client closed by reset	The total number of times that the server sent a TCP Reset (RST) which caused the session to be disconnected
Client received RST	The total number of TCP Resets sent by the server

Stat	Description
Client concurrent	The total number of TCP sessions concurrently opened by the client
Client attempted	The total number of TCP connections attempted by the client
Server established	The total number of TCP sessions established by the server
Server closed	The total number of TCP sessions closed by the server
Server closed by reset	The total number of times that the client sent a TCP Reset (RST) which caused the session to be disconnected
Server received RST	The total number of TCP Resets sent by the client
Server concurrent	The total number of TCP sessions closed by the server
Client State "LISTEN"	The total number of TCP connections on the client's side that were in the LISTEN state
Client State "SYN_SENT"	The total number of TCP connections on the client's side that were in the SYN-SENT state
Client State "SYN_ RECEIVED"	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
Client State "ESTABLISHED"	The total number of TCP connections on the client's side that were in the ESTABLISHED state
Client State "CLOSE_WAIT"	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
Client State "FIN_WAIT_1"	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state
Client State "CLOSING"	The total number of TCP connections on the client's side that were in the CLOSING state
Client State "LAST_ACK"	The total number of TCP connections on the client's side that were in the LAST_ ACK state
Client State "FIN_WAIT_2"	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state

Stat	Description
Client State "TIME_WAIT"	The total number of TCP connections on the client's side that were in the TIME-WAIT state
Server State "LISTEN"	The total number of TCP connections on the server's side that were in the LISTEN state
Server State "SYN_SENT"	The total number of TCP connections on the server's side that were in the SYN-SENT state
Server State "SYN_ RECEIVED"	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
Server State "ESTABLISHED"	The total number of TCP connections on the server's side that were in the ESTABLISHED state
Server State "FIN_WAIT_1"	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
Server State "CLOSING"	The total number of TCP connections on the server's side that were in the CLOSING state
Server State "LAST_ACK"	The total number of TCP connections on the server's side that were in the LAST-ACK state
Server State "FIN_WAIT_2"	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
Server State "TIME_WAIT"	The total number of TCP connections on the server's side that were in the TIME-WAIT state
Maximum client concurrent	The maximum number of TCP sessions concurrently opened by the client
Maximum server concurrent	The maximum number of TCP sessions concurrently opened by the server
Concurrent network flows max	The maximum number of concurrent application flows reached
Frame transmit rate	The rate at which frames were transmitted (in fps) by the component
Frame data transmit rate	The rate at which data was transmitted (in Mbps) by the component

Stat	Description
Average transmit frame size	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
Frame receive rate	The rate at which frames were received (in fps) by the component
Frame data receive rate	The rate at which data was received (in Mbps) by the component
Average receive frame size	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
Average frame latency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
TCP frame transmit rate	The rate (in fps) at which TCP frames are transmitted by the component
TCP data transmit rate	The rate (in Mbps) at which TCP data is transmitted by the component
TCP frame receive rate	The rate (in fps) at which TCP frames are received by the component
TCP data receive rate	The rate (in Mbps) at which TCP data is received by the component
Average TCP setup time	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average TCP response	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.

Stat	Description
Average TCP time to close	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Average duration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
Client establish rate	The rate at which TCP sessions are established by the client
Client close rate	The rate at which TCP sessions are closed by the client
Client attempt rate	The rate at which TCP connections are attempted by the client
Server establish rate	The rate at which TCP sessions are established by the server
Server close rate	The rate at which TCP sessions are closed by the server
Application frame transmit rate	The rate (in fps) at which frames are transmitted by the component; this value only includes application traffic.
Application data transmit rate	The rate (in Mbps) at which layer 7 traffic is transmitted by the component; this value only includes application traffic.
Application frame receive rate	The rate (in fps) at which frames are received by the component; this value only includes application traffic.
Application data receive rate	The rate (in Mbps) at which layer 7 traffic is received by the component; this value only includes application traffic.
Application flow rate	The number of new application flows that are opened per second

Stat	Description
Application attempt rate	The number of new application flows that are attempted by the component per second; this value accounts for all flows that have sent a Transaction Start packet.
Application success rate	The number of application flows that are successfully established per second; this value accounts for all flows that have sent a Transaction End packet.
Application failure rate	The rate at which application flows fail; this value accounts for all flows that have sent a Transaction Start packet, but no Transaction End packet.
Application average response time	The average response time for an application flow; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is received.
Maximum frame transmit rate	The maximum rate (in fps) at which frames are transmitted by the component
Maximum frame data transmit rate	The maximum rate (in Mbps) at which data is transmitted by the component
Maximum frame receive rate	The maximum rate (in fps) at which frames are received by the component
Maximum frame data receive rate	The maximum rate (in Mbps) at which data is received by the component
Maximum establish client rate	The maximum rate at which the client establishes TCP connections
Network Flow Rate Max	The maximum rate at which application flows were opened

Real-Time Statistics

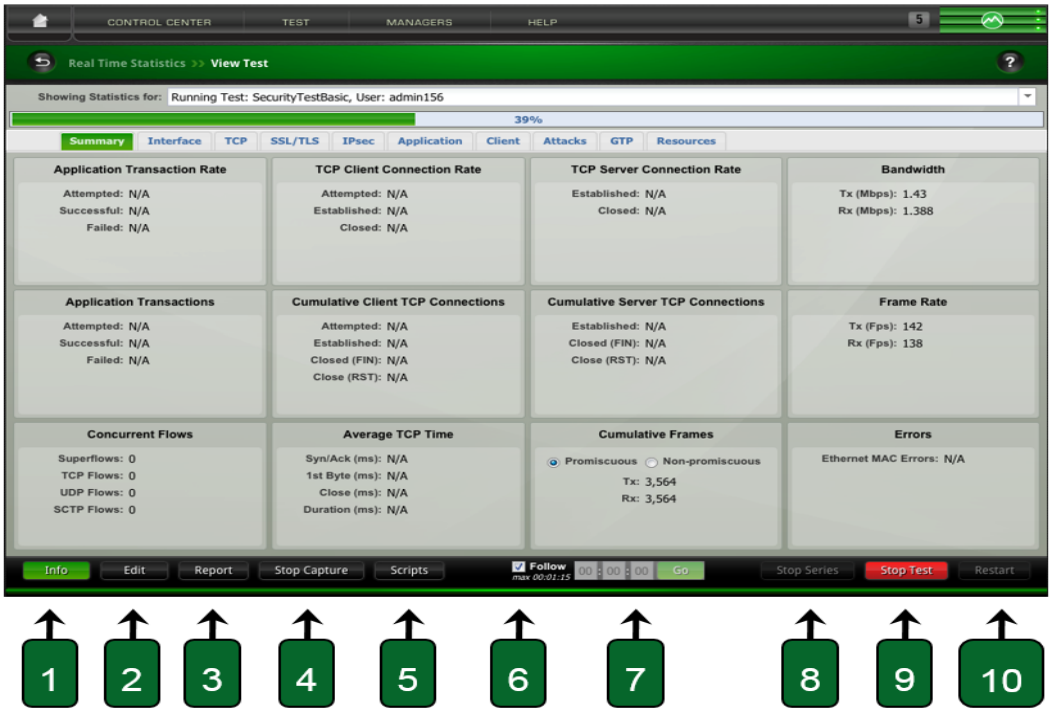
Real-Time Statistics let you instantly see the progress of a running test at any given point in time. This window will be displayed once the test starts. These stats provide the aggregate results for all the test components used in a test. For multi-box tests, these stats show the aggregate results for all the systems used in the test.

The Real-Time Statistics window consists of interactive graphs that instantly update as the test is running. You can control what you see in the Real-Time Statistics window by clicking on any of the tabs at the top of the Real-Time Statistics window.

Note: When the **Show Aggregate Cumulative Frames** option is enabled (the default setting) on the Administration page, all packets that arrive at a port are reported in the statistics regardless as to how the packets reached the port. When the **Show Aggregate Cumulative Frames** option is disabled, only packets that match the configuration of the current network neighborhood are reported in the statistics. See Show Aggregate Cumulative Frames for additional information about the option.

Real-Time Statistics Tabs

Tab	Description
Summary	Displays the aggregate totals for TCP connections, application traffic, data rate, bandwidth, and transmitted/received frames.
Interface	Displays the frame rate and data rate for each interface.
TCP	Displays TCP connection rate for the attempted and successful TCP connections.
SSL/TLS	Displays the rate at which SSL handshakes are started, completed, and aborted as well as the data rate at which they are established.
IPSec	Displays the rate at which encrypted traffic travels over a network.
Application	Displays the number of application flows that were attempted, successful, and unsuccessful; the number of transmitted and received bytes; and the data rate at which flows were transmitted and received.
Client	Displays the logging results of the application layer. The Exceptions section gives you the ability to compare and verify data bit by bit as it is being transferred. The valid/invalid statistics will increment when data validation is enabled in a Super Flow. The Client tab allows you to monitor TCP and Application level statistics.
Attacks	Displays the number of attacks that were blocked, allowed, and errored, as well as the number of pings that were sent and received by Stack Scrambler.
GTP	Displays the number of GTP tunnels sent and received during LTE and 3G testing.
Resources	Displays the memory, processor usage and performance of BreakingPoint allowing you to monitor load in real time.



The image and table below describe the Real-Time Statistics window controls.

Real-Time Statistics Page Description

Callout	Function	Description
1	Test Information	Provides the name, progress, result, and host IP for each test that was run.
2	Edit Test	Enables you to edit the test; this function should only be used once the test has completed.
3	View Report	Displays the report for the test.
4	Stop Capture	Stops the capture from running. Data will be stored from the beginning of the capture until the capture was stopped. Once you have stopped the capture, you can restart it with this feature.
5	On Demand Scripts	Displays On Demand DUT scripts configured for a test any time during a test.
6	Follow	If enabled, this function allows you to follow the live statistics generated by the test. If disabled, you will be able use the Go function to view a specific point in time in the test.

Callout	Function	Description
7	Go	Enables you to jump to a specific time in a test. To use this function, enter the time (in hh:mm:ss) and click the Go button.
8	Stop Series	Stops the test series from running.
9	Stop Test	Stops the test from running.
10	Restart Test	Restart a test or restart from the beginning of a test series (if running a test series).

Navigating the Real-Time Statistics Page

You can hover over any of the points on the line graphs to obtain its exact value. These values represent the aggregate totals for all the test components in the test.

Since the graph will constantly update as the test is running, you can deselect the **Follow** option, and input a time within the test to go to. This enables you to control which points of the test you want to view.

In addition to viewing the values for each statistic, you can control the line graphs that are displayed for each statistic. To do this, simply click on any of the options listed in the legend. Sometimes, graphs will overlap other graphs, making it difficult to see the results; therefore, you may want to reduce the number of line graphs that are displayed so that each graph is more easily visible.

Returning to the Real-Time Statistics Page

After the test has completed the initialization process, you can leave the Real-Time Statistics Page to other tasks. After you have completed your other tasks, you can return to the Real-Time Statistics page.

To return to the Real-Time Statistics screen:

1. Select **Control Center > Device Status** from the Menu bar.
2. Click the P**rogress** icon located above one of the active ports in your test. The Test Running dialog box is displayed.
3. Click **View**. The Real-Time Statistics screen is displayed.

Exceptions

BreakingPoint emits approximately five flow exceptions per second. Approximately 3,000 exceptions can be reported per component. Exceptions are tracked separately for each component within your test. For example, if you have two Application Simulator components in your test, each one tracks its own limit on flow exceptions.

A list of the flow exceptions that could occur includes the following:

- A flow closed when more data was expected
- TCP Reset was received

- An unexpected TCP FIN was received
- There were too many retries on a TCP connection
- A timeout occurred while waiting for a UDP packet
- The received data for a flow was incomplete
- A packet was received for a connection that was already closed
- When doing a token substitution, the data to substitute was too large to fit in the packet, and was truncated
- Destination was unreachable
- A connection was refused
- A gateway was unreachable
- There was a cryptographic error

This list is provided to let you know the basis on which the list of exceptions is filtered; however, it is not a complete list of all of the exceptions. New flow exceptions will be added periodically.

Note: Exceptions are reported relative to when the first datapoint is sent. Therefore, a negative time value indicates that (after the test was initialized) the exception occurred before the first datapoint was sent.

Real-Time Statistics Summary Tab

lists the statistics for the Summary tab of the Real-Time Statistics page.

Summary Tab Statistics

Statistic	Description
Application Transaction Rate: Attempted	Flows that have sent a TRANSACTION_START packet per second, computed as (appAttempted(curr) - appAttempted(last)) / (timestamp(curr) - timestamp (last)).
Application Transaction Rate: Successful	Flows that have sent a TRANSACTION_END packet per second, computed as (appSuccessful(curr) - appSuccessful(last)) / (timestamp(curr) - timestamp (last)).
Application Transaction Rate: Failed	Flows that have sent a TRANSACTION_START packet but no TRANSACTION_ END packet before closing per second, computed as (appUnsuccessful(curr) - appUnsuccessful(last)) / (timestamp(curr) - timestamp(last)).
Application Transactions: Attempted	Increments when a packet in a flow marked as TRANSACTION_START is sent.
Application Transactions: Successful	Increments when a packet in a flow marked as TRANSACTION_END is sent.

Statistic	Description
Application Transactions: Failed	Increments when a flow that has sent a TRANSACTION_START packet is terminated before sending a TRANSACTION_END packet, and the reason is not due to the test ending early.
Concurrent Flows: Superflows	The number of active superflows.
Concurrent Flows: TCP Flows	The number of active TCP flows.
Concurrent Flows: UDP Flows	The number of active UDP flows.
TCP Connection Rate: Client (Attempted)	The rate at which initial SYN packets are sent for new TCP sessions. This does not count retries.
TCP Connection Rate: Client (Established)	The rate at which final ACK packet of the 3-way handshake are sent for TCP sessions.
TCP Connection Rate: Client (Closed)	The rate at which final ACK packet of the 3-way handshake are sent for TCP sessions.
TCP Connection Rate: Server (Established)	Server establish rate.
TCP Connection Rate: Server (Closed)	Server close rate.
Cumulative TCP Connections: Client (Attempted)	Client attempted.
Cumulative TCP Connections: Client (Established)	Client established.

Statistic	Description
Cumulative TCP Connections: Client (Closed (FIN))	Client closed.
Cumulative TCP Connections: Server (Close (RST))	Client closed by reset.
Cumulative TCP Connections: Server (Established)	Server established.
Cumulative TCP Connections: Server (Closed (FIN))	Server closed.
Cumulative TCP Connections: Server (Close (RST))	Server closed by reset.
Average TCP Time (Syn/Ack)	Average time from first SYN to SYN|ACK, only applicable to client-side connections. Computed as tcpResponseTime_total / sum(tcpResponseTime_ [10|100|1000|10000|high]).
Average TCP Time (1st Byte)	Average TCP setup time. The average time from first SYN to ESTABLISHED, only applicable to client-side connections, computed as tcpSetupTime_total / sum (tcpSetupTime_[10|100|1000|10000|high]).
Average TCP Time (Close)	Average TCP time to close. The average time from the first FIN|ACK to the last ACK, only applicable to client-side connections, computed as tcpCloseTime_ total / sum(tcpCloseTime_[10|100|1000|10000|high]).
Average TCP Time (Duration)	Average duration. The average time spent in the ESTABLISHED state, computed as tcpSessionDuration_total / sum(tcpSessionDuration_ [10|100|1000|10000|high]).
Interface Stats: Bandwidth (Tx)	Ethernet transmit rate. This statistic may or may not include Layer 1 data, depending on how the 'Set data rate as L2' option is set.

Statistic	Description
Interface Stats: Bandwidth (Rx)	Ethernet receive rate. This statistic may or may not include Layer 1 data, depending on how the 'Set data rate as L2' option is set.
Interface Stats: Frame Rate (Tx)	Ethernet frame transmit rate.
Interface Stats: Frame Rate (Rx)	Ethernet frame receive rate.
Cumulative Frames: Tx	Ethernet frames transmitted.
Cumulative Frames: Rx	Ethernet frames received.
Ethernet MAC errors	Total errors.

Real-Time Statistics Interface Tab

lists the statistics for the Interface tab of the Real-Time Statistics page.

Interface Tab Statistics

Statistic	Description
Frame Rate (Rx)	Ethernet frame receive rate.
Frame Rate (Tx)	Ethernet frame transmit rate.
Data Rate (Rx)	Ethernet receive rate.
Data Rate (Tx)	Ethernet transmit rate.

Real-Time Statistics TCP Tab

lists the statistics for the TCP tab of the Real-Time Statistics page.

TCP Tab Statistics

Statistic	Description
TCP Connection Rate: Attempted	Client attempt rate. The rate at which initial SYN packets are sent for new TCP sessions. This does not count retries.
TCP Connection Rate: Successful	The client establish rate.

Statistic	Description
TCP State: Syn_Sent (Client)	Client State "SYN_SENT".
TCP State: Syn_Sent (Server)	Server State "SYN_SENT".
TCP State: Syn_ Received (Client)	Client State "SYN_RECEIVED".
TCP State: Syn_ Received (Server)	Server State "SYN_RECEIVED".
TCP State: Established (Client)	Client State "ESTABLISHED".
TCP State: Established (Server)	Server State "ESTABLISHED".
TCP State: Concurrent (Client)	Client concurrent.
TCP State: Concurrent (Server)	Server concurrent.
TCP State: FIN_ WAIT_1 (Client)	Client State "FIN_WAIT_1".
TCP State: FIN_ WAIT_1 (Server)	Server State "FIN_WAIT_1".
TCP State: Closing (Client)	Client State "CLOSING".
TCP State: Closing (Server)	Server State "CLOSING".
TCP State: Close_ Wait (Client)	Client State "CLOSE_WAIT".
TCP State: Close_ Wait (Server)	Server State "CLOSE_WAIT".
TCP State: FIN_ WAIT_2 (Client)	Client State "FIN_WAIT_2".
TCP State: FIN_ WAIT_2 (Server)	Server State "FIN_WAIT_2".

Statistic	Description
TCP State: Close_ Wait (Client)	Client State "CLOSE_WAIT".
TCP State Close_Wait (Server)	Server State "CLOSE_WAIT".
TCP State: TIME_ WAIT (Client)	Client State "TIME_WAIT".
TCP State: TIME_ WAIT (Server)	Server State "TIME_WAIT".
TCP State: LAST_ACK (Client)	Client State "LAST_ACK".
TCP State: LAST_ACK (Server)	Server State "LAST_ACK".
TCP State: Close_ Wait (Server)	Server State "CLOSE_WAIT".

Real-Time Statistics SSL/TLS Tab

The table below lists the statistics for the SSL/TLS Tab of the Real-Time Statistics page.

SSL/TLS Tab Statistics

Statistic	Description
Handshake Rate: Started	Handshakes Started.
Handshake Rate: Finished	Handshakes Finished.
Handshake Rate: Aborted	Handshakes Aborted.
Encrypted Data Rate	Encrypted data transmit rate.
Decrypted Data Rate	Encrypted data receive rate.

Real-Time Statistics IPSec Tab

lists the statistics for the IPSec tab of the Real-Time Statistics page.

IPSec Tab Statistics

Statistic	Description
Request Rate	Displays the rate at which IPsec tunnels are initiated and retried.

Statistic	Description
Connection Rate	Displays the tunnel connection rate for the attempted and successful tunnel connections.
Teardown Rate	Displays the number of tunnels torn down per second.
Tx Rate	The rate at which tunnels were transmitted by the component
Rx Rate	The rate at which tunnels were received by the component

Real-Time Statistics Application Tab

lists the statistics for the Application tab of the Real-Time Statistics page.

Application Tab Statistics

Statistic	Description
Attempted	Application attempt rate. The flows that have sent a TRANSACTION_START packet per second. Computed as (appAttempted(curr) - appAttempted(last)) / (timestamp (curr) - timestamp(last)).
Successful	Application success rate. The flows that have sent a TRANSACTION_END packet per second, computed as (appSuccessful(curr) - appSuccessful(last)) / (timestamp(curr) - timestamp(last)).
Unsuccessful	Application failure rate. Flows that have sent a TRANSACTION_START packet but no TRANSACTION_END packet before closing per second. Computed as (appUnsuccessful(curr) - appUnsuccessful(last)) / (timestamp(curr) - timestamp (last)).
TX Bytes	Application frame bytes transmitted. This includes all packet overhead, including l2, l3, l4 headers, ethernet CRC and inter-packet gap (20 bytes per frame).
RX Bytes	Application frame bytes received. This includes all packet overhead, including l2, l3, l4 headers, ethernet CRC and inter-packet gap (20 bytes per frame).
TX Mbps	Application data transmit rate.
RX Mbps	Application data receive rate.

Real-Time Statistics Client Tab

lists the statistics for the Client tab of the Real-Time Statistics page.

Client Tab Statistics

Statistics	Description
Client: Attempted	Application attempt rate. The flows that have sent a TRANSACTION_START packet per second, computed as (appAttempted(curr) - appAttempted(last)) / (timestamp (curr) - timestamp(last)).
Client: Successful	Application success rate. The flows that have sent a TRANSACTION_END packet per second, computed as (appSuccessful(curr) - appSuccessful(last)) / (timestamp(curr) - timestamp(last)).
Client: Unsuccessful	Application failure rate. The flows that have sent a TRANSACTION_START packet but no TRANSACTION_END packet before closing per second, computed as (appUnsuccessful(curr) - appUnsuccessful(last)) / (timestamp(curr) - timestamp (last)).
Client: Valid	N/A
Client: Invalid	N/A
TCP Connections: Attempted	Client attempted.
TCP Connections: Established	Client established.
TCP Connections: Closed	Client closed.

Real-Time Statistics Attacks Tab

lists the statistics for the Attacks tab of the Real-Time Statistics page.

Attacks Tab Statistics

Statistics	Description
Cumulative Attacks: Blocked	The number of security strikes that have been blocked by the DUT during the test.
Cumulative Attacks: Allowed	The number of security strikes that have been allowed through the DUT during the test.

Statistics	Description
Cumulative Attacks: Errored	The number of security strikes that have experienced an error during the test.
Pings: Sent	The number of diagnostic ping packets sent by Stack Scrambler.
Pings: Received	The number of diagnostic ping packets received by Stack Scrambler. For Stack Scrambler pings, this determines whether the device is dropping data during the test run.

Real-Time Statistics GTP Tab

lists the statistics for the GTP tab of the Real-Time Statistics page.

GTP Tab Statistics

Statistic	Description
Client Side Tunnels	The total number of GTP bearers created on the client side of the GTP session.
Server Side Tunnels	Total number of GTP bearers created on the server side of the GTP session.
Client Side Tunnels Sent	Total number of client GTP bearers that have transmitted payload data.
Client Side Tunnels Received	Total number of client GTP bearers that have received payload data.
Server Side Tunnels Sent	Total number of server GTP bearers that have transmitted payload data.
Server Side Tunnels Received	Total number of server GTP bearers that have received payload data.

Real-Time Statistics Resource Tab

lists the statistics for the Resources tab of the Real-Time Statistics page.

Resources Tab Statistics

Statistic	Description
Processor Usage: System Controller	Current processor resources used by the System Controller.
Processor Usage: Network Processor 1	Current processor resources used by Network Processor 1.
Processor Usage: Network Processor 2	Current processor resources used by Network Processor 2.

Statistic	Description
Memory Usage: System Controller	Current memory used by the System Controller.
Memory Usage: Network Processor 1	Current memory used by Network Processor 1.
Memory Usage: Network Processor 2	Current memory used by Network Processor 2.

Test Interfaces

You can select any combination of interfaces to act as the server and the client. One component can have multiple server interfaces and client interfaces. The configuration you have defined for the test component will be used to generate all network traffic transmitted from any of the client interfaces.

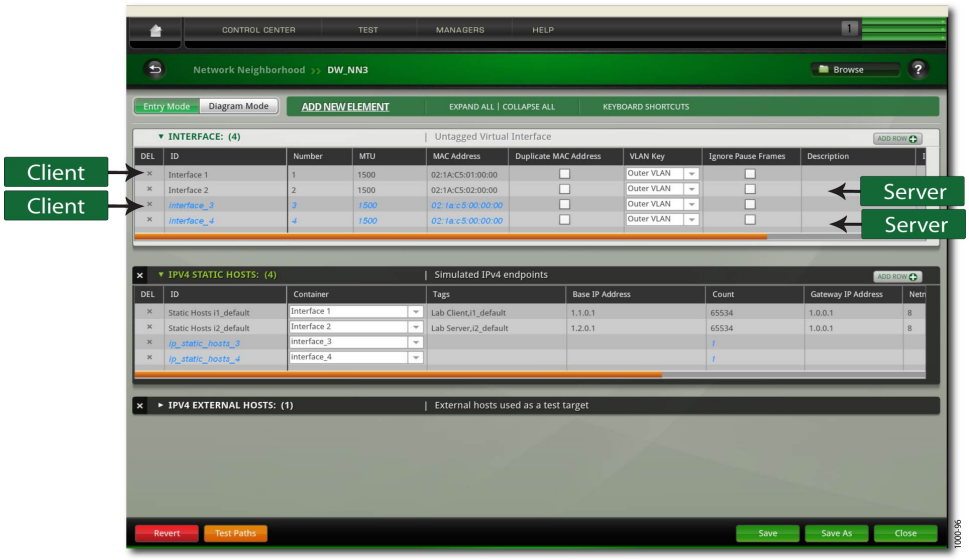
The Bit Blaster and Routing Robot test components can only have one transmitting (client) interface assigned per component; however, you can assign up to 4 receiving (server) interfaces.

Note: Routing Robot supports up to four unique VLAN source and destination VLAN tags. Do not exceed four VLAN source and destination tags when running a Routing Robot test. If you attempt to use more than 4 VLAN tags for a Routing Robot test, you may receive an error message.

Network traffic will be transmitted from the interfaces designated as the client and received on the interfaces designated as the server. The system will randomly select the server/client pairs that it will use if you have multiple client and server interfaces selected. If you want to have control over the client/server pairs that are used by the system, you will need to create a separate component for each client/server pair you want.

For example, if you want a Session Sender component that uses Interface 1 as the client and Interface 2 as the server, but you also want to use Interface 3 as the client and Interface 4 as the server. In this case, you may use two versions of the test component. One Session Sender component will set up Interface 1 as the client and Interface 2 as the server; the second Session Sender component will set up Interface 3 as the client and Interface 4 as the server.

Sample Test Interface Set Up



For this example, you could have used a single Session Sender component, and selected Interface 1 and 3 as the client and Interface 2 and 4 as the server (see on the previous page); however, this would have given you the following client/server pairs: 1/2, 1/4, 3/2, and 3/4. Using individual Session Sender components in this example allowed you to control the interfaces that are used by the test.

Test Series

A test series is a group of one or more tests that are executed sequentially. You can either create your own test series or use one of the pre-configured test series that come with BreakingPoint.These default test series cannot be modified, and the tests associated with default test series will be grayed out.

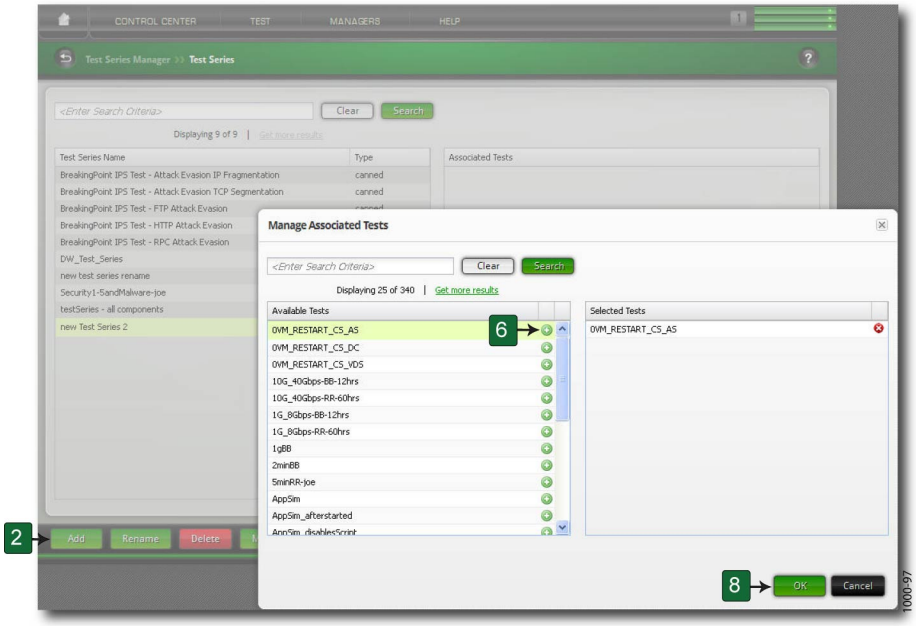
The test series will use the pass/fail criteria set for each test to determine whether or not the test series passes or fails. If one test does not meet its pass/fail criteria, then the test series will fail.

If none of the default test series meet your testing requirements, you can create custom tests series using user-created tests or default tests. For more information on creating test series, see the section Creating a Test Series below.

Creating a Test Series

A test series can be created using default tests and user-created tests. At a minimum, a test series must have at least 1 test associated with it, but it cannot have more than 20 tests.

Creating a Test Series



To create a test series:

1. Select **Test > Test Series** from the Menu bar.
2. Click the Add button.
3. Enter a name for the test series.
4. Click the **Update** button.
5. Click the Manage button. The Manage Associated Tests dialog box is displayed.
6. Click **('+')** button located to the right of the tests to add them to the test series. The tests you select will be displayed in the Selected Tests lists.

   Note: The order in which the tests appear in the Selected Tests list determines the order in which the tests are executed. Be sure to add the tests in the order in which you want them to be executed.
7. Repeat step 6 until all the desired tests have been added.
8. To narrow the list of available tests, enter search criteria into the Search field.
9. Click OK when all the desired tests have been added.

Searching for Test Series

With the Search field of the Test Series Manager page, you can search for individual Test Series based on details such as title, author, and dut. To perform a search, enter one of the items listed in the following table into the Search field on the Test Series Manager page.

contains some of the query strings that can be used to search for specific types of Test Series. Enter these query strings into the search field to narrow your search.

Query Strings For Test Series

Query Type	Description	Example
author	Lists Test Series according to the name of the user that last modified the Test Series.	author:BreakingPoint
class	Lists Test Series according to class.	class:exploit
createdby	Lists Test Series according to the name of the user that created the Test Series.	createdby:BreakingPoint
dut	Lists the type of device under test in the Test Series.	dut:BreakingPoint Default
network	Lists the name of the Network Neighborhood	network:switching
testmodel	Lists the name of a test run in the plan	testmodel:appsim
title	Lists Test Series Names that contain some or all of the criteria specified.	title:Security

Running a Test Series

Running a test series will execute the tests in the order in which they are arranged in the Test(s) list. When a test series is run, the Real-Time Statistics screen will display.

Upon the completion of the test, the system will display a popup window showing the results of the test series. The results of the test are determined by using the success criteria established for each test. If one test in the test series fails, then the test series will fail.

To run a test series:

1. Select **Test > Test Series** from the Menu bar.
2. Select a test series from the **Test Series** list.
3. Click the **Run** button.

This page intentionally left blank.

CHAPTER 19 Test Labs

Quick Test - RFC 2544 General Information

The RFC 2544 defines a number of tests that can be used to measure the performance and determine the behavior of network devices. As the RFC 2544 states, "Not all the tests apply to all types of devices"; thus, you should only utilize the tests that are relevant to your device under test (DUT).

This release of the RFC 2544 is specifically designed to test throughput. BreakingPoint enables you to transmit frames at a specific rate and identifies the fastest rate at which the frames were transmitted by the DUT. The frame rate is determined when the number of transmitted frames equals the number of received frames and the frame rate, and the frame loss% and the corrupted frame% meet the criteria you have defined.

There are three ways in which the test will determine the fastest frame rate at which the DUT successfully transmitted and received traffic: binary, step, and combined searches. For more information on these search types, see the section.

The RFC 2544 test will send packets of different sizes at varying frame rates. You will define the different frame sizes that will be tested and you will define the how the frame rate will be incremented during the each test "load" or "iteration". Each frame size and frame rate combination has its own test iteration, and the iteration will last for however long you have defined. You can either define a duration for the entire test or define the duration for each iteration. For more information on durations, see.

Test Set Up

Before running any tests, verify that your test set up matches the specifications laid out by the RFC 2544. If your test setup only utilizes one DUT, then you must:

- Connect the transmitting ports on the BreakingPoint system to the receiving ports on the DUT.
- Connect the transmitting ports on the DUT to the receiving ports on the BreakingPoint system.

If you require the connection of two devices to the BreakingPoint Storm, please connect them according to the specifications defined in the RFC 2544. For more information of RFC 2544 testing, visit http://www.faqs.org/rfcs/rfc2544.html.

Quick Test - RFC 2544 Requirements and Restrictions

The following list details the requirements and restrictions for the RFC 2544 test:

- There is only one working copy of the RFC 2544 test. Each time a user modifies and saves the test, it will overwrite the existing settings that were originally stored in the test. There is currently no way to export a copy of the RFC 2544 test, nor is there a way to save the test under a

different name.

- The MTU defined for the transmitting and receiving ports on the BreakingPoint system must be able to support the frame sizes defined for the RFC 2544 test. Therefore, you should always check the MTU settings for each port before running the test to ensure that the port supports the frame sizes defined in the test.Resul
- For each iteration, the system will send slow start packets in the reverse direction to the DUT. This enables the DUT to determine the ports of the MAC addresses that the BreakingPoint system is using.
- The RFC 2544 test utilizes logical interface 1 as the transmitting interface and logical interface 2 as the receiving interface.

RFC 2544 Test Editor

The RFC 2544 test editor is composed of four different areas:

- Test Configuration – Defines the DUT Profile and Network Neighborhood for the test.
- Test Load Units – Defines the duration either for the entire test or for each iteration, and defines the packet type and payload.
- Traffic Load – Defines the frame rate for the test, as well as how the frame rate is incremented during that frame load's iteration.
- Frame Size – Defines the frame sizes at which each frames will be sent at for each test iteration.

When you set up the RFC 2544 test, you will need to select the Network Neighborhood and the DUT Profile that the test will use. Then, you must define how long the test will last and how the packets will appear on the wire. For more information on setting the duration, see .

Once you have done that, you are ready for the actual test configuration: defining the frame rates and the frame sizes. For more information on frame rates, see the section on, and for more information on frame sizes, see .

Slow Start Frames

For each iteration, the system will send slow start packets in the reverse direction to the DUT. This enables the DUT to determine the ports of the MAC addresses that the BreakingPoint system is using. Therefore, in the Traffic Overview section of the RFC 2544 test report, you will see slow start packets listed for each data rate that was tested.

Test Duration

There are two ways to set the test duration; you can set:

- The duration for the entire test.
- The duration for each iteration.

Setting the Total Duration for the Test

Setting the total duration for the test is a slight misnomer. The value you specify for the total test duration is an estimate of how long the test may last. The test duration is really calculated based on

the time per iteration, the initialization time for each iteration, and the total number of iterations for the entire test, or:

Total Test Time = (Time Per Iteration + Initialization Time Per Iteration) * Number of Iterations

The system will estimate that the total initialization time for each iteration is to be 20 seconds; however, depending on the DUT, this time may vary.

The number of iterations and the time per iteration are both values that fluctuate based on the test's configuration, therefore, neither of these values is constant.

To set the duration for the test:

1. Enter an integer in the **Test Duration** field, located under the **Test Load Units** area of the Quick Test - RFC 2544.
2. Select seconds, minutes, or hours from the drop-down menu located next to the **Test Duration** field.
3. Click the **Total** radio button.

Setting the Duration for Each Iteration

By setting the duration for each iteration, you can better estimate the total duration of the test. As previously mentioned, the test duration is calculated based on the time per iteration, the initialization time for each iteration, and the total number of iterations for the entire test, or:

Total Test Time = (Time Per Iteration + Initialization Time Per Iteration) * Number of Iterations

The initialization time for each iteration is estimated to be 20 seconds; therefore, if you know how many iterations the test will have, you can estimate the total amount of time the test will take. For example, if you set the time per iteration to 30 seconds, and you know the number of iterations is 12, then:

Total Test Time = (30 + 20) * 12

Note: To determine the number of iterations per test, see the Calculating the Number of Iterations section.

To set the duration for each iteration:

1. Enter an integer in the **Test Duration** field, located under the **Test Load Units** area of the Quick Test - RFC 2544.
2. Select seconds, minutes, or hours from the drop-down menu located next to the **Test Duration** field.
3. Click the **Per Iteration** radio button.

Calculating the Number of Iterations

In order to calculate the total number of iterations in an RFC 2544 test, you will need to know the number of frame sizes that will be tested and the number of frame rates at which those frame sizes will be tested.

For example, the following test configuration will have 30 iterations:

- Mode: Step
- Rate lower limit: 50%
- Rate higher limit: 100%
- Step rate: 10%
- Frame Sizes: 62, 128, 256, 512, 1024

To determine the number of iterations, we used the following calculation:

Total Iterations = Number of Frame Sizes Tested x Number of Frame Rates Tested

In this case, the R**ate lower limit** of 50% and the **Rate higher limit** of 100% with a **Step Rate** of 10% yields 6 frame rates (one at 50%, 60%, 70%, etc.). There are five frame sizes, thus:Total Iterations = 6 x 5

Defining the Payload

You can set up the payloads for the test from the Test Load Units area of the Quick Test - RFC 2544. By defining the payload, you are setting up how the traffic will appear on the wire. The table below lists the payload options and descriptions.

Note: Packets generated by this test will reserve the last 16 bytes of the payload for internal use by BreakingPoint Systems. These bytes will not contain the payload value that you have defined, if any.

Payload Fields

Field	Description
Packet Type	Sets the packet type; packets can be Ethernet, IP, UDP, ICMP, or TCP packets.
Packet Width	Defines the width of the data (in bits) being inserted into the payload; the width can be 8, 16, or 32.

Field	Description
Payload	The payload can be set to be any of the following: **0** – Payload is 0s. **1** – Payload is all 1s. **Random** – Payload is defined using random Hex values. **Increment** – Payload is defined using ascending values starting at 0. **Decrement** – Payload is defined using descending values starting at 0xff. **User-Defined**– Payload is defined using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
User Defined	You can use standard hexadecimal notation to define the payload; this information is inserted after the Ethernet header. This field is defined only if you have set the Payload to be **User Defined**.

Frame Rate

From the Traffic Load for Throughput Search area of the Quick Test - RFC 2544, you can define the maximum throughput for the test. You can either specify **Maximum Possible**, which will use the maximum throughput possible for the port, or you can explicitly specify the throughput. To do this, simply deselect the **Maximum Possible** option, and specify the throughput in Mbps or Gbps.

Frame Rate Searches

There are three ways to find the successful frame rate for each iteration:

- Binary Search
- Step Search
- Combined Search

Binary Search

A binary search is the quickest way to determine the iteration's fastest frame rate. Using this search method, the test will use the percentages you have defined for Rate Lower

Limit and Rate Higher Limit to determine the transmit rate.The test will select the Lower Limit of Overall Load as the starting transmit rate. For example, If the Overall Load is configured as 5000Mbps and the Rate Lower limit is 10%, the starting transmit rate will be 500Mbps ((5000*10)/100).

If the starting transmit rate is achieved, the next transmit rate will be a rate that is the sum of half of the lower limit and the higher limit, as long as the difference between the upper and lower limit is less than the resolution. For example, if you have set Rate Lower Limit to 10%, Rate Higher Limit to 100%, and the Resolution to 10%, the next transmit rate will be 2750Mbps. The value of Resolution is 500Mbps and the difference between the lower limit and the next transmit rate is greater than the Resolution. Please see the table below for additional scenarios (which apply regardless of frame size).

Iteration	Attempted Frame Rate	Achieved Frame Rate	Result	Resolution Check
1	50	50	Achieved	
2	2750	2000	Failed	(2750-50)>500
3	1400	1400	Achieved	(2750-1400)>500
4	2075	2075	Failed	(2075-1400)>500
5	1738	1738	Achieved	(2075-1738)<500

- Attempted Frame Rate 1^{st} Iteration (Low) = (Lower Limit * Overall Load)/100.
- Attempted Frame Rate for 2^{nd} iteration (Mid) = (Low + Overall Load)/2.

Note: The Maximum Throughput is defined in the Overall Load area.

If the DUT successfully transmits at the attempted frame rate, the system will increase the frame rate to be half of the last frame rate and the higher limit. If the DUT does not successfully transmit at the attempted frame rate, the system will decrease the frame rate to half of the last frame rate and the lower limit.

Step Search

This is the most straight-forward method of finding the fastest frame rate. Basically, you will set **Rate lower limit** and **Rate higher limit**; these will serve as the lower and upper bounds of your throughput. Additionally, you will set the **Step Rate**; the test will use this value to increment Rate lower limit until it reaches Rate higher limit.

For example, if you set Rate lower limit to 10%, Rate higher limit to 100%, and the step rate to 10%, the test will start at 10% and increment the frame rate by 10% until it reaches 100%.

Combined Search

The combined search will start with a step search: it will first step through the frame sizes and step through the data rates; while it is stepping through the data rates, it will perform a binary search between the successful data rates and the unsuccessful data rates.

During a step search, you will set **Rate lower limit**, **Rate higher limit**, and **Step Rate**; the test will use the **Step Rate** to increment Rate lower limit until it reaches Rate higher limit.

With the binary search, the system starts at a rate that is halfway between the **Rate lower limit** and the **Rate higher limit**. If the DUT successfully transmits at that frame rate, then the system will increase the frame rate to be half of the last frame rate and the higher limit. If the DUT does not

successfully transmit at that frame rate, then the system will decrease the frame rate to be half of the last frame rate and the lower limit.

The test will continue using the binary search until it reaches the Resolution%.

Ignoring Pause Frames

To configure the system to ignore pause frames, you will need to disable pause frames from the port settings. This feature is useful for ignoring pause frames transmitted during RFC 2544 testing.

To ignore pause frames:

1. Select **Control Center** > **Device Status** from the Menu bar.
2. Right-click on a port on a reserved blade.
3. Select **Configure Port** from the menu.
4. Select **Ignore Pause Frames**.
5. Click the **Apply** button.

Frame Sizes

Since the RFC 2544 requires that each test condition be tested using five different frame sizes, BreakingPoint provides the ability to set frame sizes for the test. There are several ways to define the frame sizes:

- Random – The system will randomly select frame sizes that fall between the minimum and maximum frame sizes defined.
- Step – The test will start at the frame size specified, and it will increment the frame size based on the value defined for **Interval**.
- RFC 2544 – The test will use the frame sizes recommended by the RFC 2544.
- User Defined – The test will use the frame sizes you have defined. You can enter the frame sizes by separating each frame size with a comma (e.g., 64,128,256, etc.).

Note: If you define a frame size that is larger than 1,500 bytes, then you must define a MTU for the data ports that supports the specified frame size. For more information on setting the MTU, see.

Setting the MTU for a Data Port

The following section provides instructions for setting the MTU for a data port. Instances in which you may want to modify the MTU includes:

- Changing the MTU to support jumbo frames
- Changing the MTU to not support jumbo frames

To define the MTU for a data port:

1. Select **Administration > Device Status** from the Menu bar.
2. Right-click on the port for which you would like to modify the MTU. A menu will display, listing the port options.

You must have the port reserved in order to configure the port settings.

3. Select **Configure Port** from the menu. A window will display, enabling you to set the MTU.
4. Enter the MTU in the MTU field.
5. Values of 46 – 9,198 are supported.
6. Click the **Apply** button.

Session Sender Lab General Information

The Session Sender Lab is a dedicated test lab that allows you to validate your device's simultaneous TCP performance. Session Sender measures a device's ability to set up and maintain a large number of TCP sessions over a period of time. Each session uses a unique combination of source addresses, destination addresses, source ports, and destination ports; therefore, there must be enough MAC/network address combinations allotted in the domain and enough source/destination port combinations to create that many sessions.

With Session Sender, you can control:

- The maximum number of simultaneous TCP sessions
- The rate at which sessions are opened
- The duration of the sessions

Session Sender Lab Requirements and Restrictions

The following list details the requirements and restrictions for the Session Sender test:

- There is only one working copy of the Session Sender test. Each time a user modifies and saves the test, it will overwrite the existing settings that were originally stored in the test. There is currently no way to export a copy of the Session Sender test, nor is there a way to save the test under a different name.
- For each iteration, the system will send slow start packets in the reverse direction to the DUT. This enables the DUT to determine the ports of the MAC addresses that the BreakingPoint system is using.
- The Session Sender test utilizes logical interface 1 as the transmitting interface and logical interface 2 as the receiving interface.

Session Sender Lab Editor

The Session Sender Lab Editor is composed of five different areas:

Device Configuration – Defines the DUT Profile and Network Neighborhood for the test.

- Payload – Defines the type of payload being tested.
- TCP Connections – Defines the type of connection being used for the test.
- TCP Options – Defines the parameters of the connection being used for the test.
- Test Control – Sets the total duration for the test.

Session Sender Lab Parameters on the facing page lists the parameters for the Session Sender Lab.

Session Sender Lab Parameters

Parameter	Description	Valid Values
Device Under Test	Searches for the device to be tested and its corresponding Network Neighborhood.	A BreakingPoint DUT or a custom DUT
Network Neighborhood	Searches for the device to be tested and its corresponding Network Neighborhood.	A BreakingPoint Network Neighborhood or a custom Network Neighborhood
Layer 4 only	Limits the payload to transport layer traffic.	Check or Uncheck
Data Type	Selects the method used to determine the maximum TCP connection establishment rate through or with the DUT.	0, 1, Random, HTTP
Packets Per Session	Specifies the number of data segments that are sent during each session.	-1 – 10000
Source Port Range	Sets the distribution of source ports for TCP connections. It specifies that the ports will be used sequentially, from minimum to maximum	1024 – 65,535
Destination Port Range	Sets the destination port for all TCP/UPD packets. Setting this parameter to 0 will randomize this value.	0 – 65,535
Application Profile	Sets the Application Profile that determines the mix of applications that will be used in the traffic.	A BreakingPoint Application Profile or a custom Application Profile
Test Mode	Sets the test mode for the test.	Maximum Mixed Open/Close Sessions, Maximum Session Open Rate, Maximum Concurrent Sessions
Minimum Rate	Specifies the connection establishment rate to be used at the start of the ramp up phase when not in Calculated mode. Must be less than or equal to Maximum Rate.	1 – 750,000
Maximum Rate	Limits the maximum connection establishment rate for the ramp up phase when not in Calculated mode.	1 – 750,000

Parameter	Description	Valid Values
Concurrent	The number of TCP sessions concurrently open at any given time.	1 – 20,000,000
Minimum Concurrent	The number of sessions that must open to pass the test.	1 – 20,000,000
Maximum Concurrent	Sets the maximum number of simultaneous sessions that will exist concurrently during the test duration.	1 – 20,000,000
Retry Quantum	Sets the amount of time (in milliseconds) that elapses before a connection is retried.	100 – 2,000
Retries	Sets the number of times a connection is attempted before it is canceled.	0 – 7
Aging Time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state after closing. Flows in the TIME_WAIT count against simultaneous sessions, but do not generate any traffic. Setting this to a high value can cause traffic to slow over time. This setting is useful for matching the flow tracking policy of the DUT.	0 – 120

Parameter	Description	Valid Values
Steady State Behavior	Determines how sessions are handled during the steady-state phase.	Open and Close Sessions – sessions are closed as they finish sending data and new ones opened in their place. Hold Sessions Open – sessions are not closed as they finish sending data. Open and Close with Reset – initiate the TCP close with a RST. This bypasses the TCP close state machine. Open and Close with Reset Response – respond to a FIN with a RST. This bypasses the TCP TIME_WAIT state.
Step Rate	The test will use this value to increment Rate lower limit until it reaches Rate higher limit.	1 - 100
Test Duration	Sets the length of the test.	hh:mm:ss
Total	Provides statistics for the total test.	Check or Uncheck
Per Iteration	Provides statistics for each iteration of the test.	Check or Uncheck

Creating a Session Sender Lab Test

The following section provides instructions on creating a test with the Session Sender Lab.

To create a test using the Session Sender Lab:

1. Select Test>Session Sender from the Menu bar. You can also select the Labs button from the Home page and select the Session Sender button on the Labs dialog box.

2. Click the Browse for Network Neighborhood and Device Under Test buttons to select the Network Neighborhood and the device to be used in your test.
3. Check the Layer 4 only check box to limit the payload of your test to transport layer traffic only. Uncheck this box to include traffic from layers 2 through 7.

   Note:
   - When this box is unchecked, the test is run as an Application Simulator test. The resulting report will be indentified as an Application Simulator test report.
   - When this box is unchecked, you have the option of selecting any available Application Profile by clicking the Browse Application Simulations button located next to the Application Profile field.
4. Select the method to be used to determine the maximum TCP connection establishment rate through or with the DUT with the Data Type drop-down list.
5. Enter the number of data segments to be sent during each session in the Packets Per Session field.
6. Enter the distribution of source ports for TCP connections in the Source Port Range fields.
7. Enter the destination port for all TCP/UPD packets in the Destination Port Range fields. Enter 0 (zero) to randomize this value.
8. Select the appropriate Test Mode from the drop-down list.
9. Enter the Minimum Rate, Maximum Rate, and number of Conncurrent sessions in the TCP Connections section.
10. Enter the amount of time (in seconds) that will elapse before a connection is retried in the Retry Quantum field.
11. Enter the number of times a connection will be attempted before it is canceled in the Retries field.
12. Enter the number of seconds that an actively closed TCP connection will remain in the flow table in the TIME_WAIT state after closing in the Aging Time field.
13. Select the method of handling sessions during the steady-state phase from the Steady State Behavior drop-down list.
14. Enter the rate at which to increment the rate of the test in the Step Rate field.
15. Enter the length of the test in the Test Duration field.
16. Click Save to save your test.
17. Click Save and Run to run your test.

Note: Because Application protocols can override the close method directly in the Super Flow, depending on the Application Profile that you select, the test may use a close method other than the one you selected.

DDoS Lab Overview

Distributed Denial of Service (DDoS) is a type of DoS attack where multiple compromised systems are used to temporarily or indefinitely interrupt or suspend services of a host connected to the Internet. DDoS attacks are often global attacks.

The DDoS Lab allows you to build DDoS test scenarios in order test your network inline (transparent) device's ability to withstand a DDoS attack. These are two-arm tests.

DDoS Lab Features:

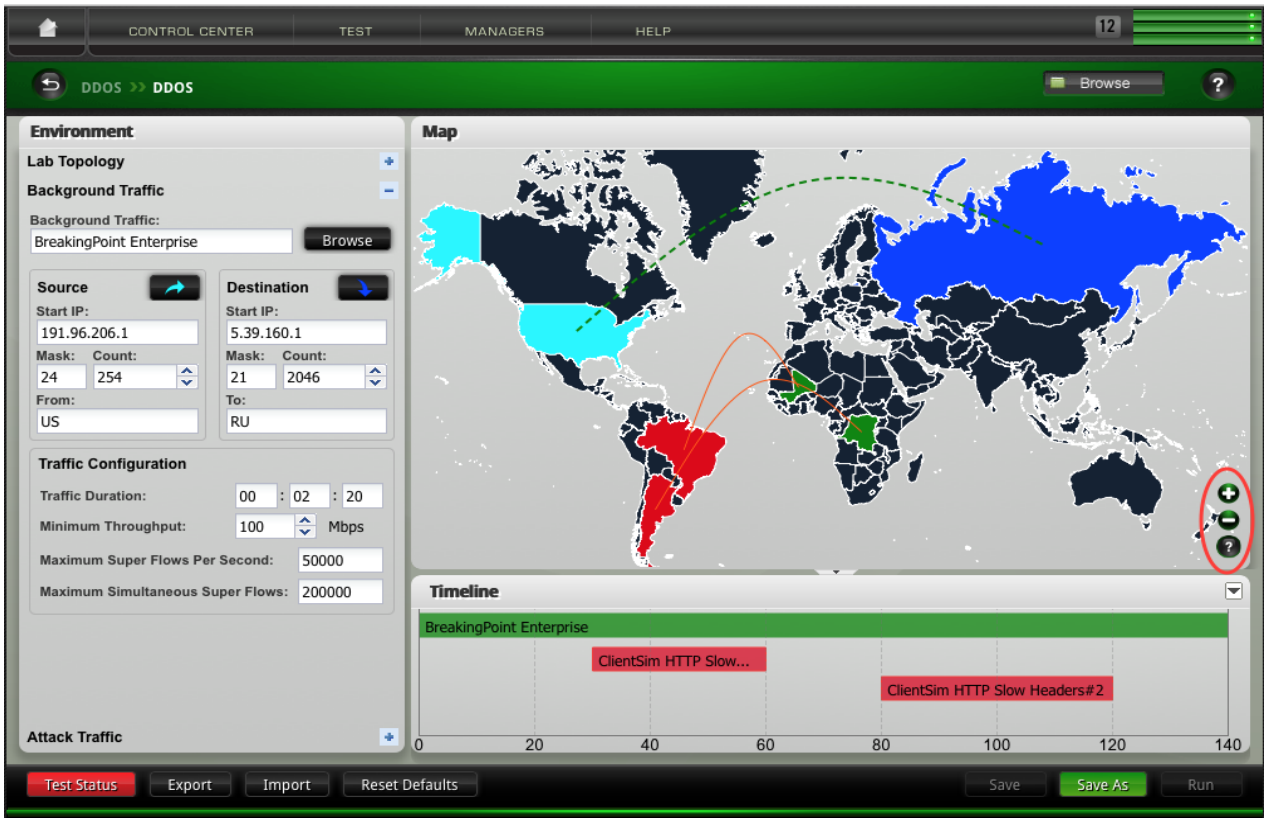
- Select background traffic from previously saved application mixes (both custom and BreakingPoint canned).
- Select attacks from a list of predefined DDoS attacks
- Use predefined DDoS Labs that recreate public DDoS events (available by way of ATI updates)
- Access to stateless and stateful DDoS patterns
- Consolidated DDoS Lab Real-Time Statistics (requires a valid ATI subscription at the time of the BPS 8.0 release)

Note: You can create custom super flows that exercise various DDoS attack types by using the "DDoS" tag.

DDoS UI Overview

To view the DDoS Lab options:

1. Select **Labs** > **DDoS** from the Admin page.
2. The Test Browser is displayed. Previously created DDoS tests will display in the browser list. You can also **Search** for specific tests.
3. Double-click a previously created DDoS Lab test or click **Create New** to create a new DDoS Lab test. The DDoS UI is displayed as shown in the image below.



The following information describes DDoS Lab UI Options:

- **Environment** - The Environment area gives you the ability to define traffic constraints, background traffic duration, and an attack's start, delay, and duration. The "+" and "-" controls can be used to expand and collapse the 3 sections of the Environment area.
  - **Lab Topology**
  - **Background Traffic**
  - **Attack Traffic**
- **Map**- Geo-location features provide the origination and destination of attacks and traffic.
  - Use the **"+"** and "**-**" icons (highlighted at the bottom right corner of the map) to control the zoom level of the map.
  - Click the "**?**" icon to view a legend that describes the colors and indicators on the map. For example, cyan (light blue) = traffic source and dotted green lines = background traffic.
  - Hover the mouse over locations on the map for the location description (country). The country that is attacking/being attacked may also display if the country is configured in that manner.

- **Timeline** - Displays the timeline of events that will occur based on the lab configuration. The X axis displays time and adjusts based on the parameter values.
  - Hover over timeline objects for a summary of the object configuration.
  - When you click a timeline object (such as an attack), the items on the map that pertain to the attack are highlighted. Other objects are faded. The configuration details for the attack are also displayed in the **Attack Traffic** section.
  - Click the object again to toggle back to the full map view or select the Background Traffic object (green object).
- **Controls** - Standard UI controls are available in the area at the bottom of the window. For example, **Save**, **Save As** and **Test Status**. The Map and Timeline areas of the window can be collapsed and expanded using the arrowhead icons that are displayed in these areas.

Create a DDoS Lab Test

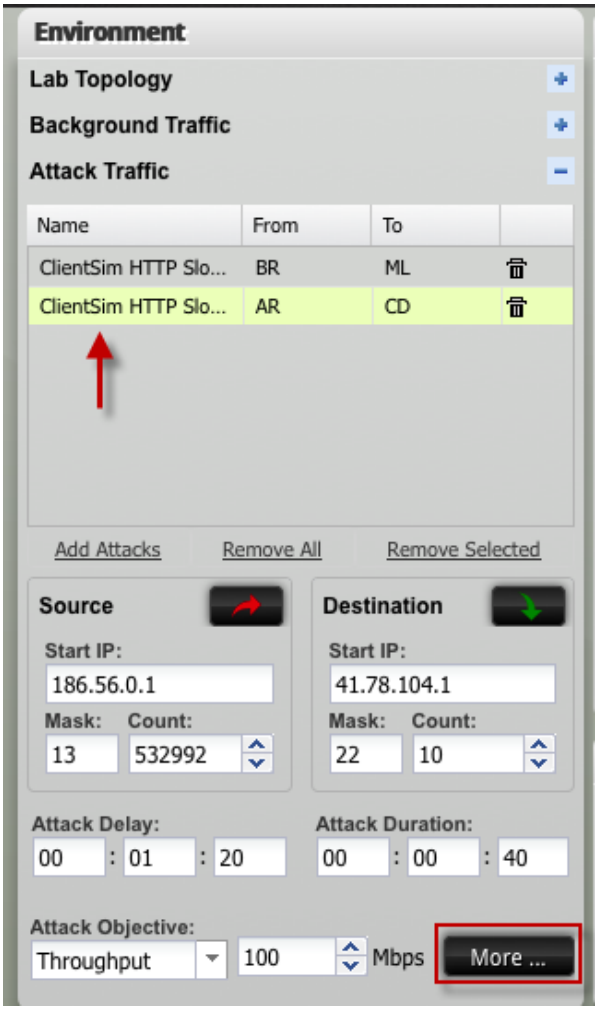
To create a new DDoS Lab test:

1. Select **Labs** > **DDoS** from the Admin page.
2. Click **Create New** at the bottom of Test Browser window.
3. Enter a **New Test Name**.
4. Click **OK**. The DDoS Lab user interface will launch.
5. Expand the **Lab Topology** section. Two physical test interfaces are used to host the simulated clients and the servers respectively, which will be logically sitting behind virtual routers. The Virtual Router IPs are configured as per the following settings.
   a. Enter the**Source**and**Destination IP Address, Mask**and**Gateway**of your lab.
6. Expand the **BackgroundTraffic** section.
   a. To configure the **Background Trafficfield**, click **Browse** to select an Application Profile that provides the traffic flows that you want to for your test and then click **OK**.
7. In the **Source** section, click the cyan (blue-green) colored arrow in order to select the country that will be the traffic source.

a. Then click a country on the map. The **Start IP**, **Mask** and **Count** fields are automatically filled with default values based on the selected country's IANA assignment. You can manually change these values.
   i. The **Count** value indicates the number of hosts that will be used. Note that the defined **Mask** must be configured to support the desired number of hosts, for example a "24" mask supports up to 254 addresses.
   ii. The **From** field displays an abbreviation for the selected country.

8. In the **Destination** section, click the blue arrow in order to select the country that will be the traffic destination. See the configuration instructions described in step 7 - the same information is applicable here (except that you are configuring the Destination instead of the Source).
9. Input and select the values for the **Traffic Configuration**.
   a. Traffic Duration (The duration of the background traffic)
   b. Minimum Throughput
   c. Maximum Super Flows Per Second
   d. Maximum Simultaneous Super Flows
10. Expand the Attack Traffic section.
   a. Click the **Add Attacks** link. The **Select Attacks** window displays.
   b. Click the "+" symbol next to the attacks you want to add. The selected attacks will move to the right side of the window. Click **OK** to add the attacks. The attack will be displayed on the **Attack Traffic** list in the Environment.
   c. Select an attack (as shown in the image below) and configure the settings for the attack. Repeat this step to configure unique values for each attack.
   d. Click the **More** button to configure the values for: **Maximum Super Flows Per Second**

and **Maximum Simultaneous Super Flows**.



11. Your DDoS Lab test is now configured. The control area at the bottom of the window provides several options such as **Save**, **Reset Defaults**, etc. Click **Run** to run your test. If the **Run** button is grayed out, click **Test Status** for information that can help you resolve the issue so that the test can be run.

Resiliency Score Lab General Information

Resiliency is a device's ability to maintain an acceptable level of service when challenges to normal operations occur. The Resiliency Score Lab consists of a set of standardized tests for measuring the resiliency of your network devices, allowing you to determine their true level of security, performance, and stability. The tests consist of a rating scheme that compares actual device performance to a theoretical maximum, and a configuration screen for easily conducting reproducible tests.

The Resiliency Score page requires minimal configuration and allows you to choose the device type and speed along with the subset of tests desired. Test progress and estimated time remaining are displayed.

The results of each test are scaled and compared to a mathematically determined theoretical maximum. The results yield either a Failed rating or a numeric value between 1 and 100. The results of security testing, if performed, are appended as an additional 1-100 score.

This allows the test results of a given device to be directly compared to the results of other devices within the same category. To standardize test results over time, tests are annotated with the OS and ATI Updates employed. Users wishing to directly compare test results obtained from different chassis will need to install the appropriate OS and ATI Updates on their BreakingPoint system.

The testing subjects (DUTs) are evaluated across a specified set of criteria. Within each of the tests, certain minimal performance criteria must be met to prevent a Failed rating for the device (for example, a device will receive a fail rating if it is unable to keep packet latency under a certain level while maintaining a specified packet-per-second throughput). A device will also fail if it stops responding or if it stops forwarding traffic. The DUTs are classified into the following categories:

- Switch
- Router
- Firewall
- Proxy
- Intrusion Prevention System (IPS)
- Unified Threat Manager (UTM)
- Application Server
- Data Center

In addition, each device is measured based on its device capacity and its attack survivability. Again, these criteria will affect the tests to which a DUT is subjected. The testing categories are:

- Throughput
- Sessions
  - When this option is selected, BPS generates a test based on the canned Resiliency Basic SessionRate test model. This test uses a Session Sender component configured to run 750.000 flows/s at 100.000.000 maximum concurrent flows. When configuring two pairs of ports, the test will use two identical Session Sender components, effectively doubling these figures.
- Robustness
- Security
- Web Virtual Machines
- Storage Virtual Machines
- Database Virtual Machines
- Email Virtual Machines

Each of these categories is composed of a number of subtests. Different tests may be performed at different rates, depending on the device type and capacity.

Device Types

Resiliency Scores are constructed to uniquely evaluate the performance and resilience of a set of network devices. These device types are selectable from the Resiliency Score page. Each DUT category has a specified configuration, embodied in the corresponding Network Neighborhood. The DUT should be configured to match the test.

Note: Resiliency Scores are not network protocol compliance or conformance scores.

The device types available for Resiliency Score testing include:

- Switch – primarily a Layer 2 and 3 forwarding device, with the same IP network on each interface.
- Router – a networking device that connects and directs packets between different Layer 3 networks
- Firewall – a Layer 3 device that selectively allows or blocks certain traffic; the test methodology will prescribe which ports/services must be allowed by the firewall
- Proxy – a device which terminates a Layer 4 connection on one interfaces and regenerates it on another interface (such as an HTTP proxy)
- Intrusion Prevention System (IPS) – Layer 2 device that connects two segments of a layer 3 network and blocks certain malicious traffic while allowing all non-malicious traffic
- Unified Threat Management (UTM) – a Layer 2 and 3 device combining firewall and Intrusion Prevention System capabilities
- Application Server – A single virtual machine with four available services (mail, Web, SQL, and file sharing)
- Data Center – A collection of virtual machines (VMs), each specialized to run one service per VM

Test Status Indication

While a test is underway, an on-screen indication shows the progress of the testing regimen, including approximate time remaining. The display also provides an indication of the DUT's performance while the tests are being performed.

Test Pass/Fail

If a device fails the Resiliency Score while the test is underway, the device will receive a Failed rating. A description of what made the test fail is provided so that corrective action can be taken on the DUT. This information includes, but is not limited to, what test was being conducted when the device stopped responding.

When a device receives a Resiliency Score displayed as a hyphen (-), it means that the test has failed and that a score of "0" (zero) has been issued for the device. This can occur if the test was interrupted and unable to be completed.

If your device receives a Resiliency Score displayed as a hyphen (-), run the Resiliency Score test again.

Port Reservation

Ports 1 and 2 must be connected to non-Application Server and Data Center DUTs. Only one port connection to Application Server and Data Center DUTs is required. However, all ports on a blade must be reserved for all Resiliency Score testing.

DUT Performance Rate

For each DUT, you will need to select the appropriate Device Capacity based on the number of users (for Application Server or Data Center) or the rated performance speed of the DUT being tested. Resiliency Scoring is used to determine a device's resiliency at a given throughput level.

Note: Specified throughput should be per interface or interface pair, not backplane speed.

Resiliency Score Parameters

For Resiliency Scoring, each network device being tested requires at least one Network Neighborhood interface. The Network Neighborhood interface settings have been pre-configured for each type of network device.

The table below provides the required IP address for testing the resiliency of each type of network device. Find the type of device that you want to test in the table. Configure the IP address of the device you are testing to match the corresponding IP address found in the table.

Resiliency Scoring IP Addresses

Network Device	Interface 1 IP Address	Interface 2 IP Address
Switch	10.0.0.1	10.0.0.1
Router	192.168.50.1	192.168.51.1
Firewall	192.168.50.1	192.168.51.1
Proxy	192.168.50.1	192.168.51.1
Intrusion Prevention System	10.0.0.1	10.0.0.1
Unified Threat Manager	192.168.50.1	192.168.51.1
Application Server	User defined	N/A
Data Center (File Server)	User defined	N/A
Data Center (DB Server)	User defined	N/A
Data Center (Web Server)	User defined	N/A
Data Center (Mail Server)	User defined	N/A

Test Setup

The DUT will have its interface(s) connected to the BreakingPoint system. These connections will be referred to as Logical Interface 1 and Logical Interface 2 (for those devices requiring two interfaces), based on the interface reservation on the BreakingPoint system.

Note: Some devices require only one logical interface.

Network Configuration

In preparation for the test, the DUT must be configured to support the appropriate network configuration for that device.

Switch

The network configuration used to test a device classified as a switch will be composed of two separate IP ranges within the same subnet, both directly attached to the device's network.

Router

The network configuration used to test a device classified as a router will be composed of two ranges of hosts in non-local networks. Each will arrive at the DUT via a router attached to a separate local subnet of the DUT. Traffic is expected to be routed through gateway IPs on the device.

Firewall

The network configuration used to test a device classified as a Firewall will be composed as follows

- Traffic originates from a network of client addresses in a local subnet.
- Client requests are handled by a set of hosts that are simulated as a multi-homed host. That is, the set of IP addresses will all originate from a single MAC address, avoiding the possibility of overflowing MAC tables on the device.
- Server hosts will be listening on a set of hosts on a nonlocal subnet.
- The server addresses are reachable by the DUT via a router on a local subnet of interface 2.
- The DUT is expected to translate the source address to one from an unspecified pool, which must be reachable by the server hosts. The specific client addresses are learned as they are observed.

Proxy

The network configuration used to test a device classified as a Proxy will be composed as follows:

- Traffic originates from a network of client addresses in a local subnet.
- Client requests are handled by a set of hosts that are simulated as a multi-homed host. That is, the set of IP addresses will all originate from a single MAC address, avoiding the possibility of overflowing MAC tables on the device.
- Server hosts will be listening on a set of hosts on a nonlocal subnet.
- The server addresses are reachable by the DUT via a router on a local subnet of interface 2.
- The DUT is expected to translate the source address to one from an unspecified pool, which must be reachable by the server hosts. The specific client addresses are learned as they are observed.

See Proxy Support on page 173 for detailed information about the BPS Proxy Implementations.

Intrusion Prevention System

The network configuration used to test a device classified as an Intrusion Prevention System will be composed of two separate IP ranges within the same subnet, both directly attached to the device's network.

Unified Threat Manager

The network configuration used to test a device classified as a Unified Threat Manager will be composed of two ranges of hosts in nonlocal networks. Each will arrive at the DUT via a router attached to a separate local subnet of the DUT. Traffic is expected to be routed through gateway IPs on the device.

Application Server

The network configuration used to test a device classified as an Application Server will be composed of one IP address attached to the device's network.

When creating a network configuration for an Application Server, ensure that the device is connected to the BreakingPoint port reserved as Logical Interface 1 and that the entire blade is reserved. You can create a network configuration for either a VM Application Server or a non-VM Application Server.

To create a network configuration for a VM Application Server:

1. Log into the VMware® ESXi client vSphere as an account that can import templates, edit configurations, and start VMs.
2. From Ixia Support website, download the following file to the vSphere client PC, `BPSTemplateGenerator.ova`.
3. From vSphere client, go to Inventory.
4. Go to **File** in the top menu.
5. Select **Deploy OVF Template**.
6. Browse to the path containing the file BPSTemplateGenerator.ova on the vSphere client PC and click **Next**.
7. Click **Next** again. In this window, give this instance a distinct name.
8. Continue clicking **Next**, accepting the default options.
9. Click **Finish** to begin the VM import procedure.
10. Select the new VM and click **Edit Virtual Machine Settings** once the import is complete.
11. Click **Network Adapter 1**. Select the network label that can access the VM Management network.
12. Click **OK** to close the window and accept the new settings.
13. Right-click the newly created VM and select **Open Console**.
14. Click **Play** to start the VM.

When the VM has completed it's boot up process, you will see a screen with instructions. Follow the instructuctions on screen to generate an Application Server Deployment.

To create a network configuration for an x86-64 non-VM Application Server:

Note: BreakingPoint has tested Ubuntu 10.04 running on an x86_64 architecture. While it is possible that Ubuntu 10.04 running on other architectures may work using the instructions in the previous section, BreakingPoint makes no claim to that effect.

1. Install Ubuntu 10.04.2-desktop-amd64.iso.

Note: The system you are using should have at least 20 gigabytes of disk space and at least 1 gigabyte of RAM. BreakingPoint also recommends that you dedicate one network interface for management and one for running tests.

2. Boot from an ISO or DVD/CDROM.
3. At the prompt, click **Install Ubuntu 10.04.2 LTS**.
4. Set your timezone and click **Forward**.
5. Select **Suggested option: USA** and click **Forward**.
6. Layout your disk and click **Forward**.
7. Enter these settings on the panel:
   a. Server Name: **resiliency-server**
   b. Username: **testuser**
   c. Password: **testuser**
8. Click **Forward**.
9. Click **Install**.
10. Click **Restart Now** when prompted.
11. Boot into Ubuntu and log in as testuser.
12. Enter testuser as the password.
13. From the GUI, select **Applications** > **Accessories** > **Terminal**.

At the prompt, type su as the root. For example:

```
testuser@resiliency-server:~$ sudo su
[sudo] password for testuser:
<testuser>
root@resiliency-
server:/home/testuser#
```

Install sendmail. For example:

```
root@resiliency-server:/home/testuser# apt-get install
sendmail
??????????
??????????
Do you want to continue: [Y/n]? Y
??????????
??????????
...
Setting up sendmail (8.14.3-9.1ubuntu1) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
root@resiliency-server:/home/testuser#
```

Install mysql-server. For example:

```
root@resiliency-server:/home/testuser# apt-get install mysql-
server
??????????
??????????
Do you want to continue: [Y/n]? Y
( At the "Configuring mysql-server-5.1" window, enter
"1q2w3e4r5t"
(without the quotes) as the password for the "MySQL root"
account.
Tab over to "Ok" and hit <ENTER>. Repeat the password in the
next
window and again tab over to "Ok" and hit <ENTER> )
??????????
??????????
...
Setting up mysql-server (5.1.41-3ubuntu12.10) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
root@resiliency-server:/home/testuser#
```

Install Samba. For example:

```
root@resiliency-server:/home/testuser# apt-get install
samba
?????????? ?????????? Do you want to continue: [Y/n]? Y
??????????
??????????
...
smbd start/running, process ????
nmbd start/running, process ????

Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
root@resiliency-server:/home/testuser#
```

Install apache2. For example:

```
root@resiliency-server:/home/testuser# apt-get install
apache2
??????????
??????????
Do you want to continue: [Y/n]? Y
??????????
??????????
...
Setting up apache2 (2.2.14-5ubuntu8.4) ...

Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
root@resiliency-server:/home/testuser#
```

Install openbsd-inetd. For example:

```
root@resiliency-server:/home/testuser# apt-get install openbsd-
inetd
??????????
??????????
...
Setting up openbsd-inetd (0.20080125-4ubuntu2) ...
* Stopping internet superserver inetd [ OK ]
* Starting internet superserver inetd [ OK ]

root@resiliency-server:/home/testuser#
```

Install Qpopper from source and build. For example:

```
root@resiliency-server:/home/testuser# mkdir /root/src
root@resiliency-server:/home/testuser# cd /root/src
root@resiliency-server:/~src#
wget ftp://ftp.qualcomm.com/eudora/servers/unix/popper/qpopper4.0.19.tar.gz
root@resiliency-server:/~src# gunzip qpopper4.0.19.tar.gz
root@resiliency-server:/~src# tar xvf qpopper4.0.19.tar
root@resiliency-server:/~src# cd qpopper4.0.19
root@resiliency-server:/~src/qpopper4.0.19# ./configure --enable-standalone --
enable-specialauth
root@resiliency-server:/~src/qpopper4.0.19# make
??????????
??????????
...
make[1]: Leaving directory '/root/src/qpopper4.0.19/popper'
root@resiliency-server:/~src/qpopper4.0.19# cd /home/testuser
root@resiliency-server:/home/testuser#
```

Shutdown all new running services. Ignore any errors. For example:

```
root@resiliency-server:/home/testuser# /etc/init.d/smbd stop
root@resiliency-server:/home/testuser# /etc/init.d/nmbd stop
root@resiliency-server:/home/testuser# /etc/init.d/apache2
stop
root@resiliency-server:/home/testuser# /etc/init.d/sendmail
stop
root@resiliency-server:/home/testuser# /etc/init.d/mysql stop
```

Install the BreakingPoint Resiliency Server Configuration files. For example:

```
root@resiliency-server:/home/testuser# mkdir ResConfig
root@resiliency-server:/home/testuser# cd ResConfig
To obtain the configuration files, you will need a valid Strike Center login.
If you do not have one, please go to https://strikecenter.ixiacom.com and
select "Create Account" directly under the "Log In" button. If you are unable
to create an account, contact your Account Manager or Breaking Point Systems
Support. Once you have a username and password, replace <USERNAME> and
<PASSWORD> in the command below:
root@resiliency-server:/home/testuser/ResConfig#
wget --no-check-certificate --user=<USERNAME> --password=<PASSWORD> \
https://strikecenter.ixiacom.com/bps/resiliency/BPSResiliencyConfigFiles.tar
root@resiliency-server:/home/testuser/ResConfig# tar xvf
BPSResiliencyConfigFiles.tar
root@resiliency-server:/home/testuser/ResConfig# mv apache2.tar samba.tar /etc/
root@resiliency-server:/home/testuser/ResConfig# mv sendmail.tar mysql_etc.tar
/etc/
root@resiliency-server:/home/testuser/ResConfig# cd /etc/
root@resiliency-server:/etc# rm -fr apache2/ samba/ mail/ mysql/
root@resiliency-server:/etc# tar xvf apache2.tar ; rm apache2.tar
root@resiliency-server:/etc# tar xvf samba.tar ; rm samba.tar
root@resiliency-server:/etc# tar xvf sendmail.tar ; rm sendmail.tar
root@resiliency-server:/etc# tar xvf mysql_etc.tar ; rm mysql_etc.tar
root@resiliency-server:/etc# cd -
root@resiliency-server:/home/testuser/ResConfig# mv www.tar /var/
root@resiliency-server:/home/testuser/ResConfig# cd /var
root@resiliency-server:/var# rm -fr www/
root@resiliency-server:/var# tar xvf www.tar; rm www.tar
root@resiliency-server:/var# cd -
root@resiliency-server:/home/testuser/ResConfig# mv mysql.tar samba_var.tar
/var/lib
root@resiliency-server:/home/testuser/ResConfig# cd /var/lib
root@resiliency-server:/var/lib# rm -fr mysql/ samba/
root@resiliency-server:/var/lib# tar xvf mysql.tar ; rm mysql.tar
root@resiliency-server:/var/lib# tar xvf samba_var.tar ; rm samba_var.tar
root@resiliency-server:/var/lib# cd -
root@resiliency-server:/home/testuser/ResConfig# mv qpopper.config
/root/src/qpopper4.0.19/
root@resiliency-server:/home/testuser/ResConfig# mv BPS_Setup.pl
StartupServicesTmp.pl /
root@resiliency-server:/home/testuser/ResConfig# mv RESET_SQL
SetupResetScript.sh /root/
root@resiliency-server:/home/testuser/ResConfig# mv ResetScriptTmp.c
addUsers.pl /root/
root@resiliency-server:/home/testuser/ResConfig# mv dyn_content.c /root/
root@resiliency-server:/home/testuser/ResConfig# cd ..
root@resiliency-server:/home/testuser# rm -fr ResConfig
root@resiliency-server:/home/testuser# mv /usr/sbin/NetworkManager
/usr/sbin/NetworkManager.gone
```

Create the path /home/TEST for Samba shares. For example:

```
root@resiliency-server:/home/testuser# mkdir /home/TEST
root@resiliency-server:/home/testuser# chmod 777
/home/TEST
```

Create the startup and reset script. For example:

```
root@resiliency-server:/home/testuser# cd /
root@resiliency-server:/# perl BPS_Setup.pl

Answer the questions and the script will create the startup script
(/StartupServices.pl), the reset script (/root/ResetScript), and will update
"/etc/services", "/etc/inetd.conf", "/etc/nsswitch.conf",
"/etc/rc2.d/S99rc.local", "/etc/default/grub", and "/etc/hosts"
```

Add users for Sendmail and Qpopper. This process may take several hours. For example:

```
root@resiliency-server:/# perl
/root/addUsers.pl
*** ADDING USER: user1 ***
*** ADDING USER: user2 ***
*** ADDING USER: user3 ***
*** ADDING USER: user4 ***
...
*** ADDING USER: user9998 ***
*** ADDING USER: user9999 ***
*** ADDING USER: user10000 ***
root@resiliency-server:/#
```

Reboot server. For example:

```
root@resiliency-server:/# reboot
```

Creating a network configuration for a non-x86-64, non-VM Application Server:

The previous section titled described how to use the `BPSResiliencyConfigFiles.tar` file. You will need this file as it contains the custom configuration and data needed by the services.

BreakingPoint recommendeds that you obtain and build the following packages from their source using the following versions:

- Sendmail: 8.14.3
- MySQL: 5.1.41
- Samba: 3.4.7
- Apache2: 2.2.14
- QPopper: 4.0.19

Pre-built packages may contain customizations from the distrubution that you are using and some of the file formats and file names may differ from how BreakingPoint packages those files in the BPSResiliencyConfigFiles.tar file.

Note: Make sure to enable InnoDB when building or installing MySQL. For example, `--enable-plugins=all` will enable all plugins, including `InnoDB`, as an argument to /configure).

If the services listed above were built and installed from the source, many of the locations defined in step Install the BreakingPoint Resiliency Server Configuration files. For example: on page 909 will be similar. For example, if you configure MySQL to use `/etc/mysql/my.cnf` as its configuration (`contained in mysql_etc.tar`), MySQL will use `/var/lib/mysql` to find the database files.

Copy the appropriate configuration files from the archives (for example: `sendmail.tar`, `apache2.tar`, and `samba.tar`).

When installing Apache from source, the configuration file will be named `httpd.conf`. The archive file apache.tar uses the file `apache2.conf` as the main configuration file and `httpd.conf` as the user configuration.

Copy `apache2.conf` to `httpd.conf` to start Apache.

Apache needs the content for the flows that the Resiliency Test will use. They are contained in the `www.tar` file and need to be copied into the DocumentRoot directory (`/var/www` or `<ServerRoot>/htdocs`, for example). You will need to compile the `dyn_content.c` file and place it into the `cgi-bin` directory (`<DocumentRoot>/cgi-bin` or `/usr/lib/cgi-bin`, for example). The source installer (make install) will also copy `printenv` and `test-cgi` into this directory as well. You will also need to set proper permissions so that the user that is running the server process can run `dyn_content` (user `www`, for example).

Follow the procedures described in the previous section to install Qpopper. The `--enable-specialauth` option is not necessary if your operating system does not use shadow passwords. For example, you will not have an `/etc/shadow` file.

Set the server's hostname to `resiliency-server`. When you enter the hostname command, the system will reply with `resiliency-server` and the command `domainname` will return none.

If the server does not queue mail or requests that the client supply a domain name, disable recipient checking in the `/etc/mail/sendmail.cf` file by commenting out the line that says `Scheck_rcpt`. For example, change:

```
# call all necessary rulesets
Scheck_rcpt
# R$@ $#error $@ 5.1.3 $: "553 Recipient address
required"
```

-- to --

```
# call all necessary rulesets
#Scheck_rcpt
# R$@ $#error $@ 5.1.3 $: "553 Recipient address
required"
```

Use `inetd` or a similar process to allow the reset service to accept connections on port 5555. Use `ResetScript.c` to manage this process and cleanup and restart services. You may need to modify this file to some degree, depending on how and where your services are installed.

Disable DNS by removing dns from `/etc/nsswitch.conf`. For example:

```
hosts: dns
files
-- to --
hosts: files
```

Add `resiliency-server` and `clientnet` to your `/etc/hosts` file. For example:

```
127.0.0.1 localhost resiliency-server resiliency-server.resiliency-
server.com
172.26.0.10 localhost resiliency-server resiliency-server.resiliency-
server.com
10.10.10.38 localhost resiliency-server resiliency-server.resiliency-
server.com
100.0.0.5 clientnet clientnet.clientnet.com
```

Note: Review the script `BPS_Setup.pl` as a guide to update configuration files.

Add the users for Sendmail and Qpopper using step Add users for Sendmail and Qpopper. This process may take several hours. For example: on page 911 of the procedure above. Use the `addUsers.pl` script as a guide. This script uses the `adduser` utility on Linux and is very specific to that operating system. For BSD operating systems, use this script as an example of the `adduser` utility. For example:

```
#!/usr/bin/perl
for ( $user = 1; $user <= 10000; $user++ ) {
open(OUT, ">/root/TMPL2");
print "*** ADDING USER: user$user ***\n";
print OUT "user$user" . "::::::::" . "/home/user$user:/bin/sh:user$user\n";
close(OUT);
$cmd = "/usr/sbin/adduser -f /root/TMPL2 1> /dev/null 2> /dev/null";
$rc = system("$cmd");
if ( $rc != 0 ) {
print "ERROR ADDING user$user!!! [$rc]\n";
exit $rc;
}
}
```

Data Center Resiliency

The network configuration used to test a device classified as a Data Center will be composed of one IP address attached to the device's network.

When creating a network configuration for a Data Center, ensure that the device is connected to the BreakingPoint port reserved as Logical Interface 1 and that the entire blade is reserved.

To create a network configuration for a Data Center:

1. Log into the VMware ESXi client vSphere as an account that can import templates, edit configurations, and start VMs.
2. From the Ixia Support website, download the following file to the vSphere client PC, `BPSTemplateGenerator.ova`.
3. From vSphere client, go to Inventory.
4. Go to File in the top menu.
5. Select **Deploy OVF Template**.
6. Browse to the path containing the file BPSTemplateGenerator.ova on the vSphere client PC and click Next.
7. Click **Next** again. In this screen, give this instance a distinct name.
8. Continue clicking **Next**, accepting the default options.
9. Click **Finish** to begin the VM import procedure.
10. Select the new VM and click **Edit Virtual Machine Settings** once the import is complete.
11. Click on **Network Adapter 1**. Select the network label that can access the VM Management network.
12. Click **OK** to close the window and accept the new settings.
13. Right-click on the newly created VM and select **Open Console**.
14. Click **Play** to start the VM.

When the VM completes its boot up process, you will see a screen with instructions.

Follow the on-screen instructuctions to generate a Data Center Deployment.

Resiliency Scoring a DUT

The table below lists the functions available for selection on the Resiliency Score page along with a description of each function.

Resiliency Score Functions

Function	Description
Select A Device	Allows you to select one of the predefined DUT categories.
Reserve Ports	Allows you to select the ports to be used for your test. You can reserve all the ports on a card for a ResiliencyScore.
Network Configuration	Allows you to select the network configuration to be used for your test. You can select a predefined network configuration, or you can create a new configuration for your test.
Device Capacity	The target performance of the DUT. Test criteria such as offered bit rate and minimal performance criteria will be scaled automatically based on the claimed performance rate of the device.
Number of Pairs	Number of reserved port pairs to use for the Resiliency Score. You can use all the ports on a card, if you have reserved them.

Function	Description
Testing Categories	Full test – Allows you to run a full battery of tests, which could potentially take several hours. This test generates an official Resiliency Score for the device when the test is run with all four testing categories selected. If any testing category is deselected, the Full test will not generate a Resiliency Score for the device.
	Quick test – Allows you to run abbreviated versions of the same tests as the Full test option. The purpose of this brief test is to provide you with an indication of how the device will perform. This test does not generate an official Resiliency Score for the device.
	Throughput – Measures the link speed of a device.
	Sessions – Traffic is run using realistic traffic engineered to stress the device's limits with respect to the rate of session churn.
	Robustness – Measures the ability of a device to correctly handle malformed traffic at different IP layers.
	Security – Measures the ability of a device to continue passing traffic when confronted with malicious traffic.
MaxTimeoutPerStrike	This Security component option defines how long to allow any single Strike to sit idle.
BehaviorOnTimeout	This Security Component option defines the behavior that occurs when a Strike has timed out. Options are "blocked" or "skipped".
Report Name	Allows you to specify a name for a particular test. This name will be attached to any subsequent report output. If this name needs to be re-entered subsequently in any screen, it will be available in a drop-down list.

You can select a single testing category or any combination of the available categories. You may not wish to perform all possible tests at any given time; therefore, you can multi-select the subset of tests desired. If you select a subset of tests, you must indicate this in the final report. No final or partial score will be given, and any omitted tests will be indicated as Test Not Completed. Standard test results will still be available.

To test the resiliency of a device:

1. Select **Test > Resiliency Score** from the Menu bar.
2. Select a device from the Select A Device area.

Note: The Network Neighborhood interface settings have been pre-configured for each type of network device. Be sure to configure the IP address of the device you are testing to match the corresponding IP address found in Resiliency Scoring IP Addresses on page 903.

3. Click **Choose Your Ports** from the Reserve Ports area.
4. Select the ports you want to use in your test. You can reserve all the ports on a card for a Resiliency Score. Click **Close** after you have selected your ports.
5. For Application Server and Data Center tests, select an existing network configuration, or create a new network configuration from the Network Configuration area. For all other tests, skip this step and go to step Select the capacity of your device from the Device Capacity area. below.

Select the capacity of your device from the Device Capacity area.

- For Application Server and Data Center tests, also select the type of virtual machines to be included in your test.

6. For Application Server tests, select At Least to find the limit of a device. When you select At Least, the test runs until it fails, usually within a short period of time. Select Exactly to test the device capacity for a specific number of users. When you select Exactly, the test runs until it is complete. Tests run with an exact number of users tend to run longer than those run with a minimum number of users selected.
7. In the **Number of Pairs** field, select the number of reserved port pairs you want to use for the Resiliency Score. You can select any number of reserved port pairs.
8. Select the categories you want to include in your test from the Testing Categories area. For Application Server and Data Center tests, skip this step and go to step Enter a name for your report in the Report Name field. below.

Note: If any testing category is deselected, the Full test will not generate a Resiliency Score for the device. The Quick test does not generate an official Resiliency Score for the device.

9. Enter a name for your report in the **Report Name** field.
10. Click **Validate**. The Connection Checklist will be displayed.
11. Verify that the ports listed on the Connection Checklist are reserved for your test. Click **Continue**. The Validation screen will be displayed.
12. Click **Run Test** once the Validation screen has completed. The progress of your test will be displayed by the device report page.
13. Once the test has completed, the Resiliency Score page will be displayed. Click **BLANK** to view the report.

Report Generation

When a Resiliency Score is completed, a test report that shows both summary information and more detailed information about subtests is generated. The report includes the OS and ATI in use, and any other information needed for another evaluator to reproduce the test scenario and obtain identical results. The test/report name is included in the report.

Note: Sections C and G of the Resililency report run the same test and share the same report.

Lawful Intercept General Information

Lawful Intercept systems facilitate detecting and capturing a few specific information flows out of a large field of untargeted flows without introducing performance degradation. Typically, these targeted

flows are made up of email traffic.

The Lawful Intercept test lab allows you to generate realistic email traffic. You can specify criteria such as ranges of user accounts and email keywords. You can also generate emails that contain random strings denoting realistic credit card or Tax Identification numbers.The Lawful Intercept test lab enables you to easily construct a scenario with configurable email traffic (with both random and specific keywords) and real-world background traffic.

Lawful Intercept Test Editor

The Lawful Intercept test editor allows you to compose the type of traffic you want to use in your lawful intercept test. The left portion of the Lawful Intercept test editor allows you to configure the background traffic and network settings for the test. The right portion allows you to configure the targeted Super Flows (a group of flows). In the targeted Super Flows, you will create a template for the patterns you want the DUT to search for. The patterns that the DUT will search for are referred to as needles (or triggers).

Note: Needles will be encoded into the protocol specified by their Super Flow. This means that needles will not always appear on the wire exactly as they are entered. However, the Lawful Intercept DUT should be capable of locating and decoding the needles.

The table below lists the fields for the Lawful Intercept test lab.

Lawful Intercept Test Lab Fields

Field	Description
Device Under Test	Select the device to test
Network Neighborhood	Select the Network Neighborhood to be used in your test
Background Traffic	Select the Application Profiles to be used in your test
Flows Per Second	Sets the number of flows per second for both background traffic and targeted Super Flows
Concurrent Flows	Sets the number of concurrent flows to be generated in your test for both background traffic and targeted Super Flows
Data Rate	Sets the maximum speed at which traffic is to be transmitted to the device for both background traffic and targeted Super Flows
Test Duration	Specify the test duration in time (hh:mm:ss)
Active	Select this field to make a target Active
Super Flow	Select the Super Flow to be used in your test
Type	Select the type of pattern (Targeted Flows)

Field	Description
Type	Select a sub-type based on the Type selected above (this option does not appear for all types)
Custom Delimiter	Available when the "File of user defined entries" Type is selected Enter the delimiter that will be used if New Line is not the delimiter
Filename (Only available when the "File of user defined entries" Type is selected)	Browse and select (or import) a file
Delimiter Type (Only available when the "File of user defined entries" Type is selected)	Select New Line or Custom as the delimiter type
Pattern (Only available when the "File of user defined entries" Type is selected)	Enter the pattern you are searching for in your test
Quantity	Indicate the number of random patterns (that conform to the format of the selected Type) that will be generated
Every	Use this option to indicate how frequently the pattern you are searching for appears in your test based on time (hh:mm:ss)
Every	Use this option to indicate how often the pattern you are searching for appears in your test based on a number of flows

Targeted Flows

A targeted flow represents the flow that contains the item that you want the Lawful Intercept device to search for. The objective of the test is to see whether your lawful intercept device can identify and capture the targeted flow from among a number of untargeted flows.

The table below lists the types of triggers (or needles) that can be used in your tests.

Lawful Intercept Type Triggers

Item	Description
Phone Number	Randomly chosen phone numbers

Item	Description
Tax Identification Number	Realistic Tax Identification Numbers (subtypes include SSN or EIN)
Credit Card Number	Realistic Credit Card Numbers (subtype include major credit cards)
User defined pattern	Enter that pattern that will be searched for during your test
File of user defined entries	A file of user defined entries that will be searched for during your test (Files containing entries are saved on a drive and then imported into BPS) See Creating User Defined Entries on the next page
List of user defined entries	A list of user defined entries (created in the GUI) that will be searched for during your test. See Creating User Defined Entries on the next page

Creating a Lawful Intercept Test

To create a Lawful Intercept Test, you will need to identify the target you are looking for. You should also create traffic that is similar to the item you are looking for, along with traffic that is not so similar.

To create a Lawful Intercept Test:

1. Select the **Labs** button from the Home page.
2. Select the **Lawful Intercept** option.
3. Click the **Browse** button next to the **Device Under Test** field and select the device you want to test.

   Note: When searching for an item in the Lawful Intercept test editor, type a portion of the item name into the Search field and click **Search**.
4. Click the **Browse** button next to the **Network Neighborhood** field and select a Network Neighborhood.
5. Click the **Browse** button next to the **Background Traffic** field and select the Application Profile that has the type of background traffic that you want to appear in your test.
6. Enter the number of **Flows Per Second** that you want in your test.
7. Enter the number of **Concurrent Flows** that you want in your test.
8. Configure the **Date Rate** in Mbps.
9. Enter the **Test Duration** (hh:mm:ss).
10. On the right side of the window, select the **Target** checkbox.
11. Click the **Browse** button next to the **Super Flow** field and select a Super Flow to include in your test.

    Note: If a Super Flow containing a target (or trigger) is not available, you will receive an error message. To resolve the error, go to the Application Manager and build a Super Flow that contains a trigger.

12. Select the type of trigger you want to include in your test by making a selection from the **Type** drop-down list.
    a. See Lawful Intercept Test Lab Fields on page 917 for a description of the Lawful Intercept Test Lab fields.
    b. See Lawful Intercept Type Triggers on page 918 for a description of the Lawful Intercept Type Trigger field.
    c. See Creating User Defined Entries below for instructions on how to configure a List of user defined entries.
13. You can add up to two additional targeted flows.
14. Click **Save** to save your test, or click **Save As** to save your test under a different name.
15. Click **Run** if you want to run your test.

> **Note:** The **Run** button may be grayed out if the **Test Status** does not display a green checkmark. If the **Test Status** displays a red "x", click **Test Status** to see the errors that are preventing the test from starting.

> **Note:** A combination of Lawful Intercept settings resulting in more than 1 flow per second will generate the following error message, "Error - The combination of settings supplied would result in a trigger rate of 10.0 per second for trigger (needle). The maximum is 1 per second".

Creating User Defined Entries

BreakingPoint generates random values for most of the Lawful Intercept Types, however you have custom options that can be implemented using lists. BreakingPoint will randomly select items from the lists you define and place them into your test.

To create a List of user defined entries:

1. Select the **List of user defined entries** option in the **Type** drop-down list.
2. Click the **Add** link to define your trigger.
    a. Type your pattern inside the field that appears.
    b. Click **Add** again if you wish to add additional triggers.
3. Configure the remaining settings, see Lawful Intercept Test Editor on page 917
4. Click **Save** to save your test, or click **Save As** to save your test under a different name.

To use a File of user defined entries:

> **Note:** You can create a text file or .csv file to list your entries with a new line delimiter or custom delimiter.

1. Select **File of user defined entries** from the Type drop-down list.
2. Enter a delimiter in the **Custom Delimiter** field (based on the delimiter that exists in the file).
3. In the Filename field, **Browse** and select a file from the list.
    a. Select a file that has a new line delimiter or a known delimiter that can be defined as a Customer Delimiter.
    b. If the file you want to use is not listed, click Import and select the file.
4. Select the **Delimiter Type** (Custom or New Line).

5. Configure the remaining settings, see Lawful Intercept Test Editor on page 917
6. Click **Save** to save your test, or click **Save As** to save your test under a different name.

Multicast General Information

Multicast is the delivery of a message to a group of destination computers simultaneously. The Multicast feature is designed to emulate streaming media environments. As opposed to sending a separate copy of the data to each host, the server sends the data only once. Routers along the way to the clients make and send copies as needed. BreakingPoint emulates Multicast clients and servers for performance testing of external clients, servers, and routers.

The Multicast functionality of BreakingPoint includes:

- The ability to measure Join/Leave latency
- Automatically generating traffic that supports IGMP versions 1, 2, and 3 (per RFC 3376)
- Interoperating with older versions of IGMP
- The ability to run most UDP applications over Multicast
- Working with standard Multicast routing infrastructures such as PIM-DM, PIM-SM, BIDIR-PIM, and SSM

The Multicast Test Lab is designed to simplify the setup of multicast tests. However, there are some configurations that cannot be created using the lab. When these situations occur, you will be required to create the test manually as described in the section titled Manually Creating a Multicast Test on page 927.

Multicast Test Lab Page Overview

From a single page, the Multicast Test Lab page allows you to define all of the sources and subscribers to be used in your Multicast test. With the Multicast Test Lab page, you can create up to 10 Multicast servers with associated IP addresses and Multicast groups and streams. The Multicast Test Lab page consists of the following five configuration sections:

- Sources below
- Network Template on the next page
- Subscribers on page 924
- Reset To Defaults on page 925
- Test Duration on page 925

Sources

The Sources section allows you to define the sources that will generate UDP multicast data streams. The test lab supports up to 10 sources. You can specify any IP address to transmit from, however, each source is configured on a /24 network in separate VLANs starting with 10 and incrementing by 1. So if you configure sources 1 and 2 with IP address 10.1.1.1, the test will generate UDP multicast traffic on VLAN 10 with IP address 10.1.1.1, and on VLAN 11 with IP address 10.1.1.1.

Other things to be aware of regarding the Multicast Test Lab Sources section:

- Each source runs in its own component
- Sources transmit from 1 to 10,000 Mbps
- Each source runs the Multicast Lab - Server Super Flow, which has single Raw Multicast flow with the following actions:
- Send Random Data (min 1000 bytes)
- Send Random Data (min 1000 bytes)
- Send Random Data (min 1000 bytes)
- Goto Action #1

Note: To avoid encountering these restrictions, you can create a multicast test manually by using the instructions described in the section titled Manually Creating a Multicast Test on page 927.

Multicast Source Fields

The table below lists the fields for the Sources section of the Multicast Test Lab page.

Multicast Sources Fields

Field	Description
IP Address	The source IP address.
Multicast Group	The destination IP address of the group to be joined.
Rate	The transmit rate.

Network Template

Each server runs on a /24 network in its own VLAN. The first server uses VLAN 10, the second uses VLAN 11, and so on. Any /24 network can be used as long as it does not overlap with the client networks. This means that it cannot fall within the range between 10.10.2.1 and 10.10.65.254.

The Network Template defines the VLANs and subnets that will be used by the subscribers (clients) in your test. This section has no effect on the sources. Three network templates are available:

- Small Network
- Medium Network
- Large Network

The Small Network template uses 16 VLANs beginning with VLAN 102 through VLAN 117. The table below lists the VLANs and subnets used by the Small Network template.

Small Network Template

VLAN	Subnet	Host Addresses Used
102	10.10.2.0/28	10.10.2.2 – 10.10.2.14
103	10.10.2.16/28	10.10.2.18 – 10.10.2.30

VLAN	Subnet	Host Addresses Used
104	10.10.2.32/28	10.10.2.34 – 10.10.2.46
105	10.10.2.48/28	10.10.2.50 – 10.10.2.62
…	…	
116	10.10.2.224/28	10.10.2.226 – 10.10.2.238
117	10.10.2.240/28	10.10.2.242 – 10.10.2.254

The Medium Network template uses 256 VLANs beginning with VLAN 102 through VLAN 357. The table below lists the VLANs and subnets used by the Medium Network template.

Medium Network Template

VLAN	Subnet	Host Addresses Used
VLAN	Subnet	Host Addresses Used
102	10.10.2.0/28	10.10.2.2 – 10.10.2.14
103	10.10.2.16/28	10.10.2.2 – 10.10.2.30
104	10.10.2.32/28	10.10.2.2 – 10.10.2.46
105	10.10.2.48/28	10.10.2.2 – 10.10.2.62
…	…	
356	10.10.17.224/28	10.10.2.2 – 10.10.17.238
357	10.10.17.240/28	10.10.2.2 – 10.10.17.254

The Large Network template uses 1024 VLANs beginning with VLAN 102 through VLAN 1125. The table below lists the VLANs and subnets used by the Large Network template.

Large Network Template

VLAN	Subnet	Host Addresses Used
102	10.10.2.0/28	10.10.2.2 – 10.10.2.14
103	10.10.2.16/28	10.10.2.2 – 10.10.2.30
104	10.10.2.32/28	10.10.2.2 – 10.10.2.46
105	10.10.2.48/28	10.10.2.2 – 10.10.2.62
…	…	

VLAN	Subnet	Host Addresses Used
1124	10.10.65.224/28	10.10.2.2 – 10.10.65.238
1125	10.10.65.240/28	10.10.2.2 – 10.10.65.254

Note: The range of addresses between 224.0.0.0 and 224.0.0.255, inclusive, is reserved for the use of routing protocols and other low-level topology discovery or maintenance protocols, such as gateway discovery and group membership reporting. Multicast routers should not forward any multicast datagram with destination addresses in this range, regardless of its TTL.

Subscribers

The Subscriber section allows you to define the subscriber (client) profiles to be used in your test. The Multicast Lab supports up to 10 subscriber profiles. Each subscriber profile has the following parameters:

- Max per Subnet allows you to define how many subscribers from this profile will be active on each VLAN/subnet defined in the Network Template.
- Multicast Group allows you to determine which group these subscribers will try to join.
- Source Specific allows you to select the source addresses to be used for source-specific multicast (leave unchecked for any source).

Each subscriber uses the Multicast Lab - Client Super Flow, which has a single Raw Multicast flow with the following actions:

- Join
- Delay 15 seconds
- Leave
- Delay 30 seconds

All subscriber profiles are run in the same test component with max concurrent sessions and max sessions/sec set to a value that corresponds to the size of the network used in the test. The table below lists the maximum number of sessions and the maximum sessions per second for each type of network.

Maximum Sessions and Maximum Sessions Per Second

Parameter	Small Network	Medium Network	Large Network
sessions.max	5,000	50,000	150,000
sessions.maxPerSecond	5,000	50,000	150,000

Multicast Subscriber Fields

The table below lists the fields for the Subscriber section of the Multicast Test Lab page.

Multicast Subscriber Fields

Field	Description
Max per Subnet	The maximum number of clients issuing join requests.
Multicast Group	The IP address of the group to be joined.
Source Specific	Allows you to define a specific Multicast SSM Source Address.

Max per Subnet Field

The Max per Subnet field determines the maximum number of clients that will join to the given group on any VLAN. For exmaple, a test configured with the following two subscribers will have at most 10 joins to group 239.0.0.1 on each VLAN:

- Subscriber profile 1:
  - Max per Subnet: 5
  - Multicast Group: 239.0.0.1
  - Source Specific: yes
- Subscriber profile 2:
  - Max per Subnet: 10
  - Multicast Group: 239.0.0.1
  - Source Specific: no

Profile 1 will have between 1 and 5 and will be source-specific joins. Profile 2 will have between 1 and 10 source-any joins. For example, one VLAN might have 3 source-specific and 7 source-any joins, while another VLAN might have no source-specific joins and 10 source-any joins.

Reset To Defaults

Selecting the Reset to defaults option resets all Multicast Test Lab settings to their default values.

Test Duration

The Test Duration setting allows you to define the duration of a multicast test.

Creating a Multicast Test With the Test Lab

Please note that conducting long-running Multicast Lab tests can consume up to 1,000 times more database capacity than other tests. Database functionality becomes severely limited once capacity exceeds 90 percent. It is important to optimize the database before it reaches 90 percent capacity.

To create a Multicast test with the test lab:

1. Select Test>Multicast from the Menu bar; or, click the Labs button, then click the Multicast button.
2. Enter the source IP address in the IP Address field.
3. Enter the IP destination address of the group to be joined in the Multicast Group field.

4. Enter the transmit rate in the Rate field. To add more sources to the test, click the '+' in the upper right-hand corner of the area. Click the '-' next to a source to remove it from the test.
5. Enter the maximum number of clients issuing join requests in the Max per Subnet field.
6. Select the IP address of the group to be joined from the Multicast Group drop-down list.
7. Select the Source Specific check box to define the Multicast SSM Source Address. Select this to use a list of known source addresses to use as a source include filter. To use an Include Any Source filter, do not select this option.

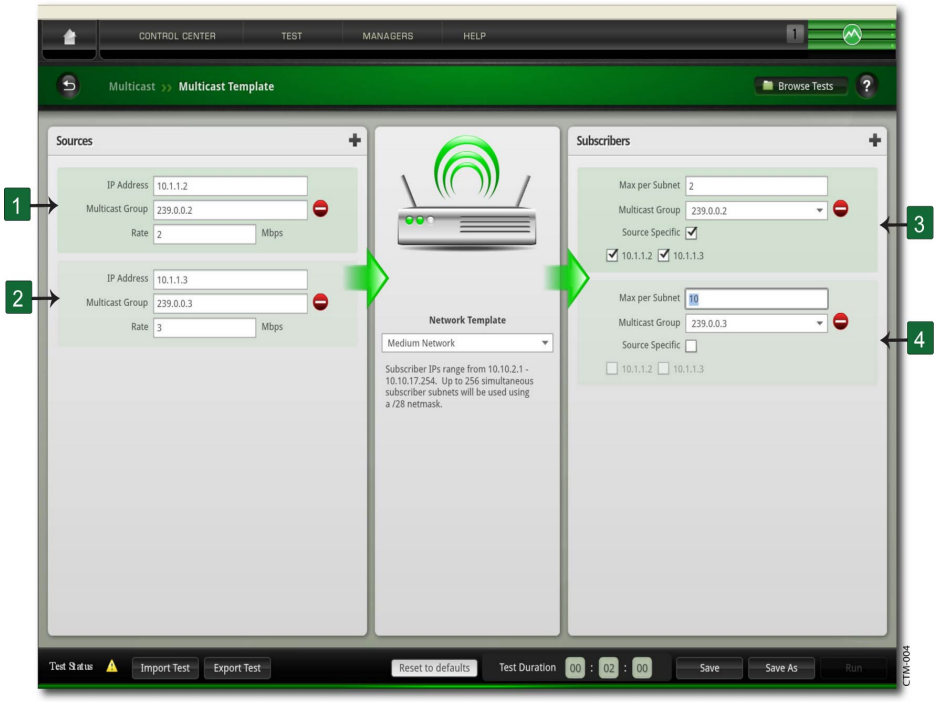
Note: To add more subscriber groups to the test, click the '+' in the upper right-hand corner of the area. Click the '-' next to a subscriber group to remove it from the test.

Multicast Test Lab Example

In this section, we examine a Multicast Test Lab test. This test creates two server components and one client component. It uses the Medium Network template and creates 256 VLANs for the client component.

Multicast Test Lab Example below provides an example configuration for a test using the Multicast Test Lab page and lists the test settings.

Multicast Test Lab Example



The following table identifies items on the Multicast Test Lab and provides a description of how the settings operate within the structure of a multicast test.

Multicast Test Lab Example

Item	Description
1	The first server component will generate a 2 Mbps UDP stream on VLAN 10 with IP source address 10.1.1.2 and IP destination address 239.0.0.2.
2	The second server component will generate a 3 Mbps UDP stream on VLAN 11 with IP source address 10.1.1.3 and IP destination address 239.0.0.3.
3	The client component will contain two Super Flows. The first client Super Flow will issue source-specific joins to group 239.0.0.2 with source 10.1.1.2. At most, 2 of these joins will be seen on any one VLAN at a time. Since there are 256 available VLANs, you will see 512 active joins from this profile at any point in time.
4	The second client Super Flow will issue source-any joins to group 239.0.0.3. At most, 10 of these joins will be seen on any one VLAN at a time. Since there are 256 available VLANs, you will see 2560 active joins from this profile at any point in time.

Manually Creating a Multicast Test

The recommended Multicast test setup uses one Application Simulator component for Multicast servers and a separate Application Simulator component for Multicast clients. Configuring servers and clients in separate components allows accurate control of the number of server flows.

A typical Multicast test uses a fixed number of servers for each Multicast group. If servers and clients are in the same component, the number of server flows are approximately determined by the simultaneous session count and the application profile weights of the component.

Note: When running bandwidth tests, Multicast Super Flows may appear to consume a disproportionate amount of bandwidth relative to the amount of traffic that they generate. When mixing Multicast client Super Flows with non-Multicast client Super Flows in an application profile, make sure to select Weight According to Flows. Additionally, be sure to set the weight of the Multicast client Super Flows much lower than the weight of the non-Multicast client Super Flows.

Multicast Server Super Flows

To create the Multicast server Super Flow:

1. Create a new Super Flow.
2. Add one flow with Multicast as the protocol for each Multicast group in the test.
3. Edit the settings for each flow and configure the following Multicast Flow Settings parameters:
   - Multicast Role = Server / Source
   - Multicast Group Address = (this will be the IP destination address)
   - Client Port = 0 (unless you want to configure an explicit UDP destination port)
   - Server Port = 0 (unless you want to configure an explicit UDP source port)

Note: By default, each server flow within this Super Flow will have the same server IP address.

If the servers require distinct addresses:

1. Click on Manage Hosts
2. Add additional servers
3. Assign the additional servers to the individual flows
4. Add flow actions.

The typical Multicast server flow will have one or more Send actions followed by a Goto action to repeat the flow. If there are multiple servers in the flow, be sure to include an explicit Close action for each flow after the Goto action. This prevents the automatic closing of flows that occurs on the last action of a flow.

Multicast Client Super Flows

To create the Multicast client Super Flow:

1. Create a new Super Flow.
2. Add one flow with Multicast to the Super Flow for each Multicast group you want to join.
3. Edit the settings for each flow and configure the following Multicast Flow Settings parameters:
   - Multicast Role = Client / Subscriber
   - Multicast Group Address = set to the group to be joined
   - Multicast SSM Source Address = Set to a comma-separated list of up to four IP addresses to define a list of source addresses to use as a source include filter. An empty list indicates an Include Any Source filter. (Explicit configuration of exclude filters is not supported.)
   - Multicast Max Clients Per Subnet/VLAN - A convenient way to control the number of clients issuing join requests is to set this value to the number you want on each VLAN and set the component's value of Max Simultaneous Sessions to value higher than the aggregate number of clients in the test.
   - Multicast Max Measurable Leave Latency - Leave latency is measured as the time between a leave request for a given multicast stream and the time the last UDP packet for that stream is received. If accurate measurements for leave latency are required, set this parameter to a value that exceeds the expected latency.
   - Client / Server Ports - These parameters are not used for multicast clients.

Note: By default, each client flow within this Super Flow will have the same client IP address.

If the clients require distinct addresses:

- Click on Manage Hosts
- Add additional clients
- Assign the additional clients to the individual flows.
- Add flow actions.

The typical Multicast client flow is a loop with delays after each Join and Leave action. The Join and Leave actions do not wait for success, they simply notify the BPS Multicast layer of the request and move on. The delay that follows a join should exceed the expected join latency. Similarly, the delay following a leave should exceed the expected leave latency. If there are multiple clients in one flow,

be sure to include an explicit Close action for each flow after the Goto action. This prevents the automatic closing of flows that occurs on the last action of a flow.

Multicast Action Parameters

Multicast allows routers to work together to efficiently deliver copies of data to interested receivers. Instead of sending a separate copy of the data to each host, the server sends the data only once. Routers along the pathway to the clients make copies as needed.

Multicast Action Parameters below lists the action and action parameters for Multicast.

Multicast Action Parameters

Action	Description	Action Parameters	Valid Values
Client: Multicast Join	Causes the flow to request a join to the multicast group defined in the flow settings.	Transaction Flag	Start, Continue, End, or Start and End
Client: Delay	Pauses the flow for a specified amount of time.	Transaction Flag	Start, Continue, End, or Start and End
		Number of Milliseconds. Number of milliseconds. If this check box is left unchecked, or if a value is not specified, the Application Manager will generate a random delay value of between 1000 and 4999 milliseconds.	1 – 1,000,000
		Maximum Number of Milliseconds	1,000,000
Client: Multicast Leave	Causes the flow to request a join to the multicast group defined in the flow settings.	Transaction Flag	Start, Continue, End, or Start and End
Server: Send Random Data	Sends randomized data.	Transaction Flag	Start, Continue, End, or Start and End

Action	Description	Action Parameters	Valid Values
Client: Goto	Causes the flow to go to another action	Transaction Flag	Available Actions
		Goto Action	
		Iterations	

Multicast Real-Time Statistics

There are no real-time statistics dedicated to Multicast. This section contains tips on using existing Real-Time Statistics tabs to monitor a Multicast test in progress. The behavior described here assumes the test contains only Multicast client and server flows.

The only Real-Time Statistics tabs that show data related to Multicast tests are the Summary Tab and the Interface Tab.

Real-Time Statistics Summary Tab

Concurrent Super Flows:

This counter shows the total number of Super Flows. It should match the number of servers plus the number of VLANs * max clients/VLAN. For example, if you have 2 servers and 8 VLANS with 4 clients per VLAN, the number of Concurrent Super Flows should equal 34.

Concurrent UDP flows:

This counter displays the total number of server flows. Subtract this value from the number of Concurrent Super Flows to obtain the total number of active clients.

Transactions:

By default, the transaction data will count the number of UDP packets transmitted by Multicast servers. If you explicitly set the transaction flags on Super Flow server actions, they count operations according to how you set the flags. Transaction flags on client flows are ignored.

TCP:

Multicast does not use TCP. All TCP counters should equal zero.

Interface Stats:

These counters show ingress/egress packets and includes all IGMP and UDP packets.

Real-Time Statistics Interface Tab

Transmit Stats:

These counters show server generated UDP packets and client generated IGMP packets.

Receive Stats:

These counters show IGMP and UDP packets received and are perhaps the most interesting real-time counters for Multicast tests. If the number of IGMP packets is small in comparison to UDP (which is usually the case), these counters show which interfaces are receiving Multicast streams. If client flows are configured to "Join; Delay; Leave; Delay; Repeat;" and if the test and network is setup and running correctly, you will see the receive rates go up and down as clients join and leave Multicast groups.

Interpreting Test Results Section by Section

Multicast statistics are not presented in consistent manner with regard to clients and servers. The general rules to keep in mind when looking at Multicast test reports are that Multicast server flows:

- Transmit only UDP packets
- Record transmit statistics at the Application layer under the Multicast application protocol
- Record transmit statistics at IP and UDP layers
- Never receive packets
- Multicast client flows:
  - Transmit only IGMP packets
  - Do not record statistics at the Application layer
  - Record transmit statistics at IP and IGMP layer

Long Term Evolution General Information

Long Term Evolution (LTE) is the next generation mobile telecommunications network standard developed by the Third Generation Partnership Project (3GPP), an industry trade group. LTE networks enable fixed-to-mobile migrations of Internet applications such as Voice over IP (VoIP), video streaming, music downloading, mobile TV and many others. Additionally, LTE networks provide the capacity to support increased demand for connectivity from consumers with devices tailored to new mobile applications.

With the LTE Test Lab, you can test your LTE devices by emulating a mobile telecommunications environment complete with mobile phone users of various types, connecting cell towers, and a variety of services. The LTE Test Lab provides the ability for emulated user equipment (UE) to contact external servers for data connections. You also have the ability to assign the profile of a given UE to a group of UEs.

LTE Test Lab Page Overview

The LTE Test Lab page is where you define the devices and connections of your test, add and name the equipment to be tested, and define the Internet services that will be used in your test.

Simulated Elements

This area of the LTE Test Lab allows you to define the devices and the connections that will be used in your test.

Simulated ElementsFields

Field	Description
Number of UEs	The total number of devices to simulate.
Bearer(s)	The path over which a UE sends and receives data via the PDN.
Starting IMSI	The first in a sequential range of IMSI numbers to be used in the test. The IMSI identifies the SIM card of each device.
APN	The type of network connection to create.
Bandwidth	The amount of bandwidth to be used in the test.
Hosts Per Second	An upper limit on the number of new UEs per second. A value of zero (0) represents the fastest attachment rate possible.
Starting Secret Key	The base value for a secret key that is generated for each UE.
Secret Key Step	The value that the Secret Key is incremented by for each UE.
Operator Variant	Specifies a unique value originally assigned by the UE manufacturer. The operator variant is usually unique to each brand of UE.
Application Profile	The mix of application traffic used by the UEs in the test.
Number of eNodeBs	The number of cell towers used in the test.
Network Address	The network address of the subnet where all of the eNodeBs will be located.
Netmask	The netmask address of the subnet where all of the eNodeBs will be located.
Gateway	The default gateway that each eNodeB will be configured with.
Starting IP	The first IP address that the eNodeBs will be given.
DNS Server IP	Provides the address of the DNS to use when resolving hostnames.
Domain Name	The default domain name to use for the given hostnames.

Equipment to Test

This area of the LTE Test Lab allows you to add the mobility management entity (MME) that you are testing to the test.

Equipment to TestFields

Field	Description
MME	The device to be tested.

Public Land Mobile Network

The Public Land Mobile Network (PLMN) is a unique identifier for each cellular network provider. It consists of the Mobile Country Code (MCC) and the Mobile Network Code (MNC). The MCC is determined by the country. For every MCC, each cellular network provider can register for one or more MNC. You can specify an MCC and MNC so that they match the MCC and MNC being used by the device under test.

Public Land Mobile Network

Field	Description
MCC	The Mobile Country Code of the device to be tested.
MNC	The Mobile Network Code of the device to be tested.

Services

This area of the LTE Test Lab allows you to define the Internet services that will be used in your test.

Services Fields

Field	Description
Network Address	The network address of the subnet where the simulated Internet services will be located.
Netmask	The netmask address of the subnet where the simulated Internet services will be located.
Gateway	The default gateway setting for all simulated Internet services.
Starting IP	The first IP address that the Internet services will use.
Number of Hosts	The total number of separate simulated hosts that will be used to provide Internet services.
BPS IP	The IP address of your BreakingPoint system

Services Fields above lists the fields for the Services area of the LTE Test Lab.

Reset To Defaults

Selecting the Reset to defaults option resets all LTE Test Lab settings to their default values.

Test Duration

The Test Duration setting allows you to define the duration of an LTE test.

Creating an LTE Test

The following section provides instructions for creating an LTE test.

To create an LTE test:

1. Select **Test>LTE** from the Menu bar or click the **Labs** button, then click**LTE**.
2. Enter the number of UEs to be simulated in the Number of UEs field.
3. Enter the IMSI to begin with in the Starting IMSI field. The IMSIs will be added as subscribers on the HSS to be tested.
4. Enter the MSISDN to begin with in the **Starting MSISDN** field. The simulated UEs will be assigned a sequential range of MSISDN numbers beginning with the value entered here.
5. Enter the type of network connection to create in the **APN** field. This value determines the type of network connection will be simulated.
   a. Enter the amount of bandwidth to be used.
6. Click **Browse** to select the Application Profile to be used. When the **Browse for App Profiles** window is displayed, enter text in the search field to search for the Application Profile you want to select.
7. Enter the number of cell towers to be used in your test in the **Number of eNodeBs** field.

Note: Each eNodeB is configured to have three cells, each with a unique cell ID and TAC. The three cells within each eNodeB will be numbered sequentially. The maximum number of eNodeBs is 4096.

8. Enter the network address of the subnet where all eNodeBs will be located in the **Network Address** field.
9. Enter the netmask of the subnet where all of the eNodeBs will be located in the **Netmask** field.
10. Enter the default gateway that each eNodeB will be configured with in the **Gateway** field.
11. Enter the IP address to begin with in the **Starting IP** field. The simulated eNodeBs will be assigned consecutive IP addresses starting with the address entered.
12. If the MME is identified by a hostname, enter the DNS server IP address in the **DNS Server IP** field.
13. Type the default domain name to use for the given hostname in the **DomainName** field. This field is only required if a DNS server IP is used.
14. Type the name of the device to be tested in the **MME** field. To add more MMEs to the test, click the '+' in the upper right-hand corner of the area. Click the '-' next to an MME to remove it from the test.

15. The number of endpoint pairs generated in a test is based on the number of unique eNodeBs and MMEs configured. For example, if a test contains two unique eNodeBs and two unique MMEs, four endpoint pairs will be generated for that test.
16. Enter the subnet address of the Internet services in the **Network Address** field.
17. Enter the netmask of the subnet of the simulated Internet services in the **Netmask** field.
18. Enter the default gateway for all of the simulated Internet services in the **Gateway** field.
19. Enter the IP address to begin with in the **StartingIP** field. Internet services will use a contiguous set of IP addresses starting with the address entered.
20. Enter the total number of hosts that will be used in the test in the **NumberofHosts** field.
21. Click **Run** to run and save the test.

For tests measuring the UE attachment and detachment rate, when the rate is not set to 0 (unlimited), the application transaction will begin at the 5 second mark. The UE Attaches-Detaches/s field allows you to configure the rate at which retry intervals occur.

Device Validation Lab Overview

The Device Validation Lab allows you to remotely measure and demonstrate the performance of network devices.

The Device Validation Lab consists of two components, the Device Validation Manager and the Device Validation Tool.

Device Validation Manager

The Device Validation Manager allows you to set up a newly created test and send it to a remote BreakingPoint device in preparation for a demonstration. You can also add network devices to your demonstration. Once you have identified the tests to be used in your demonstration, you can map your demonstration information to the devices to be tested.

Demos Tab

The Demos tab allows you to add existing tests and create new tests to be used in your demonstration.

Demos Tab Elements below lists the elements of the Demos tab of the Device Validation Manager page.

Demos Tab Elements

Element	Description
Search Field	Allows you to search for available test files based on filename and author
Clear	Clears the Search Field
Search	Searches for available tests
Name	Name of the test

Element	Description
Author	Lists tests according to the name of the user that last modified the test
Description	User-defined description of the test
Edit	
Create New	Allows you to create a new demonstration
Clone	Allows you to replicate an existing demonstration
Export	Allows you to export an existing test into the Demo page

The Hardware Tab

The Hardware Tab allows you to add network devices to the demonstration.

Hardware Tab Elements below lists the elements for the Hardware tab of the Device Validation Manager page.

Hardware Tab Elements

Element	Description
Search Field	Enter criteria to search for the device you want
Clear	Clears the Search Field
Search	Searches for available devices
IP	IP address of the device being tested
Username	Name of the user
Hardware Status	Indicates whether the device is connected to the network
IP Address	Enter IP address of the device to be added to the test
Username	Enter user name
Password	Enter password
Verify	Displays new hardware information entered
Submit	Initiates demonstration

The Push Demos Tab

The Push Demos Tab allows you to map your demonstration information to the devices to be tested. Once this information has been mapped, the Device Validation Tool can be used to run the

demonstration.

Push Demos Tab Elements below lists the elements for the Push Demos tab of the Device Validation Manager page.

Push Demos Tab Elements

Element	Description
Search Field	Enter criteria to search for the test you want
Clear	Clears the Search Field
Search	Searches for available tests
Name	Name of the test
Get more results	Shows additional tests
Author	Username of the creator of the test
Created On	Date test was created
Add	Add test to the demonstration
Search Field	Enter criteria to search for the device you want
Clear	Clears the Search Field
Search	Searches for available device
Get more results	Shows additional devices
IP	IP address of the device being tested
Username	Username of the user
Hardware Status	Indicates whether the device is connected to the network
Add	Add device to the demonstration
Push Demos	Sends demonstration to the connected

Creating a new Device Validation Demonstration

The following section provides instructions for creating a new demonstration using the Device Validation Manager.

To create a Device Validation Demonstration:

1. Select Managers > Device Validation Manager from the Menu bar.
2. Click Create New.

3. Enter a name for the demonstration in the Demo Name: field.
4. Enter a description of the demonstration in the Description: field (optional).
5. Select the type of test you will be demonstrating from the Test type: field.
6. Enter search criteria to narrow your search for the test(s) you want to include in your demonstration in the Search Field and click Search.
7. Click the green plus sign (+) next to the test(s) you want to include in your demonstration.

   Note: Click the red X next to a test you want to remove from your demonstration.

8. Click Save and Exit once you have selected all of the tests to be included in your demonstration.

Cloning an existing Device Validation Demonstration

The following section provides instructions for cloning an existing demonstration using the Device Validation Manager.

To clone an existing Device Validation Demonstration:

1. Select Managers > Device Validation Manager from the Menu bar.
2. Select the demonstration you want to replicate and click Clone.
3. Enter a name for the demonstration in the New Demo Name: field.
4. Click Ok.

Device Validation Tool

The Device Validation tool allows you select tests that you have previously created and sent to a remote BreakingPoint device.

Note: BreakingPoint recommends verifying that the setup will work properly prior to running the demonstration at the customer site. To do this, BreakingPoint recommends running the test using the Device Validation Tool on the BreakingPoint system used to create the demonstration.

The following section provides instructions for running a demonstration using the Device Validation Tool.

To run a Demonstration:

1. Select Test > Device Validation from the Menu bar.
2. Select the device you want to test from the Select Hardware to Test column.
3. Select the demonstration to be run from the Select Test to Run column.
4. Click Run Test.

Multi-box Testing Overview

Multi-box testing allows you to concurrently run tests on up to five BreakingPoint systems. One system will act as the main system, which is the system that will be used as the management interface for all secondary systems.

The multi-box test will be created on the main system. The main system must contain the tests, Network Neighborhoods, DUT Profiles, Strike Lists, App Profiles, and capture files that the multi-box

test will use. All the data that will be used in a multi-box test will be copied from the main system onto the secondary systems. Any data on the secondary system that shares a name with data on the main system will automatically be overwritten.

The data that will be overwritten include:

- Tests
- Capture files
- Strike Lists
- DUT Profiles
- App Profiles
- Network Neighborhoods

Multi-box Requirements

When you create a multi-box test, you will need to know the IP addresses of the secondary systems as well as the authentication information for each system. The system will authenticate the login information for each system once you run the multi-box test.

Additionally, you must ensure that the Active Group assignments for the ports on the main system match up with the Active Group assignments on the secondary systems. For example, if Slot 1 and its ports on the main system are assigned to Active Group 1, but Slot 1 on the secondary system has its ports assigned to Active Group 2, the multi-box test will not run. Instead, when you attempt to run the test, the system will display an error stating that the secondary system's ports are already in use.

If you start a multi-box test while one of the secondary systems is running a test, the multi-box test will not run. Before running a multi-box test, check the status of each system to ensure that no other tests are being run. Additionally, if a secondary system loses its connection during a multi-box test run, all multi-box tests will stop running.

Administering Secondary Systems

The multi-box interface does not provide control over any administrative tasks; therefore, you will still need to log into each individual system to manage user accounts and update the system. Each system should still be managed and administered as an individual system and is subject to the same installation and configuration requirements as a single-box system. This means that each box should be connected to a computer system either through a serial connection or through a hub.

Port Reservations and Mapping for Secondary Systems

On the Multi-box Test Editor screen, there is a button called **Add Box**. This button allows you to identify secondary boxes for your multi-box test. When you enter the information into the Add New Multibox Test dialog box and click the Apply Change button, the **Device Status** screen for the secondary system will be displayed.

You can use this feature to map ports on the secondary systems. This is important because you will need to verify that all secondary devices have the same Active Group selected for its slots/ports as the main system.

For example, if Slot 1 and its ports are assigned to Active Group 1 on the main system, then Slot 1 on the secondary system must also have Slot 1 and its ports assigned to Active Group 1.

Expect Scripting

If you plan on using automation through Expect scripting, the DUT must be connected to the Target Control ports on the system. Device automation will be regulated by the main system. You can assign a different DUT Profile for each test; however, the DUT Profiles must be stored on the main system.

Note: All systems used in a multi-box test must all use the same firmware and ATI Update versions. BreakingPoint does not support forward or backwards compatibility for the multi-box functionality.

Static Routes

If you are utilizing systems that are on a different network (e.g., 1.1.0.0 and 192.16.123.0), you will need to disable DHCP for the main system and add a static route for the network outside of the main system's route.

To disable DHCP and add a static route:

1. Telnet to the primary system.
2. Enter the authentication information for the system.
3. Enter the command `updateNetwork -dhcp no`.
4. Enter `exit` to leave the telnet prompt.
5. Go to the Administration area of the Control Center.
6. Click on the **Routes** tab.
7. Click the **Host** radio button.
8. Enter the IP address for the secondary system that is located on an outside network in the **IP Address** field.
9. Enter the gateway address for the network in the **Gateway** field.
10. Click the **Add Route ('+')** button.

Reports

Once a multi-box test completes, a single report will be generated for all the tests that were run. Each system will store the results for its portion of the test in a single report. This works in the same way as it does for single-box tests.

A comprehensive report, or one that contains all the results from the multi-box test, will be stored on the main system. There will not be individual results for the main system's portion of the test results. If the reports related to the multi-box test are deleted from the secondary systems, the report from the main system will not be able to retrieve the data, so it will not be able to generate any results from the multi-box test.

Aggregate stats are not compiled for the multi-box test. Instead, only individual test results are reported for each system.

The following video link provides a tutorial on how to Review BPS reports:
https://www.youtube.com/channel/UCanJDvvWxCFPWmHUOOlUPIQ/videos

Note: You may not be able to view more than five multi-box test reports at a time. This limitation is a case-by-case situation, and it may be caused by limitations imposed by your browser.

Deleting Multi-box Reports

If you delete the multi-box report from the main system, the system will attempt to delete the related reports on the secondary systems. The system will attempt to log into the secondary systems using the authentication information stored for each system. If another user is logged directly into the secondary system using that authentication information, they will be logged out of the system.

Network Neighborhood Configuration

With multi-box testing, you can generate traffic from the same IP addressing pools. The addressing works the same was as it does with a single box; however, with multiple boxes, you will need to configure a range of addresses for all traffic transmitted/received on each interface.

Note: The range of addresses cannot overlap. For example, you cannot have a range of 1.0.1.1 – 1.0.1.254 for one interface and a range of 1.0.1.1 – 1.0.1.125 for another interface.

You can use a single Network Neighborhood for the entire test, if you configure multiple domains for each interface in the Network Neighborhood, and then assign them to server/client interfaces in the test.

If you prefer to use multiple Network Neighborhoods instead, you can assign a different Network Neighborhood to each test in a multi-box test as long as each Network Neighborhood is on the main system. The Network Neighborhoods used in the test will be copied over to the secondary systems, so any Network Neighborhoods that share the same names will automatically be overwritten.

Sample Network Neighborhood Configuration

Sample Network Neighborhood Configuration below provides a sample configuration of multiple Network Neighborhoods for a multi-box test. All of the interfaces use the default domain.

Sample Network Neighborhood Configuration

Device	Network	Interface 1	Interface 2	Interface 3	Interface 4
P1	NN1	1.0.1.0/8 range 1.0.1.1 – 1.0.1.254	1.0.2.0/8 range 1.0.2.1 – 1.0.2.254	1.0.3.0/8 range 1.0.3.1 – 1.0.3.254	1.0.4.0/8 range 1.0.4.1 – 1.0.4.254

Device	Network	Interface 1	Interface 2	Interface 3	Interface 4
S1	NN2	1.0.5.0/8 range 1.0.5.1 – 1.0.5.254	1.0.6.0/8 range 1.0.6.1 – 1.0.6.254	1.0.7.0/8 range 1.0.7.1 – 1.0.7.254	1.0.8.0/8 range 1.0.8.1 – 1.0.8.254
S2	NN3	1.0.9.0/8 range 1.0.9.1 – 1.0.9.254	1.0.10.0/8 range 1.0.10.1 – 1.0.10.254	1.0.11.0/8 range 1.0.11.1 – 1.0.11.254	1.0.12.0/8 range 1.0.12.1 – 1.0.12.254
S3	NN4	1.0.13.0/8 range 1.0.13.1 – 1.0.13.254	1.0.14.0/8 range 1.0.14.1 – 1.0.14.254	1.0.15.0/8 range 1.0.15.1 – 1.0.15.254	1.0.16.0/8 range 1.0.16.1 – 1.0.16.254

Creating a Multi-box Test

There are a few things you must do before you create a multi-box test:

1. Select the system that will be used as the main system.
2. Create the tests you want to run on the main system.
3. Set up the Network Neighborhoods, DUT Profiles, App Profiles, and Strike Lists that will be used on the main system.
4. Import any PCAP files that will be used for any Recreate tests to the main system.
5. Verify that the secondary systems do not share any names for tests, Network Neighborhoods, DUT Profiles, App Profiles, and Strike Lists with the main system.
6. Gather the authentication information for each system.
7. Verify that all secondary boxes are connected to a DUT.
8. Verify that all systems have the same ATI Update and firmware versions.

To create a multi-box test:

1. Select **Test > Multibox Testing** from the Menu bar.
2. Click the **New Test** button.
3. Enter a name for the multi-box test in the **Give the new multi-box a name:** field.
4. Click the **OK** button.
   a. A multi-box test will be created. It will contain a test that has already been set up with authentication and device information for the main system.
5. Click the **Add Box** button. The Add New Multibox Test dialog box is displayed.

6. Click on the **Browse** button located next to the **Test** field. The Browse For Test dialog box is displayed.
7. Select a test from the Browse For **Test** dialog box.
8. Click the **OK** button.
9. Click the **Accept** button.
10. Click the **Browse** button located next to the **DUT** field.
11. Select a DUT Profile from the Browse Device Under Test (DUT) dialog box.
12. Click the **OK** button.
13. Click on the **Browse** button located next to the **Network Neighborhood** field. The Browse Network Neighborhood dialog box is displayed
14. Select a Network Neighborhood from the Browse Network Neighborhood dialog box.
15. If you need to edit or create a Network Neighborhood or DUT Profile, and DUT Profiles on page 95.
16. Enter the IP address of the primary box in the **IP Address** field.
17. Enter your username in the **Username** field.
18. Enter your password in the **Password** field.
19. Click the **Apply Change** button.
20. Click th**e Save Multi-box** button.
21. Click the **Add Box** button. The Add New Multibox Test dialog box is displayed.
22. Click on the **Browse** button located next to the **Test** field. The Browse For Test dialog box is displayed.
23. Select a test from the Browse For **Test** dialog box.
24. Click the **OK** button.
25. Click the **Accept** button.
26. Click the **Browse** button located next to the **DUT** field.
27. Select a DUT Profile from the Browse Device Under Test (DUT) dialog box.
28. Click the **OK** button.
29. Click on the **Browse** button located next to the **Network Neighborhood** field. The Browse Network Neighborhood dialog box is displayed.
30. Select a Network Neighborhood from the Browse Network Neighborhood dialog box.
31. Enter the IP address of the secondary box in the IP Address field.
32. Enter your username in the Username field.
33. Enter your password in the Password field.
34. Click the **Apply Change** button.
35. Repeat steps Click the Add Box button. The Add New Multibox Test dialog box is displayed. above through Click the Apply Change button. above for any additional systems you want to add to the multi-box test.
    a. You can have up to five boxes in a multi-box test. Since you can only assign one test per box, you can run up to five tests concurrently using the multi-box feature.
36. Click the **Save Multi-box** button to save the test.

This page intentionally left blank.

CHAPTER 20 Reporting

This section covers:

Reporting Overview

Reports provide detailed information about the test, such as the components used in a test, the addressing information, the DUT profile configuration, the system versions, and the results of the test. All reports will include an aggregated test results section, which provides the combined statistics for all of the test components.

All reports will be automatically generated in HTML and viewable with a Web browser; however, you may export the test results in XLS, HTML, PDF, RTF, CSV, or ZIP (CSV files). For more information on exporting reports, see theExporting Reports on page 956section.

Reports are automatically generated each time a test is run and are viewable from the Results page. The reports can be sorted by clicking on any of the column headings (Result, Config Name, User, Duration, etc.) to easily locate reports.

Note:

- Reports generated from a test series will be listed under each individual test's name. There is not a consolidated report for a test series.
- When multiple ranges are defined on the same physical interface, the total number of received multicast packets reported by a test will be multiplied by the number of ranges defined.

The statistics contained within each report depends on the test components used within the test. For more information on the statistics available for each test component, see the Statistics Reported Per Component below.

Statistics Reported Per Component

The following sections list the statistics that are tracked and reported for each test component.

Bit Blaster Statistics

The following table lists the statistics reported for the Bit Blaster test component.

Bit Blaster Statistics

Statistic	Description
Frame Counts	The number of frames transmitted and received per second
Frame Rate	The rate (in fps) at which frames are transmitted and received per second
Frame Data	The number of bytes transmitted and received per second

Statistic	Description
Frame Data Rate	The data rate (in Mbps) at which frames are transmitted and received per second
Frame Latency	The frame latency (in microseconds)
Frame Size	The average frame size for transmitted and received frames
Transmitted Frames by Size	The size of transmitted frames
Received Frames by Size	The size of received frames

Routing Robot Statistics

The following table lists the statistics reported for the Routing Robot test component.

Routing Robot Statistics

Statistic	Description
Frame Counts	The number of frames transmitted and received per second
Frame Rate	The rate (in fps) at which frames are transmitted and received per second
Frame Data	The number of bytes transmitted and received per second
Frame Data Rate	The data rate (in Mbps) at which frames are transmitted and received per second
Frame Latency	The frame latency (in microseconds)
Frame Size	The average frame size for transmitted and received frames
Transmitted Frames by Size	The size of transmitted frames
Received Frames by Size	The size of received frames

Session Sender Statistics

The following table lists the statistics reported for the Session Sender test component.

Session Sender Statistics

Statistic	Description
TCP Concurrent Connections	The number of TCP sessions concurrently open at any given time

Statistic	Description
TCP Connection Rate	The number of TCP connections attempted and established per second
TCP Aggregate Connections	The total number of TCP connections attempted and established per second by the server and the client
TCP Average Time to Open	The average amount of time it takes each TCP connection to set up (in ms)
TCP Average Time to Response Packet	The average amount of time it takes for the response packet to be sent (in ms)
TCP Average Time to Close	The average amount of time it takes for a connection to close
Frame Counts	The number of frames transmitted and received per second
Frame Rate	The rate (in fps) at which frames are transmitted and received per second
Frame Data	The number of bytes transmitted and received per second
Frame Data Rate	The data rate (in Mbps) at which frames are transmitted and received per second
Frame Latency	The frame latency (in microseconds)
Frame Size	The average frame size for transmitted and received frames
Transmitted Frames by Size	The size of transmitted frames
Received Frames by Size	The size of received frames

Security Statistics

The following table lists the statistics reported for the Security test component.

Security Statistics

Statistic	Description
Strike Category Assessment	The number of Strikes that have been blocked by the device and the number of Strikes that have evaded detection
Strike Detection Assessment	The IP address and port of the Strike's origin and destination

Stack Scrambler

The following table lists the statistics reported for the Stack Scrambler test component.

Stack Scrambler Statistics

Statistic	Description
Transmitted Frames	The number of frames transmitted per second
Pings	The number of pings sent and received

Application Simulator

The following table lists the statistics reported for the Application Simulator test component.

Application Simulator Statistics

Statistic	Description
App Concurrent Flows	The number of concurrent UDP and TCP flows open at any given time
App Bytes Transmitted	The total number of bytes transmitted per protocol
App Bytes Received	The total number of bytes received per protocol
App Throughput	The transmitting and receiving data rate for each protocol
App Transaction Rates	The rate at which application transactions are set up
Exceptions	Exceptions received
Frame Counts	The number of frames transmitted and received per second
Frame Rate	The rate (in fps) at which frames are transmitted and received per second
Frame Data	The number of bytes transmitted and received per second
Frame Data Rate	The data rate (in Mbps) at which frames are transmitted and received per second
Frame Latency	The frame latency
Frame Size	The average frame size for transmitted and received frames
TCP Concurrent Connections	The number of concurrent TCP sessions open at any given time
TCP Connection Rate	The number of TCP connections attempted and established per second

Statistic	Description
TCP Aggregate Connections	The total number of TCP connections attempted and established per second by the server and the client
TCP Average Time to Open	The average amount of time it takes each TCP connection to set up (in ms)
TCP Average Time to Response Packet	The average amount of time it takes for the response packet to be sent (in ms)
TCP Average Time to Close	The average amount of time it takes for a connection to close
Transmitted Frames by Size	The size of transmitted frames
Received Frames by Size	The size of received frames
Router Discards	The number of packets received after the test has stopped
No RX Queue	Discarded packets received but not sent to the RX Queue

Recreate

The following table lists the statistics reported for the Recreate test component.

Recreate Statistics

Statistic	Description
Concurrent Flows	The number of concurrent UDP and TCP flows open per second
Bytes Transmitted	The total number of bytes transmitted per protocol
Bytes Received	The total number of bytes received per protocol
Throughput	The transmitting and receiving data rate for each protocol
Transaction Rates	The rate at which application transactions are set up
TCP Concurrent Connections	The number of concurrent TCP connections open at any given time
TCP Connection Rate	The number of TCP connections attempted and established per second
TCP Aggregate Connections	The total number of TCP connections attempted and established per second by the server and the client
TCP Average Time to Open	The average amount of time it takes each TCP connection to set up (in ms)

Statistic	Description
TCP Average Time to Response Packet	The average amount of time it takes for the response packet to be sent (in ms)
TCP Average Time to Close	The average amount of time it takes for a connection to close
Frame Counts	The number of frames transmitted and received per second
Frame Rate	The rate (in fps) at which frames are transmitted and received per second
Frame Data	The number of bytes transmitted and received per second
Frame Data Rate	The data rate (in Mbps) at which frames are transmitted and received per second
Frame Latency	The frame latency
Frame Size	The average frame size for transmitted and received frames
Transmitted Frames by Size	The size of transmitted frames
Received Frames by Size	The size of received frames

Aggregate Statistics

The following table lists the aggregate statistics for the test. Aggregate statistics will be tagged with the 'Ethernet' label, and they measure the results for traffic across all ports for all test components.

Aggregate Statistics

Statistic	Description
Ethernet Frames	Total number of frames transmitted and received from of all ports
Ethernet Frame Rate Stats	Total frame rate for all network traffic across all ports
Ethernet Data Rate Stats	Total data rate for all network traffic across all ports
Ethernet Data	Total number of bytes transmitted and received
Ethernet Errors	Total number of errors for all transmitted and received frames across all ports

Statistic	Description
Concurrent Flows	Total number of concurrent UDP, TCP, and Super Flows open at any given time
Flow Rates	The total rate at which UDP, TCP, and Super Flows are opened.

Selectable Reporting

An Include in Report check box appears on the information tab of each component in a test setup. Select the check box to include the statistics from the test in the report. Deselect the check box to disable the detailed section of the report for that component. Deselecting the check box not only removes the statistics from the test report, it prevents the test statistics from being captured. This feature helps improve database efficiency by allowing you to prevent nonessential information from consuming space in your database.

Once you have selected a section for a particular report, that section will appear in the report whenever it is run in the future. Additionally, the system will attempt to record your previous selections and include those sections in subsequent test reports.

Comparing Test Results

The Comparison Report feature allows you to run multiple iterations of the same test on different blades or different ports and compare the results. The tests being compared must be run on the same component and with the same settings. If any differences exist between the tests being compared, the comparison will produce invalid results.

With the Comparison Report feature, you can compare the results of up to three comparison tests to one base test. You have the option of comparing all sections of the tests, or you can select only certain sections to be included in the comparison.

To compare test results:

1. Select Test > Reporting from the BPS Menu bar (or Results from the Ixia Menu bar).
2. Select the report you would like to view.
3. You can sort the reports by clicking on any of the column headings and scroll through the pages by clicking on the page numbers.
4. Click the **View** button. The test results will open in a new Web browser window.
5. Click the Download drop-down list and select Comparison Report.
6. Select the test(s) to be compared to the base test.

   Note: You can select a maximum of three tests from the Compare this test to section to compare to the base test.

7. Select the section(s) of the report to be included in the comparison from the Choose Sections section.
8. Select the type of data to be included in the comparison from the Data Type section.
9. Select the format of the comparison from the File Format section.

10. Select the check box of the sections to be included in the comparison from the Export - Select Sections Below section.
11. Click the Compare Selected button.

Emailing Test Results

Test results can be automatically e-mailed once a test completes. This option is set per user; therefore, results will be sent to the e-mail configured for the user account. The format in which the test is sent depends on the format that is selected.

From the Results tab of the User Preferences window, you can configure the type of test results you want emailed to you and their format.

To configure the emailed results:

1. Click My Profile | User Preferences from the Ixia Menu bar.
   The User Preferences window displays with the Results tab in the foreground.
2. Click the Results tab.
3. Configure the emailed results settings. (See for fields and parameter descriptions.)
4. Click OK to apply the changes, or Cancel to discard them.

lists and describes the parameters of the Results Tab of the User Preferences window.

Results Tab

Parameter	Description
Email preferences	Type of test results you want to have emailed to you: • Email all test results: Results for all tests. • Email only errors: Only results for tests that experience errors. No results: None. **Note:** If you choose to receive results, you will only receive results for tests run under your user account -- you will not receive results for tests run by other users.
Default report format	File format of emailed test results.

Viewing Reports

Reports provide detailed information about the test – such as the components used in a test, the addressing information, the DUT Profile configuration, the system versions, and the results of the test.

All reports include an aggregated test results section, which provides the combined statistics for all of the test components.

All reports will be automatically generated in HTML; however, you may export the test results in XLS, PDF, RTF, CSV, or ZIP (CSV files). Reports are automatically generated each time a test is run and is viewable from the Reports area in the BPS Control Center or the Results page of the Ixia Menu bar. The reports can be sorted by clicking on any of the column headings (Config Name, User, Duration, etc.) to easily locate reports. The data contained within each report depends on the test components used within the test.

Note: Application Simulator test results will only show a subset of data in its reports. To view the entire report, use the ZIP or XLS formats.

The statistics for Concurrent Flows located in the Slot detail section of the report counts the number of concurrently open Super Flows. This represents the maximum number of simultaneous sessions.

The statistics for Concurrent Flows located in the Component detail section of the report counts the maximum number of concurrent TCP and UDP flows seen since the last statistics update. Since Super Flows often change states during the time between statistic updates, it is possible for a flow to be counted more than once, causing the total values in this section to be greater than the maximum number of simultaneous sessions.

To view a report:

1. Select **Test > Reporting** from the BPS Menu bar or Results from the Ixia Menu bar.
2. Select the report you would like to view from the list.
3. You can sort the reports by clicking on any of the column headings and scrolling through the pages using the scroll bar.
4. Click the **View** button. The report is displayed.

Results page

The Results Page allows you to search for and view all of the reports that are currently available on your system. lists the fields of the Results page.

Results Page Fields

Field	Description
Application Type	Type of test
Config Name	Name of the test
Status	Test status
Result	Final state of the test run
Duration	Duration of the test
Started At	Start time of the test

Field	Description
Ended At	Time test ended (A currently running test will display a blank Ended At time)
User	Name of the user who initiated the test

Searching for Reports

With the Search field of the Results page, you can search for individual reports based on details such as name, userid, and result. To perform a search, enter one of the items listed in the following table into the Search field on the Results page.

contains some of the query strings that can be used to search for specific reports. Enter these query strings into the search field to narrow your search.

Query Strings For Reports

Query Type	Description	Example
`dut`	Lists the type of device under test	`dut:BreakingPoint Default`
`host`	Lists the IP address that a multibox test was run on	`host:10.10.11.204`
`internalid`	Lists the internal reporting ID number	`internalid:322`
`iteration`	Specifies the number of times a test is repeated	`iteration:7`
`name`	Name of the test report	`name:client`
`network`	Lists the type of network used	`network:loopback`
`result`	Lists the final state of the test run	`result:passed`
`subresiliency`	Indicates whether or not the test was part of a Resiliency test	`subresiliency:true`
`testtype`	Lists the type of traffic used by the test components	`testtype:layer4`
`userid`	Lists the login ID of the user who initiated the test	`userid:admin`

Viewing Packet Capture Information

The packet capture information is located in the Capture Enabled and Snap Length columns of the test report.

To view packet capture information in the test report:

1. Select Test > Reporting from the BPS Menu bar or the Results page from the Ixia Menu bar.
2. Select the report you would like to view from the list.

3. Click the View button. The report is displayed.
4. Click the + next to Test Environment in the Navigate section of the report.
5. Select Interfaces.
6. Locate the Capture Enabled and Snap Length columns.

Capture Enabled

The Capture Enabled column will contain one of three possible values:

- NA - Means no information was recorded. This could be caused by a null value, an old test, or unsupported on hardware.
- True - Means the port was set to capture packet data when the test was running.
- False - Means the port was not set to capture packet data (high performance mode).

Snap Length

This value refers to the amount of data for each frame that is actually captured by the network capturing tool and stored into the Capture File. The value is expressed in bytes.

Exporting Reports

By default, the reports will be viewable through a Web browser. If you need to view the report in a different format, you can export the report in any of the following formats: PDF, HTML, RTF, CSV, XLS, BPT, or ZIP (CSV files).

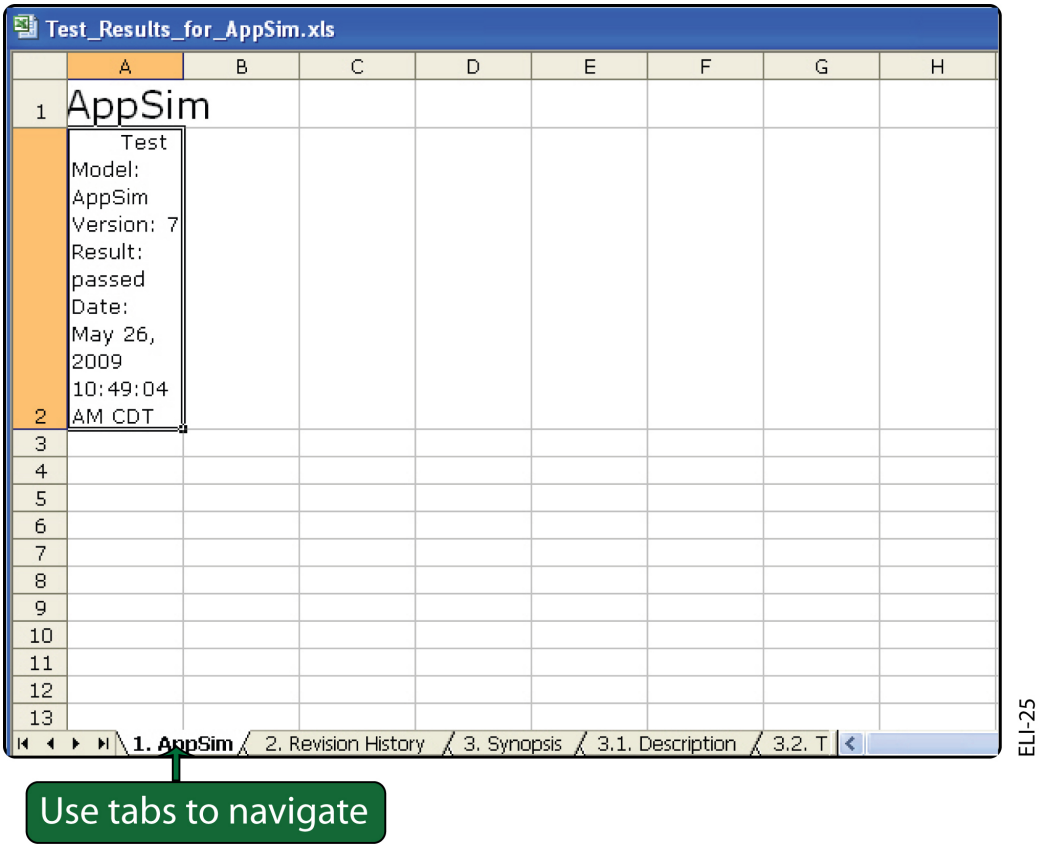
To export a report:

1. Click the Download drop-down button located in the upper right-hand corner of the report.
2. Select **Report**.
3. Select the sections, data type, and file format that you want for your report.
4. Click **Export**.
5. Click the **Save File** option when the **Save** window displays.
6. **Browse** to the location where the report will be stored.
7. Click **OK**.

XLS Files

A report exported as an Excel (XLS) file can be viewed in Excel 2003 or higher. For XLS files, you will need to use the tabs located on the bottom of the spreadsheet to navigate through the different areas of the report. See Report in Excel below.

Report in Excel



CSV Files

A report exported as a CSV file will result in one large spreadsheet, containing all statistics and results from the test.

ZIP Files

ZIP files will contain both text and CSV files; the CSV files will contain the actual test results, and the. text files will contain the section titles from the test report.

Test Model Files

The Test Model is a .bpt file that contains information regarding the executed test. This file includes:

- The Network Neighborhood used by the test
- The DUT Profile used by the test
- Test components used within the test
- Parameter configuration for each test component
- Bandwidth usage on each interface
- Server/Client interface assignment(s)

Note: Reports using the XLS format cannot be viewed with Microsoft Excel 2000.

Deleting Reports

Deleting a report will remove it from the system, and it will no longer be viewable from the system.

To delete a report:

1. Open a web browser.
2. In the URL field, type the IP address or hostname of the BreakingPoint chassis and then press Enter.
   The Login page is displayed.
3. In the **Username** field, type your user ID.
4. In the **Password** field, type your password.
5. If you want the browser to automatically fill in the Username and Password field for future logins, check the **Remember Me** box.
6. Click **Login**. The **Ixia Sessions** page is displayed.
7. Select the **Results** tab. The **Reports** page is displayed.
8. Select the **BreakingPoint** tab. Reports run on the BreakingPoint system will be displayed.
9. Select the report to be deleted. The **Delete** button will become active.
10. Click the **Delete** button.
11. Click the **Yes** button when the confirmation window displays.

The Report Command

The `report` command allows you to list, export, and open test reports.

```
NAME

report - list, export, and open test reports

SHORTCUT: rep

CONFIGURATION

report_command: open
report_dir: /Users/username/reports
report_format: pdf

SYNOPSIS

report [cmd] [args]

report - synonomous with "report list all"
report export <tid> [format [format] [...]] - export the report with test ID
<tid>
report open <tid> [format [format] [...]] - export and open the report with
test ID <tid>
report list - list up to 25 reports created by any user
report list <user|mine|all> - list up to 25 reports created by <user|mine|all>
report list <user|mine|all> <query> - list all reports matching <query> created
by <user|mine|all>
report list <query> - list all reports with title matching <query>

DESCRIPTION

Valid report formats: bpt csv html pdf rtf xls zip

The commands report, report list, and report list all are synonymous.
```

Configuration Options for the Report Command

The configuration option `report_dir` is the directory where reports will be saved after an export. The default location is $HOME/reports.

The configuration option `report_command` is the command that specifies which application to use to view exported reports. The default value is open.

The `report_format` configuration option indicates the default report format used for export. This can be a space-separated string of formats. The default format is .pdf.

Listing Reports

If the first argument after the `list` subcommand is the word mine, then only reports created by you will be returned.

If the first argument after the `list` subcommand is a valid system username, then only reports created by that user will be returned.

If the first argument after the `list` subcommand is the word all or does not match anything else above, then reports created by all users will be returned.

To list all reports:

```
username@storm (group:1)% report
list
```

To list reports with http in the title:

```
username@storm (group:1)% report list http
[+] showing reports belonging to all users with 'http' in the title
[1077] (admin) "Resiliency_HTTP_Client_2_2" May 6, 2012 2:26:44 AM CDT
(0:01:27.532)
[1076] (admin) "Resiliency_HTTP_Client_1_1" May 6, 2012 2:25:39 AM CDT
(0:01:27.545)
...
```

Exporting Reports

To export the report for test ID 1176:

```
username@storm (group:1)% report export
1176
```

To export the report for test ID 1176 in CSV, PDF, and XLS formats:

```
username@storm (group:1)% report export 1176 csv pdf
xls
```

Opening Reports

To export and open the report for test ID 1176:

```
username@storm (group:1)% report open
1176
```

To export and open the report for test ID 1176 in CSV, PDF, and XLS formats:

```
username@storm (group:1)% report open 1176 csv pdf
xls
```

Note: You cannot automatically open an exported report while running `esh` directly on a BreakingPoint device.

CHAPTER 21 Tcl API

This section covers:

Tcl API Introduction

Tcl (Tool Command Language) is a highly extensible and flexible scripting language that runs on Windows, UNIX, and Mac platforms. The BreakingPoint Storm comes with a Tcl shell that enables you to automate device testing via Tcl. If you plan on utilizing BreakingPoint's Tcl API, you will need to download the shell from the system's Start Page. For more information on downloading the Tcl shell, see the section Downloading the Tcl Shell on page 969. Tcl 8.5 syntax is required. Earlier versions of Tcl are not supported.

The BPS Tcl shells are StandAlone RunTime Kits (Starkits) that allow us to wrap and deliver our Tcl shell in a single, self-contained application. You can unwrap a Starkit by using the Starkit Developer Extension (SDX). For more information on Starkits and SDX, visit http://www.equi4.com/starkit/ .

Supported Features

You can automate most of your device testing with the Tcl API. The tasks you can perform with the Tcl API include:

- Creating a connection object for the system
- Creating multiple test configurations
- Creating a Network Neighborhood
- Selecting a Network Neighborhood for a test context
- Selecting a DUT Profile
- Creating App Profiles and Super Flows
- Creating Strike List
- Creating Load Profiles
- Configuring test components
- Creating tests, test series, and multi-box tests
- Running tests, test series, and multi-box tests
- Importing tests
- Importing PCAP files
- Viewing, deleting, and exporting test reports
- Exporting packet buffers
- Mapping and reserving ports
- Rebooting the system
- Performing some administrative tasks – such as creating and modifying user accounts, setting user preferences, retrieving build numbers and ATI Update versions, and performing factory and previous reversions on your system.

Unsupported Features

You will need to log into the Control Center to perform tasks that cannot be done through the Tcl interface. These tasks include:

- Creating DUT Profiles
- Importing CA certificates, client certificates, and private keys
- Importing files for URI messages/attachments in flows
- Importing and exporting Strike List
- Performing some administrative tasks – such as setting the time zone
- Modifying a subnet within a Network Neighborhood

Downloading the Tcl Shell

You can download the Tcl shell from the Ixia Start Page. You must download the latest Tcl shell each time you update BreakingPoint to the latest release. The Tcl Shell version must match BreakingPoint build number.

Note: For Linux and MAC OS X (and greater versions), you will need to run the BPS shell from the command line. You cannot double-click on the executable file to launch the BPS shell.

To download the Tcl shell:

1. Open a web browser.
2. In the URL field, type the IP address or hostname of the Ixia chassis where the Ixia Web App server components are installed, followed by the port number that the Ixia Web App server is listening on (the default is 8080), and then press Enter.
   a. For example: 192.168.100.56:8080
   b. The Login page is displayed.
3. In the Username field, type your user ID.
4. In the Password field, type your password.
   a. If you want the browser to automatically fill in the Username and Password field for future logins, check the Remember Me box.
5. Click **Login**.
   The Sessions page displays.
6. Select **Help > Downloads > Download Tcl Shell**.
7. Click one of the following links:
   - Tcl Shell – Windows Version
   - Tcl Shell – Linux Version
   - Tcl Shell – Mac OS X Version
8. Click the **Save** button.
9. Select the location to store the .exe file.
10. Click the **Save** button.
11. Double-click the executable file to open the Tcl interface.

Accessing the BPS Tcl Shell via SSH

You can also access the BreakingPoint Tcl Shell via SSH. This means that you can place automation scripts directly onto your BreakingPoint Storm. You can run those scripts without having to install the BreakingPoint Tcl Shell anywhere.

This makes the BreakingPoint Tcl Shell accessible from `chroot` environment. So, after you log in via Telnet, SSH, or Serial as a normal (non-root) user, you can use the following example to login:

```
bps> bpsh
Active BPS connection available as $bps
% set c [$bps getChassis]
::bps::BPSConnection::bPSConnection0::chassisClient0
%
```

Or, if you want to connect to the BreakingPoint Tcl Shell using an account different from the one you logged in with, you can use the following example to do so:

```
bps> bpsh -nologin
% set bps [bps::connect 127.0.0.1 admin
admin]
bPSConnection2
```

To connect to a completely different BreakingPoint device over the logged-in device's management Ethernet interface, you can use the following example to do so:

```
bps> bpsh -nologin
% set bps [bps::connect 10.10.10.10 admin
admin]
bPSConnection3
```

Combining the BPS Tcl Shell with Existing Tcl Shells

In order to combine the BreakingPoint Tcl shell with an existing Tcl shell, you will need to point the BPS Tcl shell to your existing Tcl extensions. Use the `auto_path` variable to point to the location path of your installed Tcl extensions. Once you point your Tcl shell to the existing extensions, you can load them into the Tcl shell using `package require`.

Macs have a full compliment of Tcl extensions installed in the directory `/System/Library/Tcl`. So to use the extra extensions provided on a Mac, run the following syntax:

```
lappend auto_path
/System/Library/Tcl
package require Tk
```

For a few extensions (e.g., iTK) you may also have to set an environment variable. In this case, you will want to run the following:

```
lappend auto_path /System/Library/Tcl
set env(ITK_LIBRARY)
/System/Library/Tcl/itk3.3
package require Itk
package require Iwidgets
```

If you have Tcl extensions installed on UNIX, they are most likely located in `/usr/lib/` or `/usr/local/lib`. For Ubuntu and some other Linux distributions, Tcl libraries are located in `/usr/share/tcltk`.

You can do the same thing on UNIX or Windows using the correct path for your system.

The following lists the shared libraries required to run the BreakingPoint Tcl shell using Linux.

```
/lib32/libnss_files-2.11.1.so
/lib32/libnsl-2.11.1.so/lib32/libc-2.11.1.so
/usr/lib32/libgcc_s.so.1/lib32/libm-2.11.1.so
/usr/lib32/libstdc++.so.6.0.13/lib32/libpthread-
2.11.1.so
/lib32/libdl-2.11.1.so/lib32/libutil-2.11.1.so
/lib32/libnss_nis-2.11.1.so
/lib32/libnss_compat-2.11.1.so/lib32/ld-2.11.1.so
```

Note:

- Version numbers and specific location will vary by distribution.
- If the library you are using is installed on a 64-bit version of Linux, you will need to install and point to your 32-bit compatibility libraries in order to use the BreakingPoint Tcl shell.

For Debian-based distributions, you can try the following:

```
sudo dpkg --add-architecture
i386

apt update

sudo apt-get install ia32-libs
```

Searching for Package Names

You can view the package names for the BreakingPoint API by using the following syntax.

Syntax

```
package
names
```

Navigating the Tcl API

Because most API calls are discoverable, you can find out what the call is by interacting with Tcl. This is especially useful when Tcl documentation is not available.

You can run the BPS Tcl shell in interactive mode and type directly into it rather than running a script.To find out the methods available on any given object, issue the object name without arguments. The error that is returned will list all available methods.

For all objects that have configurable parameters, there are methods available that list the available parameters. For example, to get a list of the actions that are available to be added for a particular Super Flow, use the `getActionChoices` method. To find the available parameters on a flow, use the

`getFlowParameters` or `getActionParameters`. When you set a parameter and enter an invalid value, you will get an error that provides the valid values.

Example

```
# Look inside a Super Flow using the superflowID proc InspectSF
{sfID} {

set sfName [$sfID cget -name]
puts "\n$sfName INFO"
puts "HOSTS"
dict for {host data} [$sfID getHosts] {
puts "\tHOST: $host\tDATA: $data"
}
puts "FLOWS"
dict for {flwID data} [$sfID getFlows] {
puts "\tflow ID: $flwID\tDATA: $data"
set p [dict get [$sfID getFlows] $flwID protocol]
puts "\n$p ACTION CHOICES:"
foreach {src xid} [$sfID getActionChoices $flwID] {
puts "\t$src could $xid"
} ; # end foreach
puts "ACTIONS"
dict for {actID data} [$sfID getActions -flowid $flwID] {
puts "\tact ID: $actID\tDATA: $data"
} ; # end dict for
} ; # end dict for
} ; # end proc
```

Tcl Commands and Syntax Overview

Tcl scripts are made up of commands separated by new lines or semicolons. The first part of a statement introduces the command, which is followed by arguments to that command. The following table details the Tcl commands that are specific to the BPS Tcl API. Some commands will be part of the BPS connection object; others will be commands of other objects you will create for items such as the chassis, Network Neighborhood, App Profile, Strike List, and Load Profile objects.

Note:

- Prior to version 2.0 of the BreakingPoint product, parsing command line arguments did not set the `$argv` argument appropriately. As a workaround, customers were required to strip the first argument of `$argv` before processing command line arguments. The BreakingPoint Storm now sets the `$argv` appropriately when parsing command line arguments. You are no longer required to strip the first argument before parsing command line parameters. Manually stripping from `$argv` will cause the first command line argument to be lost.
- The following commands are not standalone commands. Each command listed must be preceded by an object.

Tcl Commands

Command	Description
`addAction flowID source actionType`	Adds an action to a Super Flow; this is a command of the Super Flow object. flowID should be replaced with the flow ID to which the action will be added; source should be replaced with either client or server; and the actionType should be replaced with the type of action that is being added (e.g., GET, PUT, POST, etc.).
`addDHCPClients interface domain ?arg arg?`	Adds a DHCP Client subnet to a domain, places one DHCP onto the subnet, and sets up clients on the subnet; this is a command of the Network Neighborhood object.
`addDomain interface domainName`	Adds a domain to an interface; this is a command of the Network Neighborhood object.
`addENodeB interface domain ?arg arg?`	Adds an eNodeB client to the subnet you created using addENodeBClients; this is a command of the Network Neighborhood object.
`addENodeBClients interface domain ?arg arg?`	Adds an LTE eNodeB (Towers) subnet to a domain, places one eNodeB onto the subnet, and sets up clients on the subnet; this is a command of the Network Neighborhood object.
`addFlow protocolType host1 host2`	Adds a flow to a Super Flow; protocolType should be replaced with the protocol on which the flow will be based; host1 should be replaced with the host name that the flow will start from; and host2 should be replaced with the host name where the flow will end (e.g.,$var addFlow pop3 client server). This is a command of the Super Flow object.
`addGGSN interface domain ?arg arg?`	Adds a GTP GGSN (Gateway GPRS Support Node) client to a subnet; this is a command of the Network Neighborhood object.
`addHost hostNickName interface hostName`	Adds a host to either the server or client interface. If the host will be on the client-side, then interface should be replaced with origin. If the host will be on the server-side, then interface should be replaced wit target. This is a command of the Super Flow object.
`addHostRange interface domain sixrd_prefix sixrd_prefix_len sixrd_ip4_mask_len sixrd_border_relay sixrd_hosts_per_ce ?arg arg?`	Adds a range of host IP addresses available on the network. This is a command of the Network Neighborhood object.

Command	Description
`addImpairment interface ?arg arg?`	Adds impairments to an interface; this is a command of the Network Neighborhood object.
`addMatchAction actionID matchID actionMatchid source matchAction`	Adds a match to a Conditional Request. This is a command of the Super Flow object. The actionID represents the action to which you would like to add the Conditional Request; the matchID represents the sequence number at which the match will be added; the actionMatchID represents the sequence number at which the action match (string) will be added; the source can either be 'client' or 'server'; and the matchAction represents the action defined for the match.
`addMMEClients interface domain ?arg arg?`	Adds an eNodeB/MME (Mobility Management Entity) subnet to a domain, places one MME onto the subnet, and sets up clients on the subnet; this is a command of the Network Neighborhood object.
`addPath sourceinterface sourcedomain sourcevlan destinterface destdomain destvlan`	Adds a defined path for a subnet.
`addSGSN interface domain ?arg arg?`	Adds a GTP SGSN (Service GPRS Support Node) client to a subnet; this is a command of the Network Neighborhood object.
`addSGSNClients interface domain ?arg arg?`	Adds a GTP SGSN (Service GPRS Support Node) subnet to a domain, places one SGSN onto the subnet, and sets up clients on the subnet; this is a command of the Network Neighborhood object.
`addSGWClients interface domain ?arg arg?`	Adds an LTE SGW/PGW (Server Gateway/PDN Gateway) subnet to a domain, places one SGW onto the subnet, and sets up clients on the subnet; this is a command of the Network Neighborhood object.
`addStrike groupName strikeName`	Adds a Strike to a group; this is a command of the Strike List object.
`addSubnet interface domainName subnetName`	Adds a subnet to a domain; this is a command of the Network Neighborhood object.
`addSuperflow superFlowName weight`	Adds a Super Flow to an App Profile and assigns it a weight; this is a command of the App Profile object.

Command	Description
addUser id password name email	Adds a user to the system. You can also add a user to a group by using the -group attribute; this is a command of the Connection Object.
aggStats objectName	Stores the aggregate statistics for a test in an object.
backup location fileName	Performs a backup to a USB or an external hard drive.
bps::connect IPaddress user password	Creates a connection object to the system.
"bps::textprogress outputChannel"	Used with the -progress attribute to show the progress of a test while it is executing. You must specify the channel in which the text should be output to; the most common channel is stdout.
cancel	Cancels a test that has been running with the -async attribute.
cget -option	Retrieves the setting of an option.
clearResults context?	Clears the stored results of a test context. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
configure -option? value?	Sets the value for a parameter.
configureContext contextName arg? arg?	Configures additional test contexts using initContext. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
createAppProfile arg? arg?	Creates an App Profile.

Command	Description
createStrikeList arg? arg?	Creates a Strike List.
createComponent arg? arg?	Creates a test component.
createEvasionProfile	Creates an Evasion Profile
createLTETest arg? arg?	Creates an LTE test.
createLawfulInterceptTest arg? arg?	Creates a Lawful Intercept test.
createLoadProfile arg? arg?	Creates a Load Profile.
createMultiboxTest arg? arg?	Creates a multi-box test.
createMulticastTest arg? arg?	Creates a Multicast test
createNetwork arg? arg?	Creates a Network object.
createRFC2544Test arg? arg?	Creates the RFC2544 Test.
createResiliencyTest arg? arg?	Creates a Resiliency Score Test.
createServerResiliencyTest arg? arg?	Creates a Server Resiliency Score Test.
createSessionLabTest ?arg arg?	Creates a Session Sender Lab test.
createStrikeList arg? arg?	Creates a Strike List.
createSuperflow arg? arg?	Creates a Super Flow.
createTest arg? arg?	Creates a test.
createTestSeries arg? arg?	Creates a test series.
delete	Deletes the connection.
deleteAppProfile arg? arg?	Removes an App Profile from the system.
deleteAttackSeries arg? arg?	Removes a Strike List from the system.

Command	Description
`deleteContext contextName`	Removes a context from the script.
`deleteEvasionProfile arg? arg?`	Removes an Evasion Profile from the system.
`deleteLoadProfile loadProfileName`	Removes a Load Profile from the system.
`deleteMultiboxTest testName`	Removes a multibox test from the system.
`deleteNeighborhood neighborhoodName`	Removes a Network Neighborhood from the system.
`deleteStrikeList arg? arg?`	Removes a Strike List from the system.
`deleteSuperFlow superflowName`	Removes a Super Flow from the system.
`deleteTest testName`	Removes a test from the system. Use the `-force` attribute to force the deletion.
`deleteTestResults arg? arg?`	Removes test results from the system.
`deleteTestSeries testseriesName`	Removes a test series from the system.
`domainNames interface`	Lists the domain names for an interface; this is a command of the Network Neighborhood object.
`exportReport -file /../reportName.type`	Exports the report in PDF, XLS, ZIP, or HTML.
`exportPacketTrace -file /location/ $slot $port direction`	Exports the packet buffer for the listed slot(s)/port(s) to the specified location. You can indicate the direction of the traffic you want exported by specifying `both`, `tx`, or `rx`. Specifying `both` will export both transmitted and received traffic, whereas `tx` will export only transmitted traffic, and `rx` will export only received traffic.
`factoryRevert`	Reverts the system back to factory settings.
`get`	Returns a list of ID-object pairs.
`getActionChoices flowID`	Returns a list of actions that can be used for a specific flow. This command is part of the Super Flow object.

Command	Description
`getActionParameters actionID`	Returns a list of Action Parameters that are available for an action. This command is part of the Super Flow object, and you must specify the action ID to view the Action Parameters.
`getActions`	Returns a list of actions that are used within a Super Flow; this is a command of the Super Flow object.
`getAggStats objectName`	Returns the values stored in the object created using the `aggStats` command.
`getAll`	Returns a list of ID-object pairs.
`getBuildId`	Returns the system's build number.
`getChassis`	Creates the chassis object.
`getComponents`	Returns a list of logical name/object pairs for the components in the test.
`getDHCPServer interface domain ?innervlan? ?outervlan?`	Adds a DHCP server to a domain; this is a command of the Network Neighborhood object.
`getDut context`	Returns the DUT Profile used for the test context.
`getFilters`	Returns a list of the filter used in your test.
`getFlows`	Returns a list of flows that are in a Super Flow; this is a command of the Super Flow object.
`getFlowParameters flowName`	Returns a list of protocol parameters for a specific flow (e.g., http, pop3, dns).
`getHosts`	Returns a list of available hosts for a Super Flow; this is a command of the Super Flow object.
`getImpairments`	Returns a list of impairments for an interface; this is a command of the Network Neighborhood object.
`getMatchActionParameters actionID matchID matchActionID`	Returns the Action Parameters that are available for a Match Action. This is a command of the Super Flow object.

Command	Description
getNeighborhood context?	Returns the Network Neighborhood used for the test context. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
getPaths	Returns a defined path of a subnet.
getQuery	Returns the query that the Smart Strike List currently has set.
getReportComponents testid	Returns the distinct component names covered by the report.
getReportContents componentID	Returns a list of report sections.
getReportSectionXML	Returns an XML document that defines a section of the report.
getReportTable sectionID sectionTitle	Returns a Tcl dictionary containing the table contents within a particular section.
getState $slot $port	Returns the information for each port on the BreakingPoint Storm; this includes the port's media type, active group, speed, auto-negotiation settings, state, port note, duplex type, link status, user reservation, and blade model; this is a command of the chassis object.
getStrikeInfo strikeName	Returns the information for a specific Strike.
getStrikepackId	Returns the ATI Update version.
getSubnets interface name	Returns the information for a specific subnet.
getSystemGlobal varName	Sets a global variable.
getSystemType	Returns the system type.
getTest context	Returns a list of systems and the tests running on them.
getVersion	Returns the system version (e.g., 1.2.1)

Command	Description
`getVlanEtherType interface`	Lists the VLAN Ethertype for the interface; this is a command of the Network Neighborhood object.
`host`	Returns the management IP address for the system.
`importPcap name arg? arg?`	Import an existing PCAP file into the system.
`importTest testName? arg? arg?`	Import an existing test into the system.
`initContext contextName arg? arg?`	Creates a test context.
`installStrikepack -file location`	Installs an ATI Update on the system.
`installUpdate -url address`	Installs an update on the system.
`listAppProfiles arg? arg?`	Lists the App Profiles that are available.
`listBackups arg? arg?`	Lists backup files
`listEvasionProfiles arg? arg?`	Lists the Evasion Profiles that are available.
`listDUTs arg? arg?`	Lists the DUT Profiles that are available.
`listLoadProfiles arg? arg?`	Lists the Load Profiles that are available.
`listNeighborhoods arg? arg?`	Lists the Network Neighborhoods that are available.
`listProtocols arg? arg?`	Lists the protocols that are available.
`listStrikeKeywords arg? arg?`	Lists the Strike keywords that are available.
`listStrikes arg? arg?`	Lists the Strikes that are available.
`listSuperflows arg? arg?`	Lists the Super Flows that are available.
`listTestResults arg? arg?`	Lists test results and user ID from the system.

Command	Description
`modifyFlow flowName`	Modifies a specific flow in a Super Flow; this command allows you to change the hosts specified for the flow and modify the protocol parameters for the flow. Use the attributes `-to` and `-from` the modify the hosts for the flow (e.g., $var modifyFlow 1 -from Server -to Client).
`modifyHost hostName`	Modifies the host. Use the `-iface` and `-name` attributes to change the interface or change the name (e.g., modifyHost DNS -iface target -name dnsServer).
`modifyMatchAction actionID matchID matchActionID matchAction`	Modifes the existing settings for a match. This is a command of the Super Flow object. The actionID represents the action to which you would like to add the Conditional Request; the matchID represents the sequence number at which the match will be added; the actionMatchID represents the sequence number at which the action match (string) will be added; the source can either be 'client' or 'server'; and the matchAction represents the action defined for the match.
`previousRevert`	Reverts the system back to the previous build.
`reboot`	Reboots the system.
`removeDHCPClients interface domain ?innervlan? ?outervlan?`	Removes the DHCP clients from the interface; this is a command of the Network Neighborhood object.
`removeDomain interface domainName`	Removes the domain from the interface; this is a command of the Network Neighborhood object. When an interface is deleted, the system will automatically resequence the interfaces. The succeeding interfaces (following the deleted interface) will be renumbered to the preceding interface's value (e.g., '6' will become '5').
`removeENodeB interface domain ?arg arg?`	Removes an LTE eNodeB (Towers) client from a subnet; this is a command of the Network Neighborhood object.
`removeENodeBClients interface domain ?innervlan? ?outervlan?`	Removes an eNodeB subnet from the domain; this is a command of the Network Neighborhood object.
`removeFilter interface`	Removes a packet filter from your test.
`removeFlow flowName`	Removes a flow from a Super Flow; this is a command of the Super Flow object.

Command	Description
`removeGGSN interface domain ?innervlan? ?outervlan?`	Removes a GTP GGSN (Service GPRS Support Node) client from a subnet; this is a command of the Network Neighborhood object.
`removeHostRange interface domain ?innervlan? ?outervlan?`	Removes a range of host IP addresses from the network; this is a command of the Network Neighborhood object.
`removeImpairment interface`	Removes impairments from an interface; this is a command of the Network Neighborhood object.
`removeMatchAction actionID matchID actionMatchID`	Removes an action from a match.
`removeMMEClients interface domain ?innervlan? ?outervlan?`	Removes an LTE eNodeB/MME (Mobility Management Entity) subnet from a domain; this is a command of the Network Neighborhood object.
`removePath sourceinterface sourcedomain sourcevlan destinterface destdomain destvlan`	Removes a defined path of a subnet.
`removeSGSN interface domain ?arg arg?`	Removes a GTP SGSN (Service GPRS Support Node) client from a subnet; this is a command of the Network Neighborhood object.
`removeSGSNClients interface domain ?innervlan? ?outervlan?`	Removes a GTP SGSN (Service GPRS Support Node) subnet from a domain; this is a command of the Network Neighborhood object.
`removeSGWClients interface domain ?innervlan? ?outervlan?`	Removes an LTE SGW/PGW (Server Gateway/PDN Gateway) subnet from a domain; this is a command of the Network Neighborhood object.
`removeStrike groupName strikeName`	Removes a Strike from an Evasion Setting; this is a command of the Strike List object.
`removeSubnet interface name ?innervlan? ?outervlan?`	Removes the subnet from a domain.
`removeSuperflow superflowName`	Removes a Super Flow from an App Profile.
`reservePort $slot $port`	Reserves the specified slot/port. This is a command of the chassis object.

Command	Description
`restoreBackup arg? arg?`	Restores backup file
`resultId context?`	Returns the variable. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
`run arg? arg?`	Runs the test.
`save arg? arg?`	Saves the current test.
`searchStrikes arg? arg?`	Searches the available Strikes.
`searchStrikeLists arg? arg?`	Searches the available Strike Lists.
`setDHCPServer interface domain ?arg arg?`	Sets the DHCP server for a test context.
`setDut name context?`	Sets the DUT Profile for a test context. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
`setFilter interface ?arg arg?`	Sets up the filter to be used in your test.

Command	Description
`setNeighborhood neighborhoodName context?`	Sets a Network Neighborhood for a test context. * The context variable is used for backwards compatibility with an earlier version of BPS Tcl API. For the currently available Tcl methods, context has the default value, "default". You should not specify the "context" parameter in the parameter list as this is deprecated and would override the default value. The methods that work with contexts (for example: configureContext, deleteContext) are deprecated and should not be used.
`setPortOrder $slot $port`	Enables you to arrange the order of the ports. This is a command of the chassis object.
`setQuery`	Sets the query that the Smart Strike List will use to locate Strikes.
`setStrikesFromQuery`	Saves a copy of a Smart Strike List. The resulting list is static and will not be updated when ATI Updates adds new Strikes that match the query.
`setVlanEtherType interface value`	Sets the Ethertype for the interface; this is a command of the Network Neighborhood object. Values can be 0x88a8, 0x8100, 0x9100, 0x9200, and 0x9300.
`unreservePort $slot $port`	Unreserves the specified slot and port. This is a command of the chassis object.
`unsetActionParameter actionID matchAction`	Reverts the match action to its default configuration.
`unsetFlowParameter flowName -protocolParameter`	Reverts the protocol parameter back to its default value.
`wait`	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the `-async` attribute.
`weightType`	Sets whether the weight of a Super Flow determines its proportion in the traffic by flow count, or by bandwidth.

Optional Arguments

Some Tcl commands have optional attributes that can run with some commands. The table below details these options.

Optional Arguments

Optional Arguments	Description
-allowMalware	Allows you to bypass the Malware error message and run a test. This is an atribute to the run command for tests, test labs, test series, and multibox tests.
-async value	Specified as an attribute to the run command. This attribute runs the test in the background, and executes the command specified.
-capture	Specified as an attribute to the reservePort and configurePort commands. Provides the ability to enable and disable the packet capture feature.
-class value	Specified as an attribute of the listTestResults command. Identifies the type of test results to list. Accepted values include single, resiliency, series, or multi.
-enodeb	Accepts a list of IPs to add to MME and eNodeB clients.
-file location	Specified as an attribute to the installStrikepack command. This references the location of the ATI update file.
-flowexceptions	Provides the ability to access flow exceptions during test runs.
-force true	Specified as an attribute to any command that creates or modifies an object. This attribute allows you to force the system to override the existing object. If you do not specify true or false after the statement, the system will automatically assume that the value is true.
-mcc	Specified as an attribute to the Network Neighborhood object. Specifies the Mobile Country Code of the device to be tested.
-mnc	Specified as an attribute to the Network Neighborhood object. Specifies the Mobile Network Code of the device to be tested.
-name value	Used with the save or configure command to name or rename an item (e.g., App Profile, Super Flow, test context, etc.).
-newid value	This attribute allows you to rename the Evasion Setting.
-onclose value	Specified as an attribute to the test context. This attribute allows you to select what happens to after a script completes. The most common value used here is exit.
-onlink	Specified as an attribute to the chassis object.This calls this attribute's callback when a link on an interface goes up or down.

Optional Arguments	Description
-onreserve	Specified as an attribute to the chassis object. This calls this attribute's callback when someone reserves or unreserves a port.
-onstate	Specified as an attribute to the chassis object.This calls this attribute's callback when there is a change in the system's state. You will most likely see this when a blade is offline.
-operator_ variant	Specified as an attribute to the Network Neighborhood object. Specifies a unique value originally assigned by the UE manufacturer. The operator variant is usually unique to each brand of UE.
-progress value	Specified as an attribute to the `run` command. This attribute lets you specify a Tcl script that will be called periodically while the test runs. The test name and a percentage of completion will be appended to the script you provide via the 'concat' command. The default value is the empty string, which means that no command will run to show the test progress.
-qci_labels	Specified as an attribute to the Network Neighborhood object. The -qci_labels attribute retrieves the qci information for eNodeB and MME clients.
-rtstats value	Specifies a callback to update your charts with Real-Time statistics.
-sctp_over_udp	Specified as an attribute to the Network Neighborhood object. Enables or disables the tunneling of SCTP over UDP.
-sctp_sport	Specified as an attribute to the Network Neighborhood object.
-secret_key	Specified as an attribute to the Network Neighborhood object. Specifies the unique identifying number of each UE.
-shortcuts value	Specified as an attribute to the test context.This attribute enables or disables the test component shortcuts. If enabled, this allows you to use the default test components. The value for this attribute is set to true by default.
-url location	Specified as an attribute to the `installUpdate` command. This references the location of the OS update file.

The Help Command

To access help for the Enhanced Shell environment, type `?` or `help` at the prompt.

Example

```
username@storm (group:1)% help

[+] Global variables:
$bps - bps connection
$ch - chassis

COMMAND ARGUMENTS DESCRIPTION
---------- ------------ ----------------------------------------------------
----
alias [alias] [val] - set, unset, or list aliases
cleanup [type] - remove temporary files
config [var] [val] - show, set, unset, or reset configuration settings
ep [cmd] [args] - create, delete, reload, or list evasion profiles
group [group] - choose your current test group explicitly or automatically
help [command] - show this help or detailed help for [command]
host [cmd] [args] - view or modify host system settings
nn [cmd] - list network neighborhoods
pcap [cmd] [args] - export and open pcaps
port [cmd] [args] - list, reserve, or unreserve ports
repeat [opts] [cmd] - repeat a command
report [cmd] [args] - list, export, and open test reports
shell [command] - invoke a local subshell (or execute [command])
strike [cmd] [args] - manage or execute strikes and strike lists
test [cmd] [args] - list or run tests
testseries [cmd] [args] - manage test series
user [cmd] [args] - add, delete, or list users

[+] TIP: use "<command> ?" to get more help for any command
```

Command-Specific Help

Type a question mark after a command to get command-specific help for that particular command.

Syntax

```
username@storm (group:1)%
help
```

Example

The following example displays command-specific help for the `test` command.

```
username@storm (group:1)% test ?

NAME

test - list or run tests

SHORTCUT: t

CONFIGURATION

test_dir: /home/username/tests
test_mode: async

SYNOPSIS

test [cmd] [args]

test - synonomous with "test list all"
test cancel - cancel all tests running under your username
test cancel <tid> [...] - cancel the specified test(s) with matching <tid>
test copy <name> to <newname> - copy test <name> to <newname>
test delete <names> - delete tests matching <names>
test import <URL> [as <name>] [force] - import test [as <name>] from <URL>
test import <pattern> [force] - import local tests matching <pattern>
test export <name> - export test named <name>
test export <pattern> - export tests matching <pattern>
test list - list all tests created by any user
test list <user|mine|all> - list all tests created by <user|mine|all>
test list <user|mine|all> <query> - list all tests matching <query> created by
<user|mine|all>
test list <query> - list all tests with title matching <query>
test list local <pattern> - list local tests matching <pattern>
test rename <name> to <newname> - rename test <name> to <newname>
test run <tests> [options] - run test(s)
test run <tests> series - run test(s) as a test series

DESCRIPTION

You can specify tests to run via the numbers shown in the test list. Test
options: mode (async or sync) and nn (network neighborhood). Test "patterns"
should contain an asterisk to indicate multiple files.
```

Tcl Objects

For most tasks in the Control Center, you will need to create an object for them. The object provides access to a subset of commands, specific to the object that was created.

The following tasks will require an object:

- Creating a chassis object
- Creating a Network Neighborhood
- Creating an App Profile
- Creating Super Flows
- Creating an Attack Plan
- Creating a Strike List
- Creating a Load Profile
- Creating a Test
- Creating a Test Series
- Creating a Multi-box Test
- Creating a Test Component
- Returning test results

Deleting Objects

Once you are done with an object, you should remove the object from the system. Before deleting the object from the system, verify that you have saved all the necessary components created from the object – such as tests, test series, multi-box tests, Load Profiles, etc.

Syntax

Use the following syntax to delete an object from the system.

```
itcl::deleteobject
objectName
```

Example

```
itcl::deleteobject SS1; #deletes the SS1
object
```

Connecting to the System

The `bps::connect` command allows you to create a connection to the system. It will create an object that represents the connection, and it will store the connection in a variable, which you can use later on to utilize commands for the BreakingPoint system.

You can call the connection object to list all of the commands available. For example, `$connectionObject` will return a list of all the commands that are available for the connection object. For a list of these commands see the following table.

Syntax

Use the following syntax to connect to the system.

```
set var [bps::connect host username password -option
arg?]
```

The following table lists breaks down the elements of connecting to the system.

Connecting to the System

Element	Description
var	Sets the variable name for the connection
bps::connect	The command for creating a connection
host	The management IP address for the system
username	User account login ID
password	User account password

Connection Object Optional Attributes

The bps::connect command includes the following connection attributes: -shortcuts, and -onclose. The following table lists descriptions for these arguments.

Connection Arguments

Argument	Description
-name	Allows you to name or rename an item.
-onclose	Determines what a script does once it finishes running.
-shortcuts	Set the value to 'true' to enable shortcut commands for test components. The default value is set to 'true, and allows you to utilize the preset test components stored on your system. Set the value to 'false' to create and utilize your own.

Example

The following example creates a simple connection object to 10.10.10.10.

```
set bps [bps::connect 10.10.10.10 john passwd -onclose exit -name test1 -
shortcuts true]
```

Creating the Chassis Object

You can create a chassis object using the connection object. The chassis object will be used to control features from the Device Status area – including port reservations, port configurations, port mappings,

and packet buffer exports. Additionally, you can use the chassis object to retrieve information for the system – including when the system status, port reservations, and link status change.

See the following table for a list of attributes and commands available for the chassis object.

Attributes for the Chassis Object

Attributes/Command	Description
`configurePort`	Configures the specified slot and port (`$chassisObject configurePort 1 0`).
`getChassis`	Creates the chassis object (`set chassisObject [$connectionObject getChassis]`) .
`getState`	Returns the information for each port on the BreakingPoint Storm; this includes the port's media type, active group, speed, auto-negotiation settings, state, port note, duplex type, link status, user reservation, and blade model. You can use this command by itself (`$chassisObject getState`) to return information for all ports on the blade, or you can narrow the results by specifying a slot and port (`$chassisObject getState 1 2`).
`reservePort $slot $port`	Reserves the specified slot and port (`$chassisObject reservePort 1 0`).
`unreservePort $slot $port`	Unreserves the specified slot and port (`$chassisObject unreservePort 1 0`).
`setPortOrder`	Sets the order of the ports (`$chassisObject setPortOrder 1 0 1 1 1 2 1 3`). This command can only be used for ports that have existing reservations.
`exportPacktTrace`	Exports the packet buffer to a specified directory location (`$chassisObject exportPacketTrace -progress {bps::textprogress stdout} /tmp 1 0 tx 1 0 rx`).
`-onstate value`	Calls the `-onstate` callback when there is a change in the system's state.
`-onreserve value`	Calls the`-onreserve` callback when someone reserves or unreserves a port.
`-onlink value`	Calls the `-onlink` callback when a link on an interface goes up or down.

Syntax

Use the following syntax to create a connection to the system and to create a chassis object.

```
set var [bps::connect host username password -option
arg?]
set chassisObjectName [$var getChassis]
```

Example

This example creates callbacks and a chassis object. The callbacks will be returned if port reservations, link statuses, or the system state are changed during the course of the test.

```
# the callback for a state change should accept these arguments
proc reportStateChange {slot port state}
{
puts "slot $slot, port $port is now in state $state"
}

# the callback for a port reservation should accept these arguments
proc reportReservation {slot port reservedBy group}
{
if {$reservedBy == ""}
{
puts "slot $slot, port $port has been unreserved"
return
}
puts "slot $slot, port $port is reserved by $reservedBy in group
$group"
}

# the callback for a link change should accept these arguments
proc reportLink {slot port link media speed duplex}
{
puts "link is now $link on slot $slot, port $port"
if {$media != ""}
{
puts "using $media at speed=$speed, duplex=$duplex"
}
}

set c1 [$bps getChassis -onreserve reportReservation \
-onstate reportStateChange \
-onlink reportLink]
$c1 reservePort 1 0 -group 1
$c1 reservePort 2 1 -group 2
$c1 unreservePort 2 1

$c1 getState

$c1 configurePort 0 0 -auto false -speed 100 -fullduplex false
```

Port Commands

The following commands are described in this section:

Reserving Ports

The chassis object has a command called `reservePort` that enables you to reserve ports on the BreakingPoint Storm.

Syntax

Use the following syntax to reserve ports on the BreakingPoint Storm.

```
$chassisObject reservePort $slotNumber $portNumber -group
$groupNumber
```

Example

The following example reserves ports 0 and 1 on slot 1.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set c1 [$var getChassis]; #creates the chassis object
$c1 reservePort 1 0; #reserves ports 0 on slot 1
$c1 reservePort 1 1; #reserves ports 1 on slot 1
```

Re-Ordering Ports

The chassis object has a command called `setPortOrder` that enables you to set the order of the ports on the BreakingPoint Storm. When you reserve ports, the system automatically maps the ports to interfaces based on the order in which you reserved the ports. Therefore, this command enables you to change that order.

Note: You can only re-order ports for which you have port reservations.

Syntax

Use the following syntax to order the ports on the BreakingPoint Storm.

```
$chassisObject setPortOrder $slotNumber1 $portNumber1 $slotNumber2 $portNumber2
$slotNumber3 $portNumber3 will make $slotNumber1/$portNumber1 become interface
1 of the group, $slotNumber2/$portNumber2 become interface 2 of the group,
$slotNumber3/$portNumber3 become interface 3 of the group
```

Example

The following example reorders the ports.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set c1 [$var getChassis]; #creates the chassis object
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
$c1 setPortOrder 1 0 1 1 1 2 1 3; #reorders the port mappings
```

Unreserving Ports

The chassis object has a command called `unreservePort` that enables you to unreserve ports on the BreakingPoint Storm.

Syntax

Use the following syntax to unreserve ports on the BreakingPoint Storm.

```
$chassisObject unreservePort $slotNumber
$portNumber
```

Example

The following example unreserves port 0 on slot 1.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set c1 [$var getChassis]; #creates the chassis object
$c1 unreservePort 1 0; #unreserves port 0 on slot 1
```

The Port Command

The port command allows you to list, reserve, or unreserve ports.

```
NAME
port - list, reserve, or unreserve ports

SHORTCUT: p

CONFIGURATION

port_force_reserve: yes
port_mtu: 9198

SYNOPSIS

port [cmd] [args]

port - synonomous with "port list"
port configure <slot/port combo> [opts] - configure specified slots/ports with
options
port details [slot/port combo] - show details of specified ports
port list [slot/port combo] - show status of slots and ports
port note <slot/port combo> - list notes for specified slots/ports
port note <slot/port combo> <msg> - set note for specified slots/ports to <msg>
port note <slot/port combo> remove - remove note from specified slots/ports
port reserve [slot/port combo] [opts] - reserve the specified slots/ports with
options
port unreserve [slot/port combo] - unreserve the specified slots/ports

DESCRIPTION

Ports can be specified in many ways. For example, if you wanted to specify
slot 2, ports 4 through 7 (inclusive), you could use any of these
interchangeably: 2/4-7 or 2/4,5,6,7 or 2/4 2/5 2/6 2/7. If you don't specify
any ports, the entire slot is assumed. If you only have a blade in slot 2,
you could leave that off, e.g. 4-7 or 4,5,6,7 or 4 5 6 7. You can also
specify the following options: auto, fullduplex, speed, mtu, and
ignorepause. Example: "port reserve 1/0-3 mtu 1500 fullduplex false speed
100"
```

Configuration Options for the Port Command

When the configuration option `port_force_reserve` is set to yes, ports will be unreserved first if they are already reserved by someone else when you try to reserve them.

The configuration option `port_mtu` specifies the default MTU that will be used when reserving ports.

Specifying Slots and Ports

An asterisk (*) means "all slots and ports".

On a multi-slot system, a single number indicates a slot and all ports on that slot.

On a single-slot system, a single number indicates the port number.

Ports are specified by a slot number, a forward-slash, and a port number. For instance, 2/0 means "slot 2, port 0".

Multiple ports can be specified by space-separated slot/port combinations (2/0 2/1 2/2 2/3), by a hyphen-separated range (2/0-3), or as a comma-separated list (2/0,1,2,3).

Listing Port Status

The commands `port`, `port list`, and `port list *` are all synonymous.

Example

Review the following example to examine the syntax for displaying the status of all ports.

```
username@storm (group:1)% port

SLOT 1 - model: np_ctm_10g, status: ok

0:1 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
1:2 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
2:3 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
3:4 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)

SLOT 2 - model: np_ctm_1g, status: ok

0:1 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
1:2 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
2:3 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
3:4 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
4:5 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
5:6 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
6:7 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
7:8 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
```

The ports for each slot are listed underneath each slot model/status line.

The first number on the line is the port number, the number after the colon (if present) is the logical interface number.

The port's test group is displayed like this: <|1|>. If the link is down, it would look like this: |1| . If the port was not reserved, it would not have a group number or vertical bars, like this: <--->.

The username of the user who reserved the port (if any) is next, in parentheses.

If a test is running on the port, the test ID is next in square brackets, followed by the test name and progress percentage. If no test is running, then the port note (if any) would be shown.

To just show the status for ports 4 through 7 on slot 1:

```
username@storm (group:1)% port list
1/4-7

SLOT 1 - model: np_ctm_1g, status: ok

4:1 <|3|> (admin) [1439] 0-as-bw
(0.38%)
5:2 <|3|> (admin) [1439] 0-as-bw
(0.38%)
6:3 <|3|> (admin) [1439] 0-as-bw
(0.38%)
7:4 <|3|> (admin) [1439] 0-as-bw
(0.38%)
```

Reserving Ports

To reserve all ports on slot 2 (on a multiple slot system):

```
username@storm (group:1)% port
reserve 2
```

To reserve ports 4-7 on slot 2:

```
username@storm (group:1)% port reserve
2/4-7
```

Displaying Port Details

To display the details for slot 1, port 3:

```
username@storm (group:1)% port details
1/3

SLOT 1, PORT 3 (ok, link up)
Auto: false
Auto-Capable: false
Duplex: full
Ignore Pause Frames: false
Media: 10Gbase-SR
MTU: 9198
NP ID: 0
Speed: 10000 Mbit/s
```

Configuring Ports

Port configuration options include: auto, fullduplex, and speed.

Review the following example to examine the syntax for changing the MTU for slot 1 port 3.

```
username@storm (group:1)% port configure 1/3 mtu
1500
[+] setting mtu for slot 1 port 3 to 1500
username@storm (group:1)% port details 1/3

SLOT 1, PORT 3 (ok, link up)
Auto: false
Auto-Capable: false
Duplex: full
Ignore Pause Frames: false
Media: 10Gbase-SR
MTU: 1500
NP ID: 0
Speed: 10000 Mbit/s
```

Unreserving Ports

Use the following syntax to unreserve all ports on slot 2 (on a multi-slot system).

```
username@storm (group:1)% port
unreserve 2
```

Use the following syntax to unreserve ports 4-7 on slot 2.

```
username@storm (group:1)% port unreserve
2/4-7
```

Setting a Port Note

Review the following example to examine the syntax for setting a port note for slot 2, port 0.

```
username@storm (group:1)% port note 2/0 My port note
[+] setting port note for slot 2 port 0 to 'My port
note'
username@storm (group:1)% port list 2/0

SLOT 2 - model: np_ctm_1g, status: ok

0 <---> NOTE: My port note
```

Displaying a Port Note

Review the following example to examine the syntax for showing a port note (even when a test is running).

```
username@storm (group:1)% port note
2/0
SLOT 2, PORT 0 - NOTE: My port note
```

Removing a Port Note

Review the following example to examine the syntax for removing a port note for slot 2, port 0.

```
username@storm (group:1)% port note 2/0
remove
[+] removing port note for slot 2 port 0
```

Network Neighborhood

The following commands are described in this section:

Creating a Network Neighborhood

Use the `createNetwork` command to create a new Network Neighborhood or one based on an existing Network Neighborhood. This will create a network client object with subobjects that you can interact with to configure specific items. For more information on Network Neighborhoods, see the What is a Network Neighborhood? on page 103.

Syntax

Use the following syntax to create a Network Neighborhood:

```
set networkObject [$connectionObject createNetwork]; #creates a new network object
```

```
set networkObject [$connectionObject createNetwork -template {BreakingPoint Switching}]; #creates a network object using the BreakingPoint Switching Network Neighborhood as a template
```

The following table lists breaks down the elements of creating a Network Neighborhood.

Creating a Network Neighborhood

Element	Description
`networkObject`	A name for the network client object.
`createNetwork`	The command to create a Network Neighborhood.
`-template`	An attribute that allows you to use an existing Network Neighborhood as a template.
`existingNeighborhood`	The name of the Network Neighborhood that you would like to use as a template.
`-name`	An attribute that lets you set the name of the new Network Neighborhood.
`networkName`	The name of the Network Neighborhood you are creating.

Example

```
set emptynetwork [$bps createNetwork]; #
set n [$bps createNetwork -template {BreakingPoint Switching}]; #
itcl::delete object $emptynetwork; #
$n configure -name 00Test; #
$n save -name 00Test -force; #
$n save -file /tmp/foo -force; #saves a network neighborhood to a file
itcl::delete object $n; #deletes the network object and any subobjects
```

Saving a Network Neighborhood

After you have created a Network Neighborhood and configured the domains for them, you must save them before you can set the Network Neighborhood for a test context.

Syntax

Use the following syntax to save the Network Neighborhood.

```
$networkObject save
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true]; #creates the connection object
set network1 [$var createNeighborhood -template "BreakingPoint Switching" -name neighborhoodA]; #creates a Network Neighborhood object called network1 and a network neighborhood called neighborhoodA
$network1 addDomain 1 domainA; #adds a domain called domainA to interface 1
$network1 addSubnet 1 domainA {
netaddr 1.0.1.0
netmask 16
gateway 1.0.0.1
behind_snapt false
ranges {
{hosts 1.0.1.1 1.0.1.254 00:00:01:00:00:00}
}
innervlan 4} #creates a n
```

```
$network1 save; #saves the network neighborhood
```

```
$var setNeighborhood neighborhoodA; #sets the network neighborhood to
neighborhood A
```

Creating an IPv6 Network Neighborhood

Use the `createNeighborhood` command to create an IPv6 Network Neighborhood based on an existing Network Neighborhood. This will create a network client object that you can customize by adding domains and defining subnets. For more information on Network Neighborhoods, see the What is a Network Neighborhood? on page 103 section.

Note: The Network Neighborhood you create cannot be set as the Network Neighborhood for the test until you have saved it.

Syntax

Use the following syntax to create an IPv6 Network Neighborhood.

```
set networkObject [$connectionObject createNeighborhood-template
existingNeighborhood -name networkName]
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set nn [$var createNeighborhood ]
$nn addDomain 1 default

dict set subnet1 netaddr fde0:6477:1e3f::
dict set subnet1 behind_snapt false
dict set subnet1 type router
dict set subnet1 ip_v 6
dict set subnet1 ranges {fde0:6477:1e3f::1:1 fde0:6477:1e3f::1:ff}
dict set subnet1 innervlan {}
dict set subnet1 outervlan {}
dict set subnet1 netmask 64
dict set subnet1 l2 02:1a:c5:01:00:00
dict set subnet1 router_ip fde0:6477:1e3f::1:0

$nn addSubnet 1 default $subnet1
$nn getSubnets 1 default
```

Adding Domains to the Network Neighborhood

After you have created the network client object, you can add a domain to the Network Neighborhood. For more information on domains, see the section Component Tags on page 149.

Note: The system will implicitly add the interface that you specify, if it has not yet been created.

Syntax

Use the following syntax to add a domain to the Network Neighborhood:

```
$networkObject addDomain interface domainName
```

The following table breaks down the elements of adding a domain to a Network Neighborhood.

Adding a Domain

Element	Description
networkObject	The network client object you created for the Network Neighborhood
addDomain	Adds a domain to the Network Neighborhood
interface	Specifies the interface to which the domain will be added; the interface specified will be implicitly created if it does not exist
domainName	A name for the domain

Example

```
set var [bps::connect 10..10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set network1 [$var createNeighborhood -template "BreakingPoint Switching" -name
neighborhoodA]; #creates a Network Neighborhood object called network1 and a
network neighborhood called neighborhoodA
```

```
$network1 addDomain 1 domainA; #adds a domain called domainA to interface 1
```

Adding Subnets to a Domain

Once you have created your domains, you can add subnets to them. For more information on subnets, see the Network Neighborhood Subnets on page 151.

Syntax

Use the following syntax to add a subnet to a domain. This syntax can be used to create a domain that assigns a MAC address to each host address. This is equivalent to enabling the "Host" option in the Network Neighborhood.

```
$networkObject addSubnet interface domainName {
netaddr x.x.x.x
netmask x
gateway x.x.x.x
behind_snapt value
ranges {
{hosts x.x.x.x x.x.x.x xx:xx:xx:xx:xx:xx}
   }
innervlan n outervlan n
   }
```

Use the following syntax to add a subnet to a domain. This syntax can be used to create a domain that uses one MAC address for all host address. This is equivalent to enabling the "Virtual Router" option in the Network Neighborhood. If you want to have one IP address for the domain, use the same address for the minimum and maximum IP addresses. For example, if you only want all traffic from the domain to be 1.0.1.3, use the syntax: `ranges {{router 1.0.1.3 1.0.1.3 00:00:03:00:00:00}}`.

```
$networkObject addSubnet interface domainName {
netaddr x.x.x.x
netmask x
gateway x.x.x.x
behind_snapt value
ranges {
{router x.x.x.x x.x.x.x xx:xx:xx:xx:xx:xx x.x.x.x}
   }
innervlan n outervlan n
   }
```

The following table lists breaks down the elements of adding a subnet to a domain.

Adding a Subnet

Element	Description
networkObject	The object created for the Network Neighborhood.
addSubnet	A command that adds a subnet to the specified domain.
interface	The interface on which the domain you are adding the subnet to is found.
domainName	The name of the domain
netaddr x.x.x.x	The base network address
netmask n	The netmask for the network address

Element	Description
`gateway x.x.x.x`	The gateway address
`behind_snapt value`	Enables or disables Network Address Translation. The value for this attribute can either be true or false.
`ranges`	The range of addresses that will be used for host addressing.
`hosts {x.x.x.x x.x.x.x xx:xx:xx:xx:xx:xx}`	Sets the host type as hosts. This enables you to use one MAC address for each host address. The first set of IP addresses (x.x.x.x) represents the range of IP addresses for the subnet, and the second address (xx:xx:xx:xx:xx:xx) represent the base MAC address.
`router {x.x.x.x x.x.x.x xx:xx:xx:xx:xx:xx x.x.x.x}`	Sets the host type as virtual router. This enables you to use one MAC address for all host addresses. The first set of IP addresses (x.x.x.x) represent the range of IP addresses for the subnet; the second address (xx:xx:xx:xx:xx:xx) represent the base MAC address; and the last IP address (x.x.x.x) represents the virtual router's address.
`innervlan value`	The inner VLAN ID
`outervlan value`	The outer VLAN ID

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set network1 [$var createNeighborhood -template "BreakingPoint Switching" -name
neighborhoodA]; #creates a Network Neighborhood object called network1 and a
network neighborhood called neighborhoodA
```

```
$network1 addDomain 1 domainA; #adds a domain called domainA to interface 1
```

```
$network1 addSubnet 1 domainA {
netaddr 1.0.1.0
netmask 16
gateway 1.0.0.1
behind_snapt false
ranges {
{hosts 1.0.1.1 1.0.1.254 00:00:01:00:00:00}
}
innervlan 4} #creates a n
```

Adding Subnet Paths to a Network Neighborhood

Once you have added subnets, you can add defined paths for them.

Syntax

Use the following syntax to add a defined path for a subnet to a domain.

```
set n [$bps createNeighborhood]
$n addPath 1 default "" 2 default ""
$n removePath 1 default "" 2 default ""
```

Adding Interfaces to a Network Neighborhood

When you add a domain to an interface, the system will implicitly add the interface to the Network Neighborhood, if it does not already exist.

Example

```
$networkObject addDomain interface domainName
```

Configure an Existing Interface

Syntax

```
$ifObj configure <dictionary of property value>
main configuring interface options:
-id value <>
-number value Interface Number <>
-mtu value Maximum Transmission Unit (IP) <1500>
-use_vnic_mac_address value Use vNIC MAC Address <true>
-mac_address value Base 48-bit hexadecimal MAC address <>
-duplicate_mac_address value Select to use one MAC address for all hosts <>
-vlan_key value Determines whether to route packets based on inner or
outer vlan in double-tagged VLAN scenarios. <outer_vlan>
-ignore_pause_frames value Disregard received ethernet pause frames <false>
-description value User-supplied description of this element <>
-impairments.drop value Drops packets <false>
-impairments.frack value Separates the packet into 8-byte portions and
randomly removes portions from the packet <false>
-impairments.corrupt_lt64 value Corrupts packets only within the first 64 bytes of
the packet <false>
-impairments.corrupt_lt256 value Corrupts packets only between the 65th and the 256th
byte of the packet <false>
-impairments.corrupt_gt256 value Corrupts packets after the first 256 bytes <false>
-impairments.corrupt_rand value Corrupts packets in a random location within the
packet byte of the packet <false>
-impairments.corrupt_chksum value Creates an invalid checksum <false>
-impairments.rate value Percentage of packets corrupted <10>
-packet_filter.filter value <and>
-packet_filter.not_src_ip value <false>
-packet_filter.src_ip value <>
-packet_filter.not_dest_ip value <false>
-packet_filter.dest_ip value <>
-packet_filter.not_src_port value <false>
-packet_filter.src_port value <>
-packet_filter.not_dest_port value <false>
-packet_filter.dest_port value <>
-packet_filter.not_vlan value <false>
-packet_filter.vlan value <>
```

Example

```
#Use the getAll interface command to return a list of ID object pairs. Then you can configure the desired one from the dictionary
set n [$bps createNetwork -template {BreakingPoint Switching} ]
set interfaceDict [$n getAll interface]
puts "All interfaces are: [dict keys $interfaceDict]"
set ifObj [dict get $interfaceDict {Interface 2}]
puts "{Interface 2} parameters are : [$ifObj configure]"
#set -duplicate_mac_address to false
$ifObj configure -duplicate_mac_address false
```

Example

```
#get the interfaces in an array instead of a dict
array set interface [$n get interface]
puts "The interface ids are: [array names interface]"
#>The interface ids are: {Interface 1} {Interface 10} {I.....
puts "Interface 1 mtu [$interface(Interface 1) cget -mtu]"
```

Example

```
#another way is to get the interface object by id
set if1obj [$n get interface {Interface 1} ]; #retrieves a specific interface object by ID number
puts "Interface 1 mtu [$if1obj cget -mtu]"
puts "Interface 1 all settings [$if1obj configure]"
puts "Set Interface 1 mac address: "
$if1obj configure -mac_address AA:AA:AA:AA:AA:AA
```

Configure a New Interface in Network Neighborhood

Syntax

```
$network_object add interface -number <index of location > -mac_address <mac address>
```

Main add interface options are the same as for Configuring an Existing Interface in Network Neighborhood.

Example

```
#adding interface and grepping the returned new id and corresponding obj
lassign [$n add interface -number 21 -mac_address AA:AA:AA:AA:AA:AA -duplicate_mac_address false;] ifid ifObj
puts "Created interafce $ifid"
$ifObj configure -duplicate_mac_address false
```

Working With Network Elements

Use the `getAll` or `get` commands to retrieve network elements.

Note: All network element types follow the same format.

Example

```
set n [$bps createNetwork -template {BreakingPoint VLAN}]; #uses the BreakingPoint VLAN Network Neighborhood as a template to create a network client object
```

```
array get vlan
```

```
$n get vlan {VLAN i2v11_default}
```

To update the configuration, you must interact with the specific object.

Example

```
$vlan(VLAN i2v11_default) configure
```

```
$vlan(VLAN i2v11_default) configure -tpid
```

```
$vlan(VLAN i2v11_default) configure -tpid 8a88
```

```
$vlan(VLAN i2v11_default) cget -tpid
```

```
lassign [$n add interface -id "New Interface" -number 20] id obj; #Adds object by giving a type of object to add, and optionally some initial configuration items
```

```
$n remove interface "New Interface"; #Remove items by giving the type and ID
```

```
% $n begin
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj
% lassign [$n add interface] id obj; # Wraps multiple operations into one long
transaction
```

```
% $n revert; # Backs all of the changes out
```

```
% $n commit; # Commits the changes
```

The following is a list of the object types that are currently supported:

- interface
- vlan
- ip_dns_config
- ip6_dns_config
- ip_external_hosts
- ip6_external_hosts
- ip_static_hosts
- ip6_static_hosts
- ip_dhcp_hosts
- ip_dhcp_server
- ip_router
- ip6_router
- sixrd_ce
- enodeb_mme
- plmn
- mobility_session_info
- ue
- ggsn
- sgsn

- enodeb
- sgw_pgw

Note: Every element corresponds to a table in the user interface and shares the same configuration options as those in the user interface.

Listing Network Neighborhoods

The `listNeigbhorhoods` command will retrieve a listing of all Network Neighborhoods stored on the system; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display Network Neighborhoods created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint-created Network Neighborhoods, or `custom`, which will return a list of all user-created Network Neighborhoods. You will use the `-timeunit` and `-timeinterval` attributes to list Network Neighborhoods by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

Syntax

Use the following syntax to view a list of available Network Neighborhoods; this includes all canned and custom Network Neighborhoods.

```
$connectionObject
listNeighborhoods
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listNeighborhoods; #returns a list of all the Network Neighborhoods stored
on the system
```

```
$var listNeighborhoods -userid admin; #returns a list of all Network
Neighborhoods created by the admin
```

```
$var listNeighborhoods -class canned; #returns a list of all default Network
Neighborhoods
```

```
$var listNeighborhoods -timeunit day -timeinterval 2; #returns a list of all
Network Neighborhoods created two days ago
```

Removing a Network Interface

Use the `remove Interface` command to remove one or more interfaces. The highest numbered interfaces are always removed first.

Syntax

Use the following syntax to remove a network interface.

```
% $n
removeInterface
```

Example

```
% $n removeInterface -
count 4
16
```

Deleting an interface object does not automatically remove the interface. Interface objects are the exception, all other objects will remove themselves from the network when deleted. The reason for the exception is to keep interfaces in consecutive order by ID and to prevent IDs in the middle from being inadvertantly deleted.

Creating Component Tags

The commands associated with domains (including `setDomain`, `unsetDomain`, `getDomains`, and `getDomain`) are now deprecated. While these commands are still supported for backwards compatibility, BreakingPoint recommends that users cease from using them as support will end in a future release.

The commands associated with domains have been replaced with configuration options that support component tagging. The new configuration options are `-client_tags` and `-server_tags`. These configuration options accept a list of tag strings which refer to tags on endpoints (host sets or UEs) in the Network Neighborhood.

Note: BreakingPoint strongly recommendeds abandoning the use of the deprecated domain-based commands in favor of the component tag configuration methodology.

Example

```
% set t [$bps createTest -template AppSim]; #creates the test object and a test
based on the AppSim test
```

```
% array set c [$t getComponents]; #retrieves available component objects
```

```
% $c(appsim1) cget -client_tags; # returns the Component Tag currently used by
the test
```

```
% $c(appsim1) configure -server_tags {"first tag" "second tag" "third tag"};
#sets the -server_tags for the test as first tag, second tag, and third tag
```

The `setDomain` commands and the component tag configuration options cannot coexist in the same test. Once the `setDomain` command is used for a test, all component tags for that test will be removed.

Conversely, once the component tags are configured using the `configure -client_tags` or `configure -server_tags` configuration options, all of the domains for the test will be removed. By convention, when Network Neighborhoods are migrated to the new release, tests that were configured to use a particular interface and domain will now have a reference to the corresponding Component Tag. So an old test with Interface 1 checked with domain default set as a Client will now reference a client Component Tag of i1_default.

This allows those preexisting tests and Network Neighborhoods to continue to function exactly as they did before.

Example

```
A previous domain named mydomain defined on Interface 3 would have the
Component Tag i3_mydomain
```

So the most simplistic migration is to replace any code of this form:

```
$c(appsim1) setDomain client 1 default
$c(appsim1) setDomain client 2 clientdomain
$c(appsim1) setDomain server 3 serverdomain
$c(appsim1) setDomain server external
someexternal
# or more generally
$component setDomain $type $interface $domain
```

with code in this form:

```
$c(appsim1) configure -client_domains {i1_default i2_clientdomain}
$c(appsim1) configure -server_domains {i3_serverdomain ext_
someexternal}
# or more generally
if {$interface == "external"} {
$component configure -${type}_domains [list ext_$domain]
} else {
$component configure -${type}_domains [list i${interface}_$domain]
}
```

Test Paths

Test paths link one endpoint (hostset or UE) ID to another endpoint ID. Test paths are returned as a list of endpoint-endpoint pairs.

Example

```
$n getPaths
hs1 hs2
```

```
$n addPath hs1 hs3; #Test paths are added by giving an endpoint - endpoint pair
```

```
$n removePath hs1 hs3; # Test paths are removed by giving an endpoint -
endpoint pair. Note that order does not matter.
```

Configuring Element Types

The `type` command returns the element type of the object and adds settings. Settings returns a dictionary where the keys are configuration settings and the data provides user-readable descriptions of those settings.

Example

```
% lassign [$n add ip_dns_config] id obj
% $obj type
% dict keys [$obj settings]
% dict get [$obj settings] -id
% dict get [$obj settings] -dns_server_
address
% $obj configure -dns_server_address
```

Querying Element Types

Use the `elementType` command to determine which element types are available from the network client object. The command returns a dictionary where the keys are valid element types. The data provides user-readable information about that specific element type.

Syntax

Use the following syntax to view a list of element types available from the network client object.

```
$networkObject
elementTypes
```

Syntax Example

```
set n [$bps createNetwork]
dict keys [$n elementTypes]
dict get [$n elementTypes] vlan label
dict get [$n elementTypes] ip_dns_config label
dict get [$n elementTypes] ip_dns_config category
```

```
dict get [$n elementTypes] ip_dns_config
description
```

Detailed Example

```
set n [$bps createNetwork]; #creates the network client object
```

```
dict keys [$n elementTypes]; #returns more information on the valid element
types
```

```
dict get [$n elementTypes] vlan label; #returns information on the VLAN label
```

```
dict get [$n elementTypes] ip_dns_config label; #returns information on the
IPv4 DNS Configuration label
```

```
dict get [$n elementTypes] ip_dns_config category; #returns the category that
the element type IPv4 DNS Configuration belongs to
```

```
dict get [$n elementTypes] ip_dns_config description; #returns a description of
the IPv4 DNS Configuration element type
```

Listing Host Sets and UEs

In addition to using the subnet ID or the Interface ID to list host sets and UEs, you can also use the `-tags` configuration option to obtain a list of all available Component Tags. The `-tags` configuration option accepts a list of tags and locates all host sets that match any of the specified Component Tags.

Example

```
% $n getAll ip_static_hosts -tags i1_default; #returns a list of all host sets
labeled with the i1_default Component Tag
```

```
% $n getAll ip_external_hosts -tags ext_default; #returns a list of all host
sets labeled with the ext_default Component Tag
```

```
% $n getAll ue -tags i1_default; #returns a list of all UEs labeled with the
i1_default Component Tag
```

You can combine tags with other query options, in which case you get host sets that have one of the tags listed, and also meet the other criteria.

Example

```
% $n getAll ip_external_hosts -tags {ext_default i1_default} -interface 1;
#returns a list of host sets labeled with the ext_default or i1_default
Component Tag on Interface 1
```

Listing DUT Profiles

The `listDUTs` command will retrieve a listing of all DUT Profiles stored on the system; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display DUT Profiles created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint created DUT Profiles, or `custom`, which will return a list of all user-created DUT Profiles. You will use the `-timeunit` and `-timeinterval` attributes to list DUT Profiles by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

This command also accepts a Google-formatted search string as a final argument.

Syntax

Use the following syntax to view a list of available DUT Profiles; this includes all canned and custom DUT Profiles.

```
$connectionObject
listDUTs
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listDUTs; #returns a list of all the DUT Profiles stored on the system
```

```
$var listDUTs -userid admin; #returns a list of all DUT Profiles created by the
admin
```

```
$var listDUTs -class canned; #returns a list of all default DUT Profiles
```

```
$var listDUTs -timeunit day -timeinterval 2; #returns a list of all DUT
Profiles created two days ago
```

```
$var listDUTs -limit 3 -offset 10 "admin" ; #returns a list of no more than 3
DUT Profiles
```

Tests

The following commands are described in this section:

Creating Tests

The recommended way to create a test is by creating an object for it. The test object will provide you with the necessary commands to set the Network Neighborhood and DUT Profile, add test components, export reports, and manually stop the packet trace.

Using the test object, you can:

- Set the Network Neighborhood and DUT Profile for the test
- View the Network Neighborhood, DUT Profile, and test components for the test
- Add and create components to the test
- Export test results
- Stop the packet trace at a defined time
- Run the test

In order to perform these tasks, the test object provides you with the following commands:

`cancel`: Cancels the test

`cget -dut`: Returns the DUT Profile used by the test

`cget -neighborhood`: Returns the Network Neighborhood used by the test

`configure -category`: Defines the category for the test

`configure -description`: Defines the description for the test

`configure -dut`: Defines the DUT Profile for the test

`configure -name`: Defines the name for the test

`configure -neighborhood`: Allows you to configure the Network Neighborhood

`createComponent`: Allows you to create a component for the test

`getAggStats`: Returns the Ethernet-related stats from a test report

`getComponents`: Returns the components used by the test

`run`: Runs the test

`save`: Saves the test

`exportReport -location`: Exports the report to the location specified

`startPacketTrace`: Starts the packet capture

`stopPacketTrace`: Stops the packet capture from running

Syntax

Use the following syntax to create a test object and to use the `createTest` command to create a test.

```
set testObjectName [$connectionObject createTest-template "templateName" -name
"testName" ]
```

Example 1

The following is a simple example that reserves ports on BreakingPoint Storm and creates a test object.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

Example 2

The following is a more detailed example that creates a complete test set up – including setting the Network Neighborhood and DUT Profile, saving and running the test, and stopping the packet trace at a specified time interval.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

```
$test1 run -group $myGroupNumber -async ProcToPerformWhenRunIsDone after 2000;
#runs the test
```

```
$test1 stopPacketTrace; #stops collecting packets
```

Listing Tests

Use the `listTests` command to display a list of tests currently on the system. This includes all user-created and BreakingPoint supplied tests.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display tests created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint created tests, or `custom`, which will return a list of all user-created tests. Use the `-timeunit` and `-timeinterval` attributes to list tests by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

This command also accepts a Google-formatted search string as a final argument.

Syntax

Use the following syntax to list all tests on the system.

```
$connectionObject
listTests
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listTests; #returns a list of tests on the system
```

Viewing Test Workspace Summary

Use the following syntax to read the summary values of the Test Workspace page.

Example

```
set t [$bps createTest -template AppSim]
::bps::BPSConnection::bPSConnection0::testClient0

$t configure
{-description {This test generates standard TCP and UDP and network traffic.}
{This test generates standard TCP and UDP and network traffic.}} {-dut
{BreakingPoint Default} {BreakingPoint Default}} {-lockedBy {} {}}
{-maxFlowCreationRate 100 100} {-maximumConcurrentFlows 100 100} {-name
testClient0 testClient0} {-neighborhood {BreakingPoint Switching}
{BreakingPoint
Switching}} {-network {BreakingPoint Switching} {BreakingPoint Switching}}
{-requiredMTU 0 0} {-seedOverride {} {}} {-totalAddresses 100 100}
{-totalAttacks 100 100} {-totalBandwidth 100 100} {-totalMacAddresses 2 2}
{-totalSubnets 2 2} {-totalUniqueStrikes 0 0} {-totalUniqueSuperflows 15 15}

# new read-only options -totalUniqueSuperflows, -totalUniqueStrikes,
# -totalMacAddresses, -totalSubnets, -requiredMTU
$t cget -totalSubnets
2

# new configurable settings -maxFlowCreationRate, -totalBandwidth,
# -maximumConcurrentFlows, -totalAttacks, -totalAddresses
$t cget -totalBandwidth
100
$t configure -totalBandwidth 50
$t configure -totalBandwidth
-totalBandwidth 100 50
```

Viewing Test Results

Results for a test are saved on a component by component basis. To access the data from your test results, you will need to create an object that you can use to query your data. Once you are done with the object, you can use the `itcl::delete` command to remove the object.

Syntax

Use the following syntax to query test results.

```
set resultObjectName [$testComponentName result]
$resultObjectName values; #returns a list of all values
$resultObjectName values-interface interfaceNumber; #returns all values for an
interface
$resultObjectName get statName; #returns the value for a statistic
$resultObjectName interfaces; #returns all interfaces used by the component
$resultObjectName protocols; #returns a list of protocols for the test
component
$resultObjectName values-protocol protocolName; #returns all values available
for a specific protocol
$resultObjectName protocols-name statName; #returns all protocols that are
related to a specific result
```

The following table breaks down the elements for querying test results.

Querying Test Results

Element	Description
resultObjectName	The name for the results object
testComponentName	The name of the test component whose results the object will store
result	The command that returns a component's results
values	A command that returns a list of values for an option
-interface	An argument to the values command that allows you to obtain data about a specific interface
get statName	A command that can be used to retrieve the value for a statistic.
interfaceNumber	The interface from which you want data
-protocol	An attribute to the values command that allows you to obtain data about a specific protocol
protocolName	The protocol for which you want data
interfaces	A command that returns interfaces related to a specific result type
protocols	A command that returns protocols related to a specific result type
-name	An attribute that allows you to specify the name of the statistic whose value will be returned
statName	The name of the statistic (e.g., txAvgFrameSize or rxAvgFrameSize)

Example

```
set var [bps::connect 10.10.10.10 joe passwd -shortcuts true -name test1];
#creates a connection to the system and the default test context
```

```
bitblaster bb1 1 2; #adds a bit blaster component to the default context that
transmits from interface 1 to interface 2
```

```
bb1 configure -rateDist.min 900 -sizeDist.min 512; #sets the data rate to 900
and the frame size to 512 bytes
```

```
$var run; #runs the test
```

```
set bb1results [bb1 result]; #stores the results in an object called bb1results
```

```
$bb1results value; #returns a list of avaliable values for the component
```

```
$bb1results interfaces; #returns the interfaces that were used by the component
```

```
$bb1results values -interface 1; #returns the results for interface 1
```

```
$bb1results protocols; #returns a list of protocols used by the component
```

```
$bb1results protocols -name txFrameRate; #returns a list of protocols that are
related to the txFrameRate result
```

```
$bb1results interfaces -name txFrameRate; #returns a list of interfaces that
are related the txFrameRate result
```

Viewing Historical Results

After the completion of your test you can use the **historicalResult** command to get all statistics values for each available timestamp.

You will need to be familiar with the **aggstats** command in order to use the **historicalResult** command. See Viewing Aggregate Statistics for detailed information about the **aggstats** command. As a quick overview, here are some basic descriptions:

- **aggStats objectName** - Stores the aggregate statistics for a test in an object.
- **getAggStats objectName** - Returns the values stored in the object created using the aggStats command.

The **historicalResult** command requires 2 parameters, **group** and **index**.

- **Group** - The statistics category available for the test.

In order to see the available stats categories, use the**getGroups** option for aggStats objects. For example, **$aggStats getGroups**.

- **Index** – The timestamps available for the test.

In order to get the last timestamp of the test, you will need to use following commands:

- set aggStats [$test getAggStats]
- set aggRaport [$aggStats result]
- set lastTimeStamp [$aggRaport get timestamp]

historicalResult example:

Note that the numbered lines in this example describe the Tcl commands that follow them, they are not Tcl commands.

1. get aggregated stats
```
% set aggStats [$test getAggStats]
::componentClient25
```
2. get aggregated report
```
% set aggRaport [$aggStats result]
result0
```
3. get the test duration/last timestamp
```
% set lastTimeStamp [$aggRaport get timestamp]
144
```
4. (optional) you can set a list of timestamps that you want to get statistics for
```
% set timeStampList "1 7 15 [expr $lastTimeStamp-10]"
1 7 15 134
```
5. get the groups of statistics in order to identify the group where the desired statistic exists
```
% $aggStats getGroups;
arpstats dhcp6_server_stats dhcp6c_stats dhcpstats dslite_stats
flowexceptions gatewaystats gtpstats impairment ipsec_router_stats l2 l4
ldapstats ltestats multicast_router_stats natstats ndpstats routerstats
slaacstats system_usage default
```
6. get the statistics inside one group in order to find the exact name of the desired statistic
```
% [$aggStats historicalResult "l2" $lastTimeStamp] values;
ethAlignmentErrors ethDropEvents ethFCSErrors ethOversizedFrames ethRxErrors
ethRxFrameData ethRxFrameDataRate ethRxFrameRate ethRxFrames ethRxPauseFrames
ethTxErrors ethTxFrameData ethTxFrameDataRate ethTxFrameRate ethTxFrames
ethTxPauseFrames ethUndersizedFrames rx_timestamp timestamp
```
7. get the value of the desired statistic at the selected timestamp
```
% foreach time $timeStampList {
> puts "ethRxFrameDataRate at time: $time --> [[$aggStats historicalResult
"l2" $time] get ethRxFrameDataRate]"
> }
ethRxFrameDataRate at time: 1 --> 66961
ethRxFrameDataRate at time: 7 --> 80000
ethRxFrameDataRate at time: 15 --> 79999
ethRxFrameDataRate at time: 134 --> 79999
```

Importing Tests

Use the `importTest` command to import a test object and a test into the system. Additionally, you can use the `-force` attribute to overwrite any test with the same name.

Syntax

Use the following syntax to import a test object and a test from a file location:

```
$bps importTest <newTestName> -file
/location/
```

Use the following syntax to import a test file from a URL.

```
$bps importTest <filename> -url
http://www.google.com/
```

Use the following syntax to import a test and force it to overwrite an existing test with the same name. You can use the `-force` attribute to overwrite an existing file if you do not want to overwrite an existing file.

```
$bps importTest <newTestName> -file /location/0-0-del.bpt -
force
```

Example

The following example imports a file called MyTest.bpt from the temp location.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
$var importTest MyTest -file /temp/0-0-del.bpt -force; #imports MyTest and
overwrites any file with that same name
```

Viewing the DUT Profile for the Test

You cannot create DUT Profiles through the Tcl interface; however, you can use the `cget -dut` command to view the DUT Profile currently set for a test object. For more information on DUT Profiles, see the DUT Profiles on page 95 section.

Note: The default DUT Profile for all test objects is BreakingPoint Default.

Syntax

Use the following syntax to view the DUT Profile currently selected for the test object.

```
$testObject cget -
dut
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 cget -dut; #returns the DUT Profile used by the test
```

Setting the DUT Profile for the Test

You cannot create DUT Profiles through the Tcl interface; however, you can use the `configure -dut` command to change the DUT Profile for a test object. For more information on DUT Profiles, see the DUT Profiles on page 95 section.

Note: The default DUT Profile for all test objects is BreakingPoint Default.

Syntax

Use the following syntax to change the DUT Profile for the test.

```
$testObject configure -dut
DUTName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

Viewing the Network Neighborhood for the Test

You can use `cget -neighborhood` to view the Network Neighborhood currently used by the test.

Syntax

Use the following syntax to view the Network Neighborhood currently selected for the test.

```
$testObject cget -
neighborhood
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 cget -neighborhood; #returns the Network Neighborhood
```

Setting the Network Neighborhood for the Test

You can use `configure -neighborhood` to change the Network Neighborhood used by the test.

Syntax

Use the following syntax to change the Network Neighborhood for the test.

```
$testObject configure -neighborhood
neighborhoodName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood NN1; #sets the Network Neighborhood to NN1
```

Overriding the Seed

The Seed Override is used to modify the seed for the test. The Seed Override enables you to control whether static or dynamic content will be generated. Explicitly setting the seed allows the system to recreate the same application flows each time the Super Flow is run. Not explicitly setting the seed causes the system to automatically randomize a seed for the Super Flow each time it is used.

Seed Override

You can use `configure -seedOverride` to modify the seed for Security, Application Simulator, and Stack Scrambler tests.

Syntax

Use the following syntax to change the seed for the test object.

```
$testObject configure -
seedOverride
```

Example

```
set t [$bps createTest]
```

```
$t cget -seedOverride
```

```
$t configure -seedOverride
0;
```

```
$t configure -seedOverride
{}
```

Note: Note: For non-integer input, control returns an error.

Defining the Test Description

You can use `configure -description` to modify the description for the test.

Syntax

Use the following syntax to change the description for the test.

```
$testObject configure -
description
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -description "app sim test for switch" ; #sets the test
description
```

Creating a Test Component

The recommended way to create a test component is to create an object for the component. You will create the object in the usual way (set componentObject...) along with the test object command `createComponent`, which will add the component to the test.

Note: You can add multiple components to a test by creating a component object for each component.

When creating the component, you can specify the interfaces that will serve as the client and server interfaces. The interface values that can be used must be a value between 1-8. The value you assign to the client/server interface correlates to an interface in the Network Neighborhood.

After you specify which Network Neighborhood interface will be used for the client interface and which will be used for the server interface, you can choose the domains for the client and server interfaces. The client and server interfaces can either use the default domain or another domain from the Network Neighborhood.

If you want to use the default domains from the Network Neighborhood, you do not need to specify any additional information other than the client and server interface numbers. Otherwise, you will need to use the `setDomains` command to configure the domains to something other than the default domain.

Note:

- If you use the 'special' name #auto, then a unique name will automatically be generated for the object.
- It is recommended that you create custom components through the Control Center if you plan on making extensive modifications to a test component's parameters.Once you create your custom components through the Control Center, you can simply refer to them in your Tcl scripts.

Syntax

The following syntax uses the `createComponent` command, which is a command of the test object, to create an object for the component and the component itself. Replace componentName with a component from the list provided in section Test Components List below.

```
set componentObject [$testObject createComponent componentName clientInterface#
serverInterface#]
```

Test Components List

The following is a list of component names from which you can create components:

appsim

appsim_100k

appsim_200k

appsim_300k

appsim_50k

appsim_Max10K

appsim_Max1K

appsim_business_user

appsim_dsl_user

appsim_ed

appsim_enterprise

appsim_enterprise_datacenter

appsim_fast_http_10

appsim_fast_http_5

appsim_fast_http_6

appsim_fast_http_7

appsim_fast_http_9

appsim_isp

appsim_mobile_user

appsim_multicast_client

appsim_multicast_server

appsim_soho

appsim_ssl_http_1_0

appsim_ssl_http_1_1

appsim_ssl_http_bulk_perf

appsim_ssl_onearm_client_handshake_perf

appsim_ssl_onearm_client_http_request_perf

appsim_three_user_prof

appsim_wanacc

bitblaster

bitblaster_10000Mbps

bitblaster_1Gbps

bitblaster_5Gbps

bitblaster_imix

bitblaster_imix_jumbo

clientsim_fast_http_10

clientsim_fast_http_5

clientsim_fast_http_6

clientsim_fast_http_7

clientsim_fast_http_9

clientsimpreset

clientsimpreset_medium

recreate

routingrobot

routingrobot_1000

routingrobot_10G

routingrobot_5G

routingrobot_imix

routingrobot_imix_jumbo

routingrobot_imix_tcp

security

security_2

security_3

security_4

security_5

security_malware_smtp_base64

sessionsender

sessionsender_100

sessionsender_200

sessionsender_300

sessionsender_400

sessionsender_50

sessionsender_500

sessionsender_600

sessionsender_fast_http_10

sessionsender_fast_http_5

sessionsender_fast_http_6

sessionsender_fast_http_7

sessionsender_fast_http_9

sessionsender_http

sessionsender_icmp

sessionsender_large

sessionsender_max

sessionsender_max_bandwidth

sessionsender_medium

sessionsender_single_stream_1g

sessionsender_single_stream_5g

sessionsender_synflood

sessionsender_synflood_1G

sessionsender_udp_1000

stackscrambler

stackscrambler_gtp

stackscrambler_icmp

stackscrambler_ip

stackscrambler_tcp

stackscrambler_udp

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -name "myTest" ]; #creates the test object and an
empty test called MyTest
```

```
set comp1 [$test1 createComponent appsim #auto 1 2]; #creates a component
object called comp1 and an App Sim component which will be named using the
auto-naming function. The client interface is 1 and server interface is 2
```

```
set comp2 [$test1 createComponent sec #auto 1 2]; #creates a component object
called comp1 and a Security component which will be named using the auto-naming
function. The client interface is 1 and server interface is 2
```

Creating a TCP SYN Flood Packet Template

You can use `configure -Templates.TemplateType` to create a packet template to be used by the test.

Example

```
set bps [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c [$bps createComponent routingrobot #auto 1 2]; #creates a routing robot
component object that will be named using the auto-naming function. The client
interface is 1 and server interface is 2
```

```
$c configure -Templates.TemplateType
```

```
$c configure -Templates.TemplateType TcpSynFlood
```

```
$c configure -Templates.TemplateType
```

```
$c cget -Templates.TemplateType
```

Component Shortcut Commands

If the `-shortcuts` is set to true for the system, you will be able to use shortcut commands to create test components. By default, shortcuts are automatically enabled for all tests. Using these shortcut commands, you can create a component based on one of the system's default test components and their presets, or you can create a component based on a custom component (or preset) that you have created. You can use any shortcut command to create a component of that type. See the following table for a list of default test component shortcuts.

Note: You may want to set the `-shortcuts` attribute to false if you are connecting to more than one system within the same script.

Shortcut Commands

Shortcut Command	Description
appsim_ enterprise	Generates realistic application traffic flows that are representative of an enterprise network. For information on appsim's parameters, see the section appsim Default Configuration on page 1038.
appsim_Max10K	(10Gb only) Generates realistic application traffic flows designed to maximize throughput. For information on appsim_max's parameters, see the section appsim_Max10K Default Configuration on page 1041.
appsim_Max1K	(1Gb only) Generates realistic application traffic flows designed to maximize throughput. For information on appsim_max's parameters, see the section appsim_Max1K on page 1039.
appsim_ed	Generates realistic application traffic flows that are representative of a higher education network. For information on appsim_ed's parameters, see the Configuring Test Components on page 1081.
appsim_isp	Generates realistic application traffic flows that are representative of a service provider network. For information on appsim_isp's parameters, see the appsim_ isp Default Configuration on page 1043 section.
appsim_wanacc	Generates realistic application traffic flows in a distribution representative of a wide-area network of satellite offices. For information on appsim_wanacc's parameters, see the appsim_wanacc Default Configuration on page 1044 section.
bitblaster	Sends 500 Mbps of valid layer 2 Ethernet frames with contrived content. For information on bitblaster's parameters, see the section bitblaster Default Configuration on page 1049.
bitblaster_ 10000Mbps	(10Gb only) Sends 10,000 Mbps of valid layer 2 Ethernet frames with contrived content. For more information on bitblaster_10000Gbps' parameters, see the bitblaster_10000Mbps Default Configuration on page 1047section.

Shortcut Command	Description
bitblaster_ 5Gbps	(10Gb only) Sends 5 Gbps of valid layer 2 Ethernet frames with contrived content. For more information on bitblaster_5Gbps, see the bitblaster_5Gbps Default Configuration on page 1048section.
bitblaster_ 1Gbps	Sends 1 Gbps of valid layer 2 Ethernet frames with contrived content. For information on bitblaster_1Gbps' parameters, see the bitblaster_1Gbps Default Configuration on page 1050section.
clientsimpreset	Sends layer 4 traffic load that is similar to that of a small office behind router or NAT device. It opens no more than 500 concurrent sessions and is useful for testing small devices with limited memory and processing resources. For more information on clientsimpreset's parameters, see the section clientsimpreset Default Configuration on page 1051.
recreate	Replays captured traffic patterns based on application data from a PCAP file. For information on recreate's parameters, see the section recreate Default Configuration on page 1054.
routingrobot	Sends 500 Mbps of valid IP packets with contrived content. For information on routingrobot's parameters, see the section routingrobot Default Configuration on page 1055.
routingrobot_ 10G	(10Gb only) Sends 10,000 Mbps of valid IP packets with contrived content. For more information on routingrobot_10G's parameters, see the section routingrobot_10G Default Configuration on page 1057.
routingrobot_5G	(10Gb only) Sends 5,000 Mbps of valid IP packets with contrived content. For more information on routingrobot_5G's parameters, see the section routingrobot_ 5G Default Configuration on page 1059.
routingrobot_ 1000	Sends 1,000 Mbps of valid IP packets with contrived content. For information on routingrobot_1000's parameters, see the section routingrobot_1000 Default Configuration on page 1061.
security	Uses Security Level 1 to target high-risk vulnerabilities in services often exposed to the Internet. This includes approximately 100 Strikes. For information on security's parameters, see the section security Default Configuration on page 1063.
security_2	Uses Security Level 2 to target all high-risk vulnerabilities. This includes approximately 450 Strikes. For information on security_2's parameters, see the section security_2 Default Configuration on page 1064.

Shortcut Command	Description
security_3	Uses Security Level 3 to target all high-risk vulnerabilities, worms, and backdoors. This includes approximately 500 Strikes. For information on security_3's parameters, see the section security_3 Default Configuration on page 1064.
security_4	Uses Security Level 4 to target all vulnerabilities, worms, and backdoors. This includes approximately 750 strikes. For information on security_4's parameters, see the section security_4 Default Configuration on page 1065.
security_5	Uses Security Level 5 to target all vulnerabilities, worms, backdoors, probes, and denial of service flaws. This includes approximately 2,800 Strikes. For information on security_5's parameters, see the section security_5 Default Configuration on page 1065.
sessionsender	Simulates layer 4 traffic loads similar to that of a small office behind router or NAT device. For information on sessionsender's parameters, see the section sessionsender Default Configuration on page 1066.
sessionsender_http	Simulates a stateless HTTP client connecting to server port 80. To use this shortcut, you must use the External interface. For more information on sessionsender_http's parameters, see sessionsender_http Default Configuration on page 1068
sessionsender_large	Simulates layer 4 traffic load similar to that of a large network. For information on sessionsender_large's parameters, see the section sessionsender_large Default Configuration on page 1069.
sessionsender_max	Uses the maximum values supported to generate TCP sessions. You can only run one sessionsender_max preset per test. Sessionsender_max is intended to utilize all available resources for session-based components; therefore, if you want to use more than one Session Sender component that uses the sessionsender_max preset, then you must adjust the data rate to account for bandwidth limitations. For information on sessionsender_max's parameters, see the section sessionsender_max Default Configuration on page 1071.
sessionsender_medium	Simulates layer 4 traffic load similar to that of a medium-sized network. For information on sessionsender_mediums's parameters, see the section sessionsender_medium Default Configuration on page 1073.
sessionsender_synflood	Simulates a SYN flood for 60 seconds. For information on sessionsender_synflood's parameters, see the section sessionsender_synflood Default Configuration on page 1074.

Shortcut Command	Description
stackscrambler	Generates intentionally corrupt packets targeting TCP, UDP, and other IP protocol stacks. For information on stackscrambler's parameters, see the section stackscrambler Default Configuration on page 1078.
stackscrambler_tcp	Generates intentionally corrupt packets targeting TCP stacks. For information on stackscrambler_tcp's parameters, see the section stackscrambler_tcp Default Configuration on page 1079.
stackscrambler_udp	Generates intentionally corrupt packets targeting UDP stacks. For information on stackscrambler_udp's parameters, see the section stackscrambler_udp Default Configuration on page 1080.

appsim Default Configuration

The following table lists the parameters for appsim and their default values.

appsim Parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint Enterprise
rampDist.down	1
rampDist.downBehavior	full
rampDist.steady	28
rampDist.steadyBehavior	cycle
rampDist.up	1
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated

Parameter	Default Value
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	3,000,000
sessions.maxPerSecond	75,000
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

appsim_Max1K

(1Gb only) The following table lists the parameters for appsim_Max1K and their default configurations.

appsim_Max1K Parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint Bandwidth
rampDist.down	1
rampDist.downBehavior	full
rampDist.steady	28

Parameter	Default Value
rampDist.steadyBehavior	cycle
rampDist.up	1
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.max	N/A
rateDist.min	1,000
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	500,000
sessions.maxPerSecond	500,000
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	65,535
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter. (1Gb only)

appsim_Max10K Default Configuration

(10Gb only) The following table lists the parameters for appsim_Max10K and their default values.

appsim_Max10K Parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint Bandwidth
rampDist.down	1
rampDist.downBehavior	full
rampDist.steady	28
rampDist.steadyBehavior	cycle
rampDist.up	1
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	500,000
sessions.maxPerSecond	500,000
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true

Parameter	Default Value
tcp.delay_acks	false
tcp.initial_receive_window	65,535
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter (10Gb only).

appsim_ed Default Configuration

The following table lists the parameters for appsim_ed and their default values.

appsim_ed Parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint Higher Education
rampDist.down	11
rampDist.downBehavior	full
rampDist.steady	30
rampDist.steadyBehavior	cycle
rampDist.up	11
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if

Parameter	Default Value
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	4,000,000
sessions.maxPerSecond	400,000
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

appsim_isp Default Configuration

The following table lists the parameters for appsim_isp and their default values.

appsim_isp parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint Service Provider
rampDist.down	22
rampDist.downBehavior	full
rampDist.steady	30
rampDist.steadyBehavior	cycle

Parameter	Default Value
rampDist.up	22
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	2,500,000
sessions.maxPerSecond	125,000
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

appsim_wanacc Default Configuration

The following table lists the parameters for appsim_wanacc and their default values.

appsim_wanacc parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint WAN Acceleration
rampDist.down	11
rampDist.downBehavior	full
rampDist.steady	30
rampDist.steadyBehavior	cycle
rampDist.up	11
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	1,250
sessions.maxPerSecond	125
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false

Parameter	Default Value
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

appsim_soho Default Configuration

The following table lists the parameters for appsim_soho and their default values.

appsim_soho parameters

Parameter	Default Value
ip.tos	0
ip.ttl	32
profile	BreakingPoint SOHO/Small Business
rampDist.down	11
rampDist.downBehavior	full
rampDist.steady	30
rampDist.steadyBehavior	cycle
rampDist.up	11
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant

Parameter	Default Value
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	1,250
sessions.maxPerSecond	125
sessions.target	1
sessions.targetPerSecond	1
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

bitblaster_10000Mbps Default Configuration

(10Gb only) The following table lists the parameters for bitblaster_10000Mbps and their default values.

bitblaster_10000Mbps parameters

Parameter	Default Value
advanced.ethTypeField	constant
advanced.ethTypeVal	FFFF
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A

Parameter	Default Value
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	1,024
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true

bitblaster_5Gbps Default Configuration

(10Gb only) The following table lists the parameters for bitblaster_5Gbps and their default values.

bitblaster_5Gbps parameters

Parameter	Default Value
advanced.ethTypeField	constant
advanced.ethTypeVal	FFFF
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random
payloadAdvanced.udfDataWidth	eight

Parameter	Default Value
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	1,024
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true

bitblaster Default Configuration

The following table lists the parameters for bitblaster and their default values.

bitblaster parameters

Parameter	Default Value
advanced.ethTypeField	constant
advanced.ethTypeVal	FFFF
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random

Parameter	Default Value
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	1,024
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true

Note: *N/A denotes that no value has been defined for the parameter.

bitblaster_1Gbps Default Configuration

The following table lists the parameters for bitblaster_1Gbps and their default values.

bitblaster_1Gbps Parameters

Parameters	Default Values
advanced.ethTypeField	constant
advanced.ethTypeVal	FFFF
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A

Parameters	Default Values
payload.dataWidth	eight
payload.type	random
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	1,024
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true

Note: *N/A denotes that no value has been defined for the parameter.

clientsimpreset Default Configuration

The following table lists the parameters for clientsimpreset and their default values.

clientsimpreset parameters

Parameter	Default Value
delayStart	0
ip.tos	0
ip.ttl	32

Parameter	Default Value
rampDist.down	2
rampDist.downBehavior	full
rampDist.steady	60
rampDist.steadyBehavior	cycle
rampDist.up	11
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	100
sessions.maxPerSecond	500
sessions.target	1
sessions.targetPerSecond	1
superflow	BreakingPoint ClientSim HTTP
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3

Parameter	Default Value
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

clientsimpreset_medium Default Configuration

The following table lists the parameters for clientsimpreset_medium and their default values.

clientsimpreset parameters

Parameter	Default Value
delayStart	0
ip.tos	0
ip.ttl	32
rampDist.down	21
rampDist.downBehavior	full
rampDist.steady	60
rampDist.steadyBehavior	cycle
rampDist.up	11
rampDist.upBehavior	full
rampUpProfile.increment	0
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	200

Parameter	Default Value
sessions.maxPerSecond	2000
sessions.target	1
sessions.targetPerSecond	1
superflow	BreakingPoint ClientSim HTTP
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

recreate Default Configuration

The following table lists the parameters for recreate and their default values.

recreate parameters

Parameters	Default Value
behavior	user
file	BreakingPoint Capture Sample
ip.tos	0
ip.ttl	32
rampDist.down	0
rampDist.downBehavior	full
rampDist.steady	30
rampDist.steadyBehavior	cycle
rampDist.up	0
rampDist.upBehavior	full
rampUpProfile.increment	0

Parameters	Default Value
rampUpProfile.interval	1
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	aggregate
rateDist.type	constant
rateDist.unit	mbps
sessions.closeFast	false
sessions.max	100
sessions.maxPerSecond	125,000
sessions.target	0
sessions.targetPerSecond	0
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

routingrobot Default Configuration

The following table lists the parameters for routingrobot and their default values.

routingrobot parameters

Parameter	Default Value
advancedIPVersion	
advancedIPv4.checksumField	actual

Parameter	Default Value
advancedIPv4.checksumVal	N/A
advancedIPv4.lengthField	actual
advancedIPv4.lengthVal	N/A
advancedIPv4.optionHeaderData	N/A
advancedIPv4.optionHeaderField	disabled
advancedIPv4.tos	0
advancedIPv4.ttl	32
advancedIPv6.extensionHeaderData	N/A
advancedIPv6.extensionHeaderField	disabled
advancedIPv6.flowLabel	0
advancedIPv6.hopLimit	32
advancedIPv6.lengthField	actual
advancedIPv6.lengthVal	0
advancedIPv6.nextHeader	0
advancedIPv6.prefixData	2001
advancedIPv6.trafficClass	0
advancedUDP.checksumField	actual
advancedUDP.checksumVal	N/A
advancedUDP.lengthField	actual
advancedUDP.lengthVal	N/A
dstPort	1
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight

Parameter	Default Value
payload.type	random
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	512
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true
srcPort	1

Note: *N/A denotes that no value has been defined for the parameter.

routingrobot_10G Default Configuration

(10Gb only) The following table lists the parameters for routingrobot_10G and their default values.

routingrobot_10G parameters

Parameter	Default Value
advancedIPVersion	
advancedIPv4.checksumField	actual
advancedIPv4.checksumVal	N/A

Parameter	Default Value
advancedIPv4.lengthField	actual
advancedIPv4.lengthVal	N/A
advancedIPv4.optionHeaderData	N/A
advancedIPv4.optionHeaderField	disabled
advancedIPv4.tos	0
advancedIPv4.ttl	32
advancedIPv6.extensionHeaderData	N/A
advancedIPv6.extensionHeaderField	disabled
advancedIPv6.flowLabel	0
advancedIPv6.hopLimit	32
advancedIPv6.lengthField	actual
advancedIPv6.lengthVal	0
advancedIPv6.nextHeader	0
advancedIPv6.prefixData	2001
advancedIPv6.trafficClass	0
advancedUDP.checksumField	actual
advancedUDP.checksumVal	N/A
advancedUDP.lengthField	actual
advancedUDP.lengthVal	N/A
dstPort	1
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random

Parameter	Default Value
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	512
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true
srcPort	1

routingrobot_5G Default Configuration

(10Gb only) The following table lists the parameters for routingrobot_5G and their default values.

routingrobot_5G parameters

Parameter	Default Value
advancedIPVersion	
advancedIPv4.checksumField	actual
advancedIPv4.checksumVal	N/A
advancedIPv4.lengthField	actual
advancedIPv4.lengthVal	N/A

Parameter	Default Value
advancedIPv4.optionHeaderData	N/A
advancedIPv4.optionHeaderField	disabled
advancedIPv4.tos	0
advancedIPv4.ttl	32
advancedIPv6.extensionHeaderData	N/A
advancedIPv6.extensionHeaderField	disabled
advancedIPv6.flowLabel	0
advancedIPv6.hopLimit	32
advancedIPv6.lengthField	actual
advancedIPv6.lengthVal	0
advancedIPv6.nextHeader	0
advancedIPv6.prefixData	2001
advancedIPv6.trafficClass	0
advancedUDP.checksumField	actual
advancedUDP.checksumVal	N/A
advancedUDP.lengthField	actual
advancedUDP.lengthVal	N/A
dstPort	1
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A

Parameter	Default Value
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	512
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true
srcPort	1

routingrobot_1000 Default Configuration

The following table lists the parameters for routingrobot_1000 and their default values.

routingrobot_1000 parameters

Parameter	Default Value
advancedIPVersion	IPv4
advancedIPv4.checksumField	actual
advancedIPv4.checksumVal	N/A
advancedIPv4.lengthField	actual
advancedIPv4.lengthVal	N/A
advancedIPv4.optionHeaderData	N/A
advancedIPv4.optionHeaderField	disabled

Parameter	Default Value
advancedIPv4.tos	0
advancedIPv4.ttl	32
advancedIPv6.extensionHeaderData	N/A
advancedIPv6.extensionHeaderField	disabled
advancedIPv6.flowLabel	0
advancedIPv6.hopLimit	32
advancedIPv6.lengthField	actual
advancedIPv6.lengthVal	0
advancedIPv6.nextHeader	0
advancedIPv6.prefixData	2001
advancedIPv6.trafficClass	0
advancedUDP.checksumField	actual
advancedUDP.checksumVal	N/A
advancedUDP.lengthField	actual
advancedUDP.lengthVal	N/A
dstPort	1
duration.durationTime	00:00:30
duration.durationFrames	N/A
payload.data	N/A
payload.dataWidth	eight
payload.type	random
payloadAdvanced.udfDataWidth	eight
payloadAdvanced.udfLength	N/A
payloadAdvanced.udfMode	disabled
payloadAdvanced.udfOffset	N/A

Parameter	Default Value
rateDist.increment	N/A
rateDist.rate	N/A
rateDist.type	constant
rateDist.unit	mbps
sizeDist.increment	N/A
sizeDist.max	N/A
sizeDist.min	512
sizeDist.rate	N/A
sizeDist.type	constant
sizeDist.unit	frame
slowStart	true
srcPort	1

Note: *N/A denotes that no value has been defined for the parameter.

security Default Configuration

The following table lists the parameters for security and their default values.

security Parameters

Parameters	Default Value
attackPlan	Strike Level 1
attackRetries	0
attackTimeoutSeconds	
evasionProfile	Default evasion settings
maxAttacksPerSecond	
maxPacketsPerSecond	
paramOverrides	N/A
randomSeed	

Note: *N/A denotes that no value has been defined for the parameter.

security_2 Default Configuration

The following table lists the parameters for security_2 and their default values.

security_2 Parameters

Parameters	Default Value
Parameters	Default Value
attackPlan	Strike Level 2
attackRetries	0
attackTimeoutSeconds	
evasionProfile	Default evasion settings
maxAttacksPerSecond	
maxPacketsPerSecond	
paramOverrides	N/A
randomSeed	

Note: *N/A denotes that no value has been defined for the parameter.

security_3 Default Configuration

The table below lists the parameters for security_3 and their default values.

security_3 Parameters

Parameters	Default Value
attackPlan	Strike Level 3
attackRetries	0
attackTimeoutSeconds	
evasionProfile	Default evasion settings
maxAttacksPerSecond	
maxPacketsPerSecond	
paramOverrides	N/A

Parameters	Default Value
randomSeed	

Note: *N/A denotes that no value has been defined for the parameter.

security_4 Default Configuration

lists the parameters for security_4 and their default values.

security_4 Parameters

Parameters	Default Value
attackPlan	Strike Level 4
attackRetries	0
attackTimeoutSeconds	0.25
evasionProfile	Default evasion settings
maxAttacksPerSecond	0
maxPacketsPerSecond	0
paramOverrides	N/A
randomSeed	0

Note: *N/A denotes that no value has been defined for the parameter.

security_5 Default Configuration

The table below lists the parameters for security_5 and their default values.

security_5 Parameters

Parameters	Default Value
attackPlan	Strike Level 5
attackRetries	0
attackTimeoutSeconds	0.25
evasionProfile	Default evasion settings
maxAttacksPerSecond	0
maxPacketsPerSecond	0

Parameters	Default Value
paramOverrides	N/A
randomSeed	0

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender Default Configuration

The following table lists the parameters for sessionsender and their default values.

sessionsender Parameters

Parameter	Default Value
delayStart	0
dstPortDist.max	1023
dstPortDist.min	6
dstPortDist.type	random
ip.tos	0
ip.ttl	32
loadprofile	none
packetsPerSession	100
payload.transport	TCP
payload.type	random
payloadSizeDist.min	1472
payloadSizeDist.type	constant
ramp.Dist.	
rampDist.down	0
rampDist.downBehavior	full
rampDist.steady	60
rampDist.steadyBehavior	cycle
rampDist.up	0

Parameter	Default Value
rampDist.upBehavior	full
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	5000000
sessions.maxPerSecond	500000
sessions.target	1
sessions.targetPerSecond	1
sessions.closeFast	false
srcPortDist.max	61000
srcPortDist.min	32768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5792
tcp.mss	1460
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender_http Default Configuration

The following table lists the parameters for sessionsender_http and their default values.

sessionsender_http parameters

Parameter	Default Value
dstPortDist.max	N/A
dstPortDist.min	80
dstPortDist.type	constant
ip.tos	0
ip.ttl	32
packetsPerSession	1
payload.data	N/A
payload.type	http
payloadSizeDist.max	1,400
payloadSizeDist.min	1,400
payloadSizeDist.type	constant
rampDist.down	1
rampDist.downBehavior	full
rampDist.steady	28
rampDist.steadyBehavior	cycle
rampDist.up	1
rampDist.upBehavior	full
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated

Parameter	Default Value
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	500
sessions.maxPerSecond	1,000
sessions.target	1
sessions.targetPerSecond	1
srcPortDist.max	61,000
srcPortDist.min	32,768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

sessionsender_large Default Configuration

The following table lists the parameters for sessionsender_large and their default values.

sessionsender_large Parameters

Parameter	Default Value
dstPortDist.max	1023
dstPortDist.min	6
dstPortDist.type	random
ip.tos	0
ip.ttl	32

Parameter	Default Value
packetsPerSession	2
payload.data	N/A
payload.type	random
payloadSizeDist.max	1,280
payloadSizeDist.min	256
payloadSizeDist.type	random
rampDist.down	
rampDist.downBehavior	full
rampDist.steady	50
rampDist.steadyBehavior	cycle
rampDist.up	5
rampDist.upBehavior	full
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	1,000,000
sessions.maxPerSecond	125,000
sessions.target	1
sessions.targetPerSecond	1
srcPortDist.max	61,000

Parameter	Default Value
srcPortDist.min	32,768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender_max Default Configuration

The following table lists the parameters for sessionsender_max and their default values.

sessionsender_max Parameters

Parameter	Default Value
dstPortDist.max	1023
dstPortDist.min	6
dstPortDist.type	random
ip.tos	0
ip.ttl	32
packetsPerSession	20
payload.data	N/A
payload.type	constant
payloadSizeDist.max	N/A
payloadSizeDist.min	1,448
payloadSizeDist.type	random
rampDist.down	

Parameter	Default Value
rampDist.downBehavior	full
rampDist.steady	40
rampDist.steadyBehavior	cycle
rampDist.up	10
rampDist.upBehavior	full
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	5,000,000 (1Gb) or 20,000,000 (10Gb)
sessions.maxPerSecond	500,000 (1Gb) or 750,000 (10Gb)
sessions.target	1
sessions.targetPerSecond	1
srcPortDist.max	61,000
srcPortDist.min	32,768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3

Parameter	Default Value
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender_medium Default Configuration

The following table lists the parameters for sessionsender_medium and their default values.

sessionsender_medium Parameters

Parameter	Default Value
dstPortDist.max	10,23
dstPortDist.min	6
dstPortDist.type	random
ip.tos	0
ip.ttl	32
packetsPerSession	20
payload.data	N/A
payload.type	random
payloadSizeDist.max	1,280
payloadSizeDist.min	256
payloadSizeDist.type	random
rampDist.down	
rampDist.downBehavior	full
rampDist.steady	28
rampDist.steadyBehavior	cycle
rampDist.up	1
rampDist.upBehavior	full
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A

Parameter	Default Value
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	50,000
sessions.maxPerSecond	50,000
sessions.target	1
sessions.targetPerSecond	1
srcPortDist.max	61,000
srcPortDist.min	32,768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender_synflood Default Configuration

The following table lists the parameters for sessionsender_synflood and their default values.

sessionsender_synflood Parameters

Parameter	Default Value
dstPortDist.max	1023

Parameter	Default Value
dstPortDist.min	6
dstPortDist.type	random
ip.tos	0
ip.ttl	32
packetsPerSession	20
payload.data	N/A
payload.type	random
payloadSizeDist.max	1,280
payloadSizeDist.min	256
payloadSizeDist.type	random
rampDist.down	
rampDist.downBehavior	full
rampDist.steady	0
rampDist.steadyBehavior	hold
rampDist.up	60
rampDist.upBehavior	syn
rampUpProfile.increment	N/A
rampUpProfile.interval	N/A
rampUpProfile.max	N/A
rampUpProfile.min	N/A
rampUpProfile.type	calculated
rateDist.scope	per_if
rateDist.type	constant
rateDist.unit	mbps
sessions.max	500,000

Parameter	Default Value
sessions.maxPerSecond	500,000
sessions.target	0
sessions.targetPerSecond	0
srcPortDist.max	61,000
srcPortDist.min	32,768
srcPortDist.type	random
tcp.add_timestamps	true
tcp.delay_acks	false
tcp.initial_receive_window	5,792
tcp.mss	1,448
tcp.retries	3
tcp.retry_quantum_ms	250

Note: *N/A denotes that no value has been defined for the parameter.

sessionsender_udp Default Configuration

The following table lists the parameters for sessionsender_udp and their default values.

sessionsender_udp Parameters

Parameter	Default Value
srcPortDist.max	61000
srcPortDist.min	32768
srcPortDist.type	random
dstPortDist.max	1023
dstPortDist.min	6
dstPortDist.type	random
payload.type	random
payloadSizeDist.type	constant

Parameter	Default Value
payloadSizeDist.min	1472
rateDist.scope	per_if
rateDist.unit	mbps
rateDist.type	constant
sessions.max	5000000
sessions.maxPerSecond	500000
sessions.target	1
sessions.targetPerSecond	1
sessions.closeFast	false
ip.ttl	32
ip.tos	0
tcp.mss	1460
tcp.retry_quantum_ms	250
tcp.retries	3
tcp.delay_acks	false
tcp.initial_receive_window	5792
tcp.add_timestamps	true
rampUpProfile.type	calculated
rampDist.up	0
rampDist.upBehavior	full
rampDist.steady	60
rampDist.steadyBehavior	cycle
rampDist.down	0
rampDist.downBehavior	full
packetsPerSession	0

Parameter	Default Value
packetsPerSession.enabled	false
packetsPerSession.auth	none
packetsPerSession.min	sslv3
packetsPerSession.max	tlsv1
packetsPerSession.clientsessionreuse	100
packetsPerSession.clientsessiontimeout	300
packetsPerSession.cipher	RSA_AES_128_SHA
loadprofile	None
delayStart	0

stackscrambler Default Configuration

The following table lists the parameters for stackscrambler and their default values.

stackscrambler parameters

Parameter	Valid Values
badChecksum	1
badIPOptions	0
badIPVersion	1
badTCPOptions	0
badUrgentPointer	1
dstPort	0
duration.durationTime	hours, minutes, seconds
duration.durationFrames	0 – 1,000,000,000
fragments	10
handshakeTCP	false
prng.offset	0
prng.seed	0

Parameter	Valid Values
rateDist.type	constant
rateDist.unit	mbps
sizeDist.max	1,500
sizeDist.min	46
sizeDist.type	random
sizeDist.unit	packet
srcPort	0
targetStack	All

Note: *N/A denotes that no value has been defined for the parameter.

stackscrambler_tcp Default Configuration

The following table lists the parameters for stackscrambler_tcp and their default values.

stackscrambler_tcp parameters

Parameter	Valid Values
badChecksum	1
badIPOptions	0
badIPVersion	1
badTCPOptions	0
badUrgentPointer	1
dstPort	0
duration.durationTime	hours, minutes, seconds
duration.durationFrames	0 – 1,000,000,000
fragments	10
handshakeTCP	false
prng.offset	0
prng.seed	0

Parameter	Valid Values
rateDist.type	constant
rateDist.unit	mbps
sizeDist.max	1500
sizeDist.min	46
sizeDist.type	random
	packet
srcPort	0
targetStack	TCP

Note: *N/A denotes that no value has been defined for the parameter.

stackscrambler_udp Default Configuration

The following table lists the parameters for stackscrambler_udp and their default values.

stackscrambler_udp parameters

Parameter	Valid Values
badChecksum	1
badIPOptions	0
badIPVersion	1
badTCPOptions	0
badUrgentPointer	1
dstPort	0
duration.durationTime	hours, minutes, seconds
duration.durationFrames	0 – 1,000,000,000
fragments	10
handshakeTCP	false
prng.offset	0
prng.seed	0

Parameter	Valid Values
rateDist.type	constant
rateDist.unit	mbps
sizeDist.max	1,500
sizeDist.min	46
sizeDist.type	random
	packet
srcPort	0
targetStack	UDP

Note: *N/A denotes that no value has been defined for the parameter.

Configuring Test Components

Use the `configure` command to define the values for the parameters in a test component. For a list of test component parameters and their descriptions, see the section Component Parameters on the next page.

Note: By calling `$componentName configure`, you can see a list of the component's configurable parameters.

Syntax

```
$componentName configure -option
value
```

The following table lists breaks down the elements of configuring test components.

Configuring Test Components

Element	Description
`componentName`	The name of the object created for the test component.
`$comp configure`	Returns the default and current values for each parameter.
`configure`	The command that allows you to configure the parameters for a test component or can be used to return all parameters and their values for a component.
`-option`	The name of the parameter you want to modify.

Element	Description
value	The value you want to set the parameter to.

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -name "myTest"]; #creates an empty test
```

```
$test1 configure -name Neighborhood1; #sets the Network Neighborhood for the
test to be Neighborhood 1
```

```
set c1 [$test1 createComponent appsim_enterprise #auto 1 2]; # creates an App
Sim component with client interface 1 and server interface 2
```

```
$c1 configure -rateDist.min 900; # sets the data rate to 900 mbps
```

```
set c2 [$test1 createComponent security #auto 1 2]; # creates a Security
component with client interface 1 and server interface 2
```

```
$c2 configure -attackPlan "Strike Level 2"; sets the Attack Plan to Strike
Level 2
```

```
$test1 save; #saves the test
```

```
$test1 run; #runs the test
```

Component Parameters

The following sections lists the parameters available for each test component. These parameters can be used to configure a test component in a Tcl script.

Example

```
$comp1 configure; returns the parameters and their values for the test
component in comp1
```

Application Simulator Parameters

Application Simulator Parameters on the facing page lists the parameters for Application Simulator and their valid values. When referencing these options (or parameters) in the Tcl interface, each

parameter is preceded with a dash (‘-’).

Application Simulator Parameters

Parameter	Description	Valid Values
ip.tos	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – ff
ip.ttl	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
loadprofile	Sets the Load Profile that will be used for the test. If a Load Profile is selected, then all session-related parameters defined on the Parameters tab will be ignored.	None BreakingPoint 10K Maximum Megabits per second BreakingPoint Maximum Simultaneous Sessions BreakingPoint 1K Maximum Megabits per second BreakingPoint 1K Maximum Simultaneous Sessions BreakingPoint Default BreakingPoint Maximum Sessions per second
profile	Sets the Application Profile that determines the mix of application that will be used in the test traffic.	A BreakingPoint Application Profile or a custom Application Profile
rampDist.down	Sets the amount of time open sessions have to close.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.downBehavior	Sets how the component will close sessions during the ramp down phase.	**full** – The full TCP session close is performed. **half** – The full TCP session close is performed, but the final ACK is omitted. **rst** – Close all sessions by sending TCP RST (reset) packets.
rampDist.steady	Sets the amount of time sessions have to open, send data, and close. The system will have to maintain the number of open session for this time period.	0 – 1,000,000
rampDist.steadyBehavior	Determines how sessions are handled during the steady-state phase.	**cycle** – Sessions are closed as they finish sending data, and new sessions are opened. **hold** – No existing sessions opened during Ramp Up are closed. cycle + rst – Once the session has finished sending data, it will wait for the server to close the session. After the server has closed the session, the client will send a RST.
rampDist.up	Sets the duration for which new sessions can be opened.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.upBehavior	Determines how sessions are opened during the ramp up phase.	**full** – The full TCP handshake is performed. **full + data** – The full TCP handshake is performed, and data will be sent once the session opens. **full + data + close** – The full TCP handshake is performed, and data will be sent once the session options. Sessions are closed as they finish sending data and new sessions are opened. **half** – The full TCP handshake is performed, but the final ACK is omitted. **syn** – Only SYN packets are sent. **data flood** – Only PSH data packets are sent. In this mode, the state machine is bypassed, so no connections are set up; therefore, the ACKs will be be invalid. Use this mode for testing QoS routing, not stateful DUTs.
rampUpProfile.increment	Sets the number of connections that the connection establishment rate will increment by for the time specified for rampUpProfile.interval.	1 – 500,000
rampUpProfile.interval	Sets the time interval that rampUpProfile.increment will use to increment the connection establishment rate.	1 – 1,000,000

Parameter	Description	Valid Values
rampUpProfile.max	Sets the maximum connection establishment rate that will the system will attempt to reach during the ramp up phase. Once the system reaches this rate, it will continue to hold this rate until the ramp up phase ends.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.min	Sets the minimum connection establishment rate that will be used to start the ramp up phase.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.type	Determines whether the connection establishment rate is a constant rate or an incremental rate – as defined by the user.	calculated – The connection establishment rate will be a constant rate. It is calculated by taking the maximum number of sessions divided by the ramp up time. This option allows the component to attempt to reach the maximum number of connections during the ramp up phase. If this option is used, all other parameters listed under the Ramp Up Profile group will be disabled. stair step – The connection establishment rate is an incremental rate, and it will be determined by the values input for Minimum Connection Rate, Maximum Connection Rate, Increment N connections per second, and Every N seconds. This option sets the minimum connection rate that the system will start with and increment until it reaches the maximum connection rate or until the ramp up phase ends. If the maximum connection rate is met, the system will hold that rate until the ramp up phase is over.

Parameter	Description	Valid Values
rateDist.scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	**per_if** – Uses the data rate as the limit for the transmitting and receiving interfaces. **aggregate** – Uses the data rate as an aggregate limit for the test component.
rateDist.type	Sets how data rates are determined.	**Constant** – Uses rateDist.min as the data rate. **Random** – Selects a random value between rateDist.min and rateDist.max as the data rate. **Range** – Starts at rateDist.min and increments until it reaches rateDist.max. The system uses an algorithm that determines the incremental value that will increase rateDist.min value. until it reaches rateDist.max.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sessions.closeFast	Disables the limit on the close rate. If this option is set to false, then the close rate is limited by the connection establishment rate. If this option is set to true, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
sessions.max	Sets the maximum number of concurrent sessions that can be set up by the system at a given time.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)

Parameter	Description	Valid Values
sessions.maxPerSecond	Sets the maximum number of connections that can occur per second.	1 – 750,000* (10Gb) or 500,000 (1Gb)
sessions.target	The number of sessions that must open to pass the test.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
sessions.targetPerSecond	The number of sessions per second that must be reached to pass the test.	1 – 750,000* (10Gb) or 500,000 (1Gb)
tcp.add_timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
tcp.delay_acks	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
tcp.initial_receive_ window	Sets the size of the initial receive window for a new connection.	1 – 65,535
tcp.mss	Sets the maximum segment size that is used during the ramp up phase.	512 – 9,146
tcp.retries	Sets the number of times a connection is attempted before it is canceled.	1 – 3
tcp.retry_quantum_ms	Sets the amount of time that elapses before a connection is retried.	100 – 2,000

Parameter	Description	Valid Values
app.emphasis	Sets whether applications will generate more complex, dynamic traffic, or will generate simpler, possibly more performant, traffic.	performance or realism

Bit Blaster Parameters

The following table lists the parameters for Bit Blaster and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash ('-').

Bit Blaster Parameters

Parameter	Description	Valid Values
advanced.ethTypeField	Sets how the component will define Ethernet Length Type field for each packet.	**constant** – Uses the value defined for advanced.ethTypeVal in the Ethernet Length Type field. **length** – Uses the packet length in the Ethernet length/type field. Bit Blaster will only substitute the Ethernet length/type field with the packet's length if the packet is less than or equal to 1,500 bytes in length and VLAN tagging is not used.
advanced.ethTypeVal	Determines the value that will be placed in the Ethernet Length Type field if advanced.ethTypeField is set to length.	2E – FFFF Values less than 2E will be replaced with 2E.
duration.durationTime	Sets the duration of the test.	hours, minutes, seconds
duration.durationFrames	Sets the length of the test in frames.	0 – 1,000,000,000
payload.data	Defines the payload; this parameter is defined only if Payload.Type is set to User Defined. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)

Parameter	Description	Valid Values
payload.dataWidth	Defines the width of the data (in bits) being inserted into the payload.	eight, sixteen, or thirty-two
payload.type	Sets how the payload is determined.	zeros – Payload is 0s. **ones** – Payload is all 1s. **random** – Payload is defined using random Hex values. **increment** – Payload is defined using ascending values starting at 0. **decrement** – Payload is defined using descending values starting at 0xff. **predefined** – Payload is defined with standard hexadecimal notation. If the payload is smaller than the packet size, the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is larger than the packet size, the value will be truncated.
payloadAdvanced.udfDataWidth	Defines the width of the data (in bits) being incremented or decremented.	eight, sixteen, or thirty-two
payloadAdvanced.udfLength	Defines the UDF length (in bytes).	1 – 9,202

Parameter	Description	Valid Values
payloadAdvanced.udfMode	Sets how the component will overwrite the existing payload.	disabled– No data or counter is inserted. *counter– Inserts a 1-to-4 byte counter that increments every frame. The counter uses the value defined for UDF length. *random– Inserts a 1-to-end-of-payload sequence of random values. *increment– Increments the payload starting at 0. Inserts a 1-to-end-of-payload sequence of incrementing values using an 8, 16, or 32 bit width. *decrement– Decrements the payload starting at 0xff. Inserts a 1-to-end-of-payload sequence of decrementing values using an 8, 16, or 32 bit width. *Define payloadAdvanced.udfDataWidth, payloadAdvanced.udfLength , and payloadAdvanced.udfOffset to use this option.
payloadAdvanced.udfOffset	Defines the number of bytes from the beginning of the payload to place the UDF data.	0 – 9,201
rateDist.increment	Sets the rate at which the data rate will increase or decrease. This parameter is used in conjunction with rateDist.rate.	-10,000 to 10,000

Parameter	Description	Valid Values
rateDist.rate	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with rateDist.increment.	1 – 30
rateDist.type	Sets how data rates are determined.	**constant** – Uses rateDist.min as the data rate. **random** – Selects a random value between rateDist.min and rateDist.max as the data rate. **range** – Starts at rateDist.min and increments until it reaches rateDist.max. Once the maximum value is met, the data rate will restart at minimum value.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sizeDist.increment	Sets the number of bytes to increase or decrease the packet size by; this parameter is used in conjunction with sizeDist.rate.	-128 to 128
sizeDist.max	Sets the maximum frame/packet size; this parameter is used only if sizeDist.type is set to range.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)
sizeDist.min	Sets the minimum frame/packet size, if sizeDist.type is set to constant. Otherwise, this is the minimum value used if sizeDist.type is set to range or random.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)

Parameter	Description	Valid Values
sizeDist.rate	Sets the time increment (in seconds) for increasing or decreasing the packet size; this parameter is used in conjunction with sizeDist.increment.	1 – 30
sizeDist.type	Sets how frame/packet sizes are determined.	**Constant** – Uses sizeDist.min for the frame/packet size. **Random** – Selects a random value between sizeDist.min and sizeDist.max for the frame/packet size. **Range** – Starts at sizeDist.min and increments until it reaches sizeDist.max. Once the maximum value is met, the packet/frame size will restart at the minimum value.
sizeDist.unit	Sets whether Bit Blaster uses frame or packets.	packet or frame
slowStart	Specifies whether the component can send a small amount of traffic to the DUT before ramping up to the full rate of the test. This allows switching devices to identify which port to send test traffic.	true or false

Client Simulator Parameters

The following table lists the parameters for Client Simulator and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash ('-').

Client Simulator Parameters

Parameter	Description	Valid Values
ip.tos	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – ff
ip.ttl	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
loadprofile	Sets the Load Profile that will be used for the test. If a Load Profile is selected, then all session-related parameters defined on the Parameters tab will be ignored.	None BreakingPoint 10K Maximum Megabits per second BreakingPoint Maximum Simultaneous Sessions BreakingPoint 1K Maximum Megabits per second BreakingPoint 1K Maximum Simultaneous Sessions BreakingPoint Default BreakingPoint Maximum Sessions per second
superflow	Determines the type of traffic that will be included in the test.	A pre-configured BreakingPoint Super Flow or a custom Super Flow
profile	Sets the Application Profile that determines the mix of application that will be used in the test traffic.	A BreakingPoint Application Profile or a custom Application Profile
rampDist.down	Sets the amount of time open sessions have to close.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.downBehavior	Sets how the component will close sessions during the ramp down phase.	**full** – The full TCP session close is performed. **half** – The full TCP session close is performed, but the final ACK is omitted. **rst** – Close all sessions by sending TCP RST (reset) packets.
rampDist.steady	Sets the amount of time sessions have to open, send data, and close. The system will have to maintain the number of open session for this time period.	0 – 1,000,000
rampDist.steadyBehavior	Determines how sessions are handled during the steady-state phase.	**cycle** – Sessions are closed as they finish sending data, and new sessions are opened. **hold** – No existing sessions opened during Ramp Up are closed. **cycle + rst** – Once the session has finished sending data, it will wait for the server to close the session. After the server has closed the session, the client will send a RST.
rampDist.up	Sets the duration for which new sessions can be opened.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.upBehavior	Determines how sessions are opened during the ramp up phase.	**full** – The full TCP handshake is performed. **full + data** – The full TCP handshake is performed, and data will be sent once the session opens. **full + data + close** – The full TCP handshake is performed, and data will be sent once the session options. Sessions are closed as they finish sending data and new sessions are opened. **half** – The full TCP handshake is performed, but the final ACK is omitted. **syn** – Only SYN packets are sent. **data flood** – Only PSH data packets are sent. In this mode, the state machine is bypassed, so no connections are set up; therefore, the ACKs will be be invalid. Use this mode for testing QoS routing, not stateful DUTs.
rampUpProfile.increment	Sets the number of connections that the connection establishment rate will increment by for the time specified for **rampUpProfile.interval**.	1 – 500,000
rampUpProfile.interval	Sets the time interval that **rampUpProfile.increment** will use to increment the connection establishment rate.	1 – 1,000,000

Parameter	Description	Valid Values
rampUpProfile.max	Sets the maximum connection establishment rate that will the system will attempt to reach during the ramp up phase. Once the system reaches this rate, it will continue to hold this rate until the ramp up phase ends.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.min	Sets the minimum connection establishment rate that will be used to start the ramp up phase.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.type	Determines whether the connection establishment rate is a constant rate or an incremental rate – as defined by the user.	**calculated** – The connection establishment rate will be a constant rate. It is calculated by taking the maximum number of sessions divided by the ramp up time. This option allows the component to attempt to reach the maximum number of connections during the ramp up phase. If this option is used, all other parameters listed under the Ramp Up Profile group will be disabled. **stair step** – The connection establishment rate is an incremental rate, and it will be determined by the values input for Minimum Connection Rate, Maximum Connection Rate, Increment N connections per second, and Every N seconds. This option sets the minimum connection rate that the system will start with and increment until it reaches the maximum connection rate or until the ramp up phase ends. If the maximum connection rate is met, the system will hold that rate until the ramp up phase is over.

Parameter	Description	Valid Values
rateDist.scope	Uses the value defined for the data rate as the limit for the transmitting and receiving interfaces or as the aggregate limit for the test component.	**per_if** – Uses the data rate as the limit for the transmitting and receiving interfaces. **aggregate** – Uses the data rate as an aggregate limit for the test component.
rateDist.type	Sets how data rates are determined.	**Constant** – Uses **rateDist.min** as the data rate. **Random** – Selects a random value between **rateDist.min** and **rateDist.max** as the data rate. **Range** – Starts at **rateDist.min** and increments until it reaches **rateDist.max**. The system uses an algorithm that determines the incremental value that will increase **rateDist.min value**. until it reaches **rateDist.max**.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sessions.closeFast	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false

Parameter	Description	Valid Values
sessions.max	Sets the maximum number of concurrent sessions that can be set up by the system at a given time.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
sessions.maxPerSecond	Sets the maximum number of connections that can occur per second.	1 – 750,000* (10Gb) or 500,000 (1Gb)
sessions.target	The number of sessions that must open to pass the test.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
sessions.targetPerSecond	The number of sessions per second that must be reached to pass the test.	1 – 750,000* (10Gb) or 500,000 (1Gb)
tcp.add_timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false
tcp.delay_acks	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
tcp.initial_receive_ window	Sets the size of the initial receive window for a new connection.	1 – 65,535
tcp.mss	Sets the maximum segment size that is used during the ramp up phase.	512 – 9,146
tcp.retries	Sets the number of times a connection is attempted before it is canceled.	1 – 3

Parameter	Description	Valid Values
tcp.retry_quantum_ms	Sets the amount of time that elapses before a connection is retried.	100 – 2,000

Recreate Parameters

The following table lists the parameters for Recreate and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash ('-').

Recreate Parameters

Parameter	Description	Valid Values
behavior	Determines whether the Recreate test component uses the data in the capture file or the parameters defined for the component.	**file** – Uses the settings within the capture file to recreate traffic. **user** – Uses the Recreate parameters options to recreate traffic. Only the payload will be used.
file	The PCAP file the system will use to pull application payloads.	PCAP file name
ip.tos	Configures the TOS field used for all IP packets	0 – ff
ip.ttl	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255

Parameter	Description	Valid Values
loadprofile	Sets the Load Profile that will be used for the test. If a Load Profile is selected, then all session-related parameters defined on the Parameters tab will be ignored.	None BreakingPoint 10K Maximum Megabits per second BreakingPoint Maximum Simultaneous Sessions BreakingPoint 1K Maximum Megabits per second BreakingPoint 1K Maximum Simultaneous Sessions BreakingPoint Default BreakingPoint Maximum Sessions per second
rampDist.down	Sets the amount of time open sessions have to close.	0 – 1,000,000
rampDist.downBehavior	Sets how the component will close sessions during the ramp down phase.	**full** – The full TCP session close is performed. **half** – The full TCP session close is performed, but the final ACK is omitted. **rst** – Close all sessions by sending TCP RST (reset) packets.
rampDist.steady	Sets the amount of time sessions have to open, send data, and close. The system will have to maintain the number of open session for this time period.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.steadyBehavior	Determines how sessions are handled during the steady-state phase.	**cycle** – Sessions are closed as they finish sending data, and new sessions are opened. **hold** – No existing sessions opened during Ramp Up are closed. **cycle + rst** – Once the session has finished sending data, it will wait for the server to close the session. After the server has closed the session, the client will send a RST.
rampDist.up	Sets the duration for which new sessions can be opened.	0 – 1,000,000
rampDist.upBehavior	Determines how sessions are opened during the ramp up phase.	**full** – The full TCP handshake is performed. **full + data** – The full TCP handshake is performed, and data will be sent once the session opens. **full + data + close** – The full TCP handshake is performed, and data will be sent once the session options. Sessions are closed as they finish sending data and new sessions are opened. **half** – The full TCP handshake is performed, but the final ACK is omitted. **syn** – Only SYN packets are sent. **data flood** – Only PSH data packets are sent. In this mode, the state machine is bypassed, so no connections are set up; therefore, the ACKs will be invalid. Use this mode for testing QoS routing, not stateful DUTs.

Parameter	Description	Valid Values
rampUpProfile.increment	Sets the number of connections that the connection establishment rate will increment by for the time specified for **rampUpProfile.interval**.	1 – 500,000
rampUpProfile.interval	Sets the time interval that **rampUpProfile.increment** will use to increment the connection establishment rate.	1 – 1,000,000
rampUpProfile.max	Sets the maximum connection establishment rate that will the system will attempt to reach during the ramp up phase. Once the system reaches this rate, it will continue to hold this rate until the ramp up phase ends.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.min	Sets the minimum connection establishment rate that will be used to start the ramp up phase.	1 – 750,000* (10Gb) or 500,000 (1Gb)

Parameter	Description	Valid Values
rampUpProfile.type	Determines whether the connection establishment rate is a constant rate or an incremental rate – as defined by the user.	**calculated** – The connection establishment rate will be a constant rate. It is calculated by taking the maximum number of sessions divided by the ramp up time. This option allows the component to attempt to reach the maximum number of connections during the ramp up phase. **stair step** – The connection establishment rate is an incremental rate, and it will be determined by the values input for Minimum Connection Rate, Maximum Connection Rate, Increment N connections per second, and Every N seconds. This option sets the minimum connection rate that the system will start with and increment until it reaches the maximum connection rate or until the ramp up phase ends. If the maximum connection rate is met, the system will hold that rate until the ramp up phase is over.
rateDist.rate	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with **rateDist.increment**.	1 – 30
rateDist.type	Sets how data rates are determined.	**constant** – Uses **rateDist.min** as the data rate. **random** – Selects a random value between **rateDist.min** and **rateDist.max** as the data rate. **range** – Starts at **rateDist.min** and increments until it reaches **rateDist.max**. Once the maximum value is met, the data rate will restart at minimum value.

Parameter	Description	Valid Values
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sessions.closeFast	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
sessions.max	Sets the maximum number of concurrent sessions that can be set up by the system at a given time.	1 – 20,000,000 (10Gb) or 5,000,000 (1Gb)
sessions.maxPerSecond	Sets the maximum number of sessions that can occur per second.	1 – 750,000* (10Gb) or 500,000 (1Gb)
sessions.target	The number of sessions that must open to pass the test.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)
sessions.targetPerSecond	The number of connections per second that must be reached to pass the test.	1 – 750,000* (10Gb) or 500,000 (1Gb)
tcp.add_timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false

Parameter	Description	Valid Values
tcp.delay_acks	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
tcp.initial_receive_ window	Sets the size of the initial receive window for a new connection.	1 – 65,535
tcp.mss	Sets the maximum segment size that is used during the ramp up phase.	512 – 9,146
tcp.retries	Sets the number of times a connection is attempted before it is canceled.	1 – 3
tcp.retry_quantum_ms	Sets the amount of time that elapses before a connection is retried.	100 – 2,000

Routing Robot Parameters

The following table lists the parameters for Routing Robot and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash ('-').

Routing Robot Parameters

Parameter	Description	Valid Values
advancedIPVersion	Enables IPv4 or IPv6 support.	IPv4 or IPv6
advancedIPv4.checksumField	Sets how the Checksum field in the IP header is determined.	**Actual** – Uses the correct checksum in the Checksum field of the IP header. **Constant** – Uses advancedIPv4.checksumVal in the Checksum field of the IP header.

Parameter	Description	Valid Values
advancedIPv4.checksumVal	Defines the Total Length field of the IP header when **advancedIPv4.checksumField** is **constant**.	0 – FFFF
advancedIPv4.lengthField	Sets how the Total Length field in the IP header is determined.	**actual** – Uses the correct IP datagram length in the Total Length field of the IP header. **constant** – Uses advancedIPv4.lengthVal in the Total Length field of the IP header.
advancedIPv4.lengthVal	Defines the Total Length field of the IP header when **advancedIPv4.lengthField** is **constant**.	0 – 255
advancedIPv4.optionHeaderData	Defines the IPv4 option data, if **advancedIPv4.optionHeaderField** is **enabled**.	Hexadecimal value (up to 56 bytes of data)
advancedIPv4.optionHeaderField	Allows up to 56 bytes of IP option data to be specified. If this parameter is disabled, the UDP header will follow the IPv4 header.	enabled or disabled
advancedIPv4.tos	Configures the TOS field used for all IP packets.	0 – ff
advancedIPv4.ttl	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
advancedIPv6.extensionHeaderData	Defines the IPv6 extension header (s), if **advancedIPv6.extensionHeaderField** is **enabled**.	Hexadecimal value (up to 56 bytes of data)

Parameter	Description	Valid Values
advancedIPv6.extensionHeaderField	Allows up to 56 bytes to be specified for the IPv6 extension header(s). If this parameter is enabled, IPv6.Next header and IPv6.Extension header data must be defined. If this parameter is disabled, the UDP header will follow the IPv6 header.	enabled or disabled
advancedIPv6.flowLabel	Configures the Flow label field used for all IP packets. Values of 0 through FFFF (hexadecimal) are supported.	0 – FFFF
advancedIPv6.hopLimit	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255
advancedIPv6.lengthField	Sets how the Payload Length field in the IP header is determined.	**actual** – Uses the correct IP datagram length in the Packet Length field of the IP header. **constant** – Uses advancedIPv6.lengthVal in the Packet Length field of the IP header.
advancedIPv6.lengthVal	Defines the Packet Length field of the IP header when **advancedIPv6.lengthField** is **constant**.	0 – 65,535

Parameter	Description	Valid Values
advancedIPv6.nextHeader	Defines the Next header in the IPv6 header if **advancedIPv6.extensionHeaderField** is **Enabled**. This is the extension header that will appear first in the Extension header data. Configure this value to 11 to indicate a UDP payload.	0 – ff
advancedIPv6.prefixData	Sets the IPv6 address prefix using a hexadecimal value.	16 – 96 bits of Hexadecimal characters
advancedIPv6.trafficClass	Defines the Traffic Class field used for all IP packets	0 – FF
advancedUDP.checksumField	Determines the value that is placed into the checksum field of the UDP header.	**actual** – Uses the correct UDP checksum in the checksum field of the UDP header. **constant** – Uses the value defined for **UDP.Checksum value** in the checksum field of the UDP header. Using a constant UDP checksum may cause the test results to report invalid IP checksums.
advancedUDP.checksumVal	Defines the value that is used in the checksum field of the UDP header. This parameter is defined only if **advancedUDP.lengthField** is set to **constant**.	0 – FFFF

Parameter	Description	Valid Values
advancedUDP.lengthField	Determines the UDP datagram length that is placed in the length field of the UDP header.	**actual** – Uses the correct UDP datagram length in the length field of the UDP header. **constant** – Uses the value defined for **UDP. Length value** in the length field of the UDP header.
advancedUDP.lengthVal	Defines the UDP datagram length that is placed in the length field of the UDP header. This parameter is defined only if **advancedUDP.lengthField** is set to **constant**.	0 – 65,535
dstPort	Establishes the UDP port to which packets are addressed.	1 – 65,535
dstPortMask	Defines how the bits will be masked for each packet. This mask is right-justified and only applies to UDP destination ports.	1 – 16
duration.durationTime	Sets the duration of the test.	hours, minutes, seconds
duration.durationFrames	Sets the length of the test in frames.	1 – 1,000,000,000
payload.data	Defines the payload; this parameter is defined only if **payload.type** is set to **predefined**. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)
payload.dataWidth	Defines the width of the data (in bits) being inserted into the payload.	eight, sixteen, or thirty-two

Parameter	Description	Valid Values
payload.type	Sets how the payload is determined.	zeros – Payload is 0s. **ones** – Payload is all 1s. **random** – Payload is defined using random Hex values. **increment** – Payload is defined using ascending values starting at 0. **decrement** – Payload is defined using descending values starting at 0xff. **predefined** – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
payloadAdvanced.udfDataWidth	Defines the width of the data (in bits) being incremented or decremented.	eight, sixteen, or thirty-two
payloadAdvanced.udfLength	Defines the UDF length (in bytes).	1 – 9,174

Parameter	Description	Valid Values
payloadAdvanced.udfMode	Sets how the component will overwrite the existing payload.	**disabled**– No data or counter is inserted. ***counter**– Inserts a 1-to-4 byte counter that increments every frame. The counter uses the value defined for UDF length. ***random**– Inserts a 1-to-end-of-payload sequence of random values. ***increment**– Increments the payload starting at 0. Inserts a 1-to-end-of-payload sequence of incrementing values using an 8, 16, or 32 bit width. ***decrement**– Decrements the payload starting at 0xff. Inserts a 1-to-end-of-payload sequence of decrementing values using an 8, 16, or 32 bit width. *Define payloadAdvanced.udfDataWidth, payloadAdvanced.udfLength and payloadAdvanced.udfOffset to use this option.
payloadAdvanced.udfOffset	Defines the number of bytes from the beginning of the payload to place the UDF data.	0 – 9,173
rateDist.increment	Sets the rate at which the data rate will increase or decrease. This parameter is used in conjunction with **rateDist.rate**.	-10,000 to 10,000

Parameter	Description	Valid Values
rateDist.rate	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with **rateDist.increment**.	1 – 30
rateDist.type	Sets how data rates are determined.	**constant** – Uses **rateDist.min** as the data rate. **random** – Selects a random value between **rateDist.min** and **rateDist.max** as the data rate. **range** – Starts at **rateDist.min** and increments until it reaches **rateDist.max**. Once the maximum value is met, the data rate will restart at minimum value.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sizeDist.increment	Sets the number of bytes to increase or decrease the packet size by; this parameter is used in conjunction with **sizeDist.rate**.	-128 to 128
sizeDist.max	Sets the maximum frame/packet size; this parameter is used only if **sizeDist.type** is set to **range**.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)
sizeDist.min	Sets the minimum frame/packet size, if **sizeDist.type** is set to **constant**. Otherwise, this is the minimum value used if **sizeDist.type** is set to **range** or **random**.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)

Parameter	Description	Valid Values
sizeDist.rate	Sets the time increment (in seconds) for increasing or decreasing the packet size; this parameter is used in conjunction with **sizeDist.increment**.	1 – 30
sizeDist.type	Sets how frame/packet sizes are determined.	**Constant** – Uses **sizeDist.min** for the frame/packet size. **Random** – Selects a random value between sizeDist.min and sizeDist.max for the frame/packet size. **Range** – Starts at sizeDist.min and increments until it reaches sizeDist.max. Once the maximum value is met, the packet/frame size will restart at the minimum value.
sizeDist.unit	Sets whether Bit Blaster uses frame or packets.	packet or frame
slowStart	Specifies whether the component can send a small amount of traffic to the DUT before ramping up to the full rate of the test. This allows switching devices to identify which port to send test traffic.	true or false
srcPort	Establishes the UDP port from which packets are addressed.	1 – 65,535
srcPortMask	Defines how the bits will be masked for each packet. This mask is right-justified and only applies to UDP source ports.	1 – 16

Parameter	Description	Valid Values
udpDstPortMode	Determines how the UDP destintation port is modified.	**constant** – Uses the port value defined for Source Port. **random** – Selects a random port value between 1 and 65,535. **increment** – Starts at the **Destination Port** value and increments the port value by 1. **decrement** – Starts at the **Destination Port** value and decrements the port value by 1.
udpSrcPortMode	Determines how the UDP source port is modified.	**constant** – Uses the port value defined for **Source Port**. **random** – Selects a random port value between 1 and 65,535. **increment** – Starts at the **Source Port** value and increments the port value by 1. **decrement** – Starts at the **Source Port** value and decrements the port value by 1.

Security Parameters

The following table lists the parameters for Security and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash (‘-’).

Security Parameters

Parameter	Description	Valid Values
attackPlan	Sets the Strike List the Security component will use to derive its attacks.	A Strike List
attackRetries	Sets the number of times to attempt an attack before determining that the DUT successfully blocked the attack	0 – 100
attackTimeoutSeconds	Sets the amount of time the system will wait for a packet to arrive at its destination before resending the attack or determining that the DUT successfully blocked the attack.	0 – 3,600
evasionProfile	Sets the default evasion options for the Strikes.	An Evasion Profile
maxAttacksPerSecond	Sets the maximum number of attacks sent every second.	0 – 100,000
maxConcurrAttacks	The maximum number of Strikes that will run simultaneously: Single Strike - Will only run one strike at a time. Default - Will run up to five strikes concurrently.	1 – 5
maxPacketsPerSecond	Sets the maximum number of packets sent per second	0 – 1,000
paramOverrides	Overrides any of the evasion options set through the Evasion Setting or the Strike List.	N/A
randomSeed	Determines whether the test will generate static or dynamic attacks. '0' will randomize the content of each strike in the strike series. Any other value defined here will keep the strike content static.	0 – 4,294,967,295

Session Sender Parameters

The following table lists the parameters for Session Sender and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash ('-').

Session Sender Parameters

Parameter	Description	Valid Values
dstPortDist.max	Sets the maximum destination port number, if dstPortDist.type is Range or Random.	0 – 65,535

Parameter	Description	Valid Values
dstPortDist.min	Sets the minimum destination port number, if dstPortDist.type is range or random. Otherwise, this will be the value used for the destination port.	0 – 65,535
dstPortDist.type	Sets how the component will obtain the destination ports for TCP connections.	**constant** – Uses dstPortDist.min as the source port. **random** – Uses random values between **dstPortDist.min** and **dstPortDist.max**. **range** – Increments dstPortDist.min by one until it reaches dstPortDist.min. Once the port number reaches the maximum destination port number, it will reset to the minimum destination port number.
ip.tos	Configures the TOS field used for all IP packets.	0 – ff
ip.ttl	Sets the maximum bound on the number-of-hops that an IP datagram can exist in an internet system before it is dropped.	0 – 255

Parameter	Description	Valid Values
loadprofile	Sets the Load Profile that will be used for the test. If a Load Profile is selected, then all session-related parameters defined on the Parameters tab will be ignored.	None BreakingPoint 10K Maximum Megabits per second BreakingPoint Maximum Simultaneous Sessions BreakingPoint 1K Maximum Megabits per second BreakingPoint 1K Maximum Simultaneous Sessions BreakingPoint Default BreakingPoint Maximum Sessions per second
packetsPerSession	Specifies how many data packets are sent during an open session.	1 – 1,000
payload.data	Defines the payload; this parameter is defined only if **payload.type** is set to **predefined**. This value is inserted after the Ethernet header.	Hex values (numbers: 0 – 9, letters: a – f)
payload.transport	Sets the protocol for Session Sender	TCP UDP ICMP UDP Lossy – A payload type of UDP Lossy indicates that UDP packets that are not received due to packet loss are not counted as errors. All – (Combines TCP, UDP, and ICMP)

Parameter	Description	Valid Values
payload.type	Sets how the payload is determined.	zeros – Payload is 0s. **ones** – Payload is all 1s. **random** – Payload is defined using random Hex values. **http** – Payload consists of a simple HTTP 1.0 GET request for the '/' URL, padded to match the payload size distribution. **predefined** – Payload is defined by the user using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
payloadSizeDist.max	Sets the maximum UDP payload and TCP segment size.	0 – 9,416
payloadSizeDist.min	Sets the minimum UDP payload and TCP segment size.	0 – 9,416
payloadSizeDist.type	Sets how the component will define the UDP payload and the TCP segment size.	**constant** – All payloads will use the size defined for payloadSizeDist.min. **range** – All payloads will use the size defined for payloadSizeDist.min and increment to the size defined for payloadSizeDist.max. The system uses an algorithm that determines the incremental value that will increase payloadSizeDist.min so that it reaches payloadSizeDist.max. **random** – All payloads will have sizes that are randomly chosen between payloadSizeDist.min and payloadSizeDist.max.
rampDist.down	Sets the amount of time open sessions have to close.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.downBehavior	Sets how the component will close sessions during the ramp down phase.	**full** – The full TCP session close is performed. **half** – The full TCP session close is performed, but the final ACK is omitted. **rst** – Close all sessions by sending TCP RST (reset) packets.
rampDist.steady	Sets the amount of time sessions have to open, send data, and close. The system will have to maintain the number of open session for this time period.	0 – 1,000,000
rampDist.steadyBehavior	Determines how sessions are handled during the steady-state phase.	**cycle** – Sessions are closed as they finish sending data, and new sessions are opened. **hold** – No sessions opened during Ramp Up are closed. **cycle + rst** – Once a session is closed, the server will respond with a RST and change to the TCP CLOSED state. This option bypasses the TCP TIME_WAIT state.
rampDist.up	Sets the duration for which new sessions can be opened.	0 – 1,000,000

Parameter	Description	Valid Values
rampDist.upBehavior	Determines how sessions are opened during the ramp up phase.	**full** – The full TCP handshake is performed. **full + data** – The full TCP handshake is performed, and data will be sent once the session opens. **full + data + close** – The full TCP handshake is performed, and data will be sent once the session options. Sessions are closed as they finish sending data and new sessions are opened. **half** – The full TCP handshake is performed, but the final ACK is omitted. **syn** – Only SYN packets are sent. **data flood** – Only PSH data packets are sent. In this mode, the state machine is bypassed, so no connections are set up; therefore, the ACKs will be invalid. Use this mode for testing QoS routing, not stateful DUTs.
rampUpProfile.increment	Sets the number of connections that the connection establishment rate will increment by for the time specified for **rampUpProfile.interval**.	1 – 500,000
rampUpProfile.interval	Sets the time interval that **rampUpProfile.increment** will use to increment the connection establishment rate.	1 – 1,000,000
rampUpProfile.max	Sets the maximum connection establishment rate that will the system will attempt to reach during the ramp up phase. Once the system reaches this rate, it will continue to hold this rate until the ramp up phase ends.	1 – 750,000* (10Gb) or 500,000 (1Gb)

Parameter	Description	Valid Values
rampUpProfile.min	Sets the minimum connection establishment rate that will be used to start the ramp up phase.	1 – 750,000* (10Gb) or 500,000 (1Gb)
rampUpProfile.type	Determines whether the connection establishment rate is a constant rate or an incremental rate – as defined by the user.	**calculated** – The connection establishment rate will be a constant rate. It is calculated by taking the maximum number of sessions divided by the ramp up time. This option allows the component to attempt to reach the maximum number of connections during the ramp up phase. If this option is used, all other parameters listed under the Ramp Up Profile group will be disabled. **stair step** – The connection establishment rate is an incremental rate, and it will be determined by the values input for Minimum Connection Rate, Maximum Connection Rate, Increment N connections per second, and Every N seconds. This option sets the minimum connection rate that the system will start with and increment until it reaches the maximum connection rate or until the ramp up phase ends. If the maximum connection rate is met, the system will hold that rate until the ramp up phase is over.
rateDist.rate	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with **rateDist.increment**.	1 – 30

Parameter	Description	Valid Values
rateDist.type	Sets how data rates are determined.	**constant** – Uses **rateDist.min** as the data rate. **random** – Selects a random value between **rateDist.min** and **rateDist.max** as the data rate. **range** – Starts at **rateDist.min** and increments until it reaches **rateDist.max**. Once the maximum value is met, the data rate will restart at minimum value.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sessions.closeFast	Disables the limit on the close rate. If this option is set to **false**, then the close rate is limited by the connection establishment rate. If this option is set to **true**, then the system will close the sessions as quickly as the bandwidth allows; therefore, the number of open sessions will be closer to the maximum number of simultaneous sessions set for the component.	true or false
sessions.max	Sets the maximum number of concurrent sessions that can be set up by the system at a given time.	1 – 20,000,000 (10Gb) or 5,000,000 (1Gb)
sessions.maxPerSecond	Sets the maximum number of sessions that can occur per second.	1 – 750,000* (10Gb) or 500,000 (1Gb)
sessions.target	The number of sessions that must open to pass the test.	1 – 20,000,000* (10Gb) or 5,000,000 (1Gb)

Parameter	Description	Valid Values
sessions.targetPerSecond	The number of connections per second that must be reached to pass the test.	1 – 750,000* (10Gb) or 500,000 (1Gb)
srcPortDist.max	Sets the maximum source port number, if srcPortDist.type is **range** or **random**.	0 – 65,535
srcPortDist.min	Sets the minimum source port number, srcPortDist.type is **range** or **random**. Otherwise, this will be the value used for the source port.	0 – 65,535
srcPortDist.type	Determines how port numbers are assigned.	**constant** – Uses **srcPortDist.min** as the source port. **random** – Uses random values between **srcPortDist.min** and **srcPortDist.max**. **range** – Increments **srcPortDist.min** by one until it reaches **srcPortDist.max**. Once the port number reaches the maximum source port number, it will reset to the minimum source port number.
tcp.add_timestamps	Allows the size of the TCP header to be expanded by 12 – 32 bytes. Disable this parameter if you are generating TCP stacks with segment sizes greater than 1,460 and do not want to generate jumbo frames.	true or false

Parameter	Description	Valid Values
tcp.aging_time	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_WAIT state before closing.	0 – 120
tcp.delay_acks	Determines whether or not to delay the ACK of TCP segments. Enable this parameter to send the ACK on the next send, or disable this parameter to send the ACK separately on receive.	true or false
tcp.handshake_data	Determines whether to add data to the client ACK packet of the TCP handshake.	true or false
tcp.initial_receive_ window	Sets the size of the initial receive window for a new connection.	1 – 65,535
tcp.mss	Sets the maximum segment size that is used during the ramp up phase.	512 – 9,146
tcp.raw_flags	Allows the specification of the TCP flags as decimal values.	- 1 – 4095
tcp.reset_at_end	Indicates whether or not a test should reset all existing TCP connections at the end.	true or false
tcp.retries	Sets the number of times a connection is attempted before it is canceled.	1 – 3
tcp.retry_quantum_ms	Sets the amount of time that elapses before a connection is retried.	100 – 2,000

Stack Scrambler Parameters

The following table lists the parameters for Stack Scrambler and their valid values. When referencing these options (or parameters) in the Tcl interface, each parameter is preceded with a dash (‘-’).

Stack Scrambler Parameters

Parameter	Description	Valid Values
badChecksum	Sets the percentage of packets that will have a malformed checksum.	0 – 100
badIPOptions	Sets the percentage of IP packets that will have malformed IP options.	0 – 100
badIPVersion	Sets the percentage of IP packets that will have a malformed IP version.	0 – 100
badTCPOptions	Sets the percentage of TCP packets that will have malformed TCP options.	0 – 100
badUrgentPointer	Sets the percentage of packets that will have a malformed urgent pointer.	0 – 100
dstPort	Sets the destination port for all TCP/UPD packets. Setting this parameter to 0 will randomize this value.	0 – 65,535
duration.durationTime	Sets the duration of the test.	hours, minutes, seconds
duration.durationFrames	Sets the length of the test in frames.	1 – 1,000,000,000
fragments	Sets the percentage of packets that will be fragmented.	0 – 100
handshakeTCP	Determines whether the system sends valid handshake packets to establish TCP sessions before fuzzing.	true or false
prng.seed	Sets a value for the seed generator. This value enables the ability to resend the same data to the device.	0 – 4,294,967,295
rateDist.rate	Sets the time increment for increasing or decreasing the data rate. This parameter is used in conjunction with rateDist.increment.	1 – 30

Parameter	Description	Valid Values
rateDist.type	Sets how data rates are determined.	**constant** – Uses rateDist.min as the data rate. **random** – Selects a random value between rateDist.min and rateDist.max as the data rate. **range** – Starts at rateDist.min and increments until it reaches rateDist.max. Once the maximum value is met, the data rate will restart at minimum value.
rateDist.unit	Sets the unit of measurement for the data rate.	mbps or fps
sizeDist.max	Sets the maximum frame/packet size; this parameter is used only if sizeDist.type is set to range.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)
sizeDist.min	Sets the minimum frame/packet size, if sizeDist.type is set to constant. Otherwise, this is the minimum value used if sizeDist.type is set to range or random.	64 – 9216 bytes (frames) 46 – 1500 bytes (packets)
sizeDist.unit	Sets whether Bit Blaster uses frame or packets.	packet or frame
srcPort	Sets the source port for all TCP/UDP packets. Setting this parameter to 0 will randomize this value.	0 – 65,535
targetStack	Sets the protocol stack to target.	All, IP, TCP, or UDP

Running Tests

Use the `run` command to run your script. If you use the `run` command without any arguments, the `run` command will run the test and block control over the script until the test completes. Once the test is done, you can resume control over the Tcl interface.

Note: If you run a test without the `-group` attribute, the test will default to group 1.

If you do not want control to be blocked, you can use the `-async` attribute, which will return control to you once a test starts. The test will run in the background, but once it completes, it will run the string 'asyncCommand' as a Tcl script. If you run a test using the `-async` attribute, you can use the `wait` command to block control until the test completes.

With the `run` command, you can use the `-progress` attribute to specify a Tcl script that will be called periodically while the test is running. This will allow you to monitor the progress of the test. The `-progress` attribute will use the `concat` command to append two attributes to the script you provide: the test's name and a percentage of completion.

If you do not want to provide a script, you can use the `bps::textprogress` command to show a text-based progress bar. You will need to specify the channel to which the command should output the text (e.g., `stdout`).

When a test completes, it will return a list of the test criteria that failed, as well as a list of pairs (i.e., the failure description and the criteria name).

With the `run` command, you can use the `-rtstats` attribute to specify a callback to update your charts with Real-Time statistics. The value that you assign to this argument is interpreted as a command that will run when new statistics become available.

The following example defines a procedure that prints out the statistics and then runs a test set up to call that procedure.

Example

```
proc print_rtstats {testid
statvals} {
dict for {var val} $statvals {
puts "$var: $val"
}
}
$t run -rtstats print_rtstats
```

Note: If the test encounters an error when you attempt to run it, you will see a Tcl exception.

Syntax

Use one of the following syntaxes to run a test.

```
$testObject run; #the simple way to run a
test
```

```
$testObject run -progress "bps::textprogress stdout" ; #runs and outputs the
test progress
```

```
$testObject run -async {puts "Test Completed" } -progress "bps::textprogress
stdout" ; #runs the test in the background
```

The following table breaks down the elements for running tests.

Running Tests

Element	Description
run	Runs the test
-allowMalware	Confirm that malware should be allowed in this test
-async value	Specified as an attributed to the run command. Runs the test in the background and executes the command specified.
-flowexceptions value	Identifies the script to run with flow exception notifications
-group value	Identifies the interface group to be used in the test
-help	Prints the list of commands with descriptions
-progress value	Allows you to monitor the progress of the test
-rtstats value	Calls the -rtstats attribute when there are new Real-Time statistics available. This attribute allows you to capture Real-Time statistics at any time during the progress of your test.
-?	Prints the list of commands with descriptions

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ];
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

```
$test1 run -progress "bps::textprogress stdout" ; #runs the test
```

Available Real-Time Statistics

lists the available Real-Time statistics that appear in the test results. Which statistics actually show up in the test results will depend on which components are included in the test.

Available Real-Time Statistics

Statistic	Description
udpTxFrames	UDP Frames transmitted
udpTxFrameRate	UDP frame transmit rate
udpRxFrames	UDP Frames received
udpRxFrameRate	UDP frame receive rate
tcpFlowsConcurrent	Concurrent TCP Flows
udpFlowsConcurrent	Concurrent UDP Flows
totalFlowsConcurrent	Total Concurrent Flows
superFlowsConcurrent	Concurrent Super Flows
superFlowRate	Super Flow rate
sctpFlowsConcurrent	Concurrent SCTP Flows
sctpRxFrameDataRate	SCTP data receive rate
sctpAvgSetupTime	Average SCTP setup time
sctpAvgResponseTime	Average SCTP response
sctpAvgCloseTime	Average SCTP time to close
sctpAvgSessionDuration	Average duration
sctpClientEstablished	Client established
sctpClientEstablishRate	Client establish rate
sctpClientClosed	Client closed
sctpClientClosedByAbort	Client closed by abort
sctpClientClosedByAbortRate	Client close by abort rate

Statistic	Description
sctpClosedNormally	Aggregate closed normally
sctpClosedNormallyRate	Aggregate normal close rate
sctpClosedByAbort	Aggregate closed by abort
sctpClosedByAbortRate	Aggregate close by abort rate
sctpClientReceivedShutdown	Client received shutdown
sctpServerReceivedShutdown	Server received shutdown
sctpClientReceivedAbort	Client received abort
sctpServerReceivedAbort	Server received abort
sctpClientCloseRate	Client close rate
sctpAttempted	Client attempted
sctpAttemptRate	Client attempt rate
sctpServerEstablished	Server established
sctpServerEstablishRate	Server establish rate
sctpServerClosed	Server closed
sctpServerClosedByAbort	Server closed by abort
sctpServerClosedByAbortRate	Server close by abort rate
sctpServerCloseRate	Server close rate
sctpSharedRxQueueFull	Shared Connection Receive Queue Full
sctpSharedRxFlowNotFound	Shared Connection Flow Not Found for Received Packet
sctpSharedTxNotClient	Shared Connection Transmit Not a Client
avgLatency	Average frame latency
avgLatencyInst	Average instantaneous frame latency
concurrentAppFlows	Concurrent application flows
appAttempted	Application attempted

Statistic	Description
appSuccessful	Application successes
appUnsuccessful	Application failures due to external events
appUnsuccessfulTcpRetries	Application failures due to TCP retry limit
appUnsuccessfulUdpRetries	Application failures due to UDP receive timeout
appUnsuccessfulResolveRetries	Application failures due to resolve timeout
appUnsuccessfulRampDown	Application failures due to ramp down
appUnsuccessfulExpectedData	Application failures due to a premature session close
appUnsuccessfulSuperflowClosed	Application failures due to a premature Super Flow close
appUnsuccessfulAppFailure	Generic application failures
appAttemptedRate	Application attempt rate
appSuccessfulRate	Application success rate
appUnsuccessfulRate	Application failure rate
appAttemptedMatches	Application attempted matches
appSuccessfulMatches	Application successful matches
appFailedMatches	Application failed matches
appChunkStartCount	Conditional Request chunk starts
appChunkEndCount	Conditional Request chunk ends
appServerDataValid	Server Response data valid count
appServerDataNotValid	Server Response data not valid count
ethTxFrames	Ethernet frames transmitted
ethTotalErrors	Total Errors
ethTxFrameRate	Ethernet frame transmit rate
ethTxFrameDataRate	Ethernet transmit rate
ethRxFrames	Ethernet frames received

Statistic	Description
ethRxFrameRate	Ethernet frame receive rate
ethRxFrameDataRate	Ethernet receive rate
tcpRxFrameDataRate	TCP data receive rate
tcpAvgSetupTime	Average TCP setup time
tcpAvgResponseTime	Average TCP response
tcpAvgCloseTime	Average TCP time to close
tcpAvgSessionDuration	Average duration
tcpClientEstablished	Client established
tcpClientEstablishRate	Client establish rate
tcpClientClosed	Client closed normally
tcpClientClosedByReset	Client closed by sending RST
tcpClientClosedByResetRate	Client close by sending RST rate
tcpClosedNormally	Aggregate closed normally
tcpClosedNormallyRate	Aggregate normal close rate
tcpClosedByReset	Aggregate closed by sending RST
tcpClosedByResetRate	Aggregate close by sending RST rate
tcpClientReceivedFin	Client received FIN
tcpServerReceivedFin	Server received FIN
tcpClientReceivedRst	Client received RST
tcpServerReceivedRst	Server received RST
tcpUnknownReceivedRst	Unknown/Closed flow received RST
tcpCorruptOptions	Corrupt TCP Options
tcpInvalidLength	Invalid TCP Header Length
tcpInvalidFlags	Invalid TCP Flag Combination

Statistic	Description
tcpClientCloseRate	Client close rate
tcpAttempted	Client attempted
tcpAttemptRate	Client attempt rate
tcpServerEstablished	Server established
tcpServerEstablishRate	Server establish rate
tcpServerClosed	Server closed normally
tcpServerClosedByReset	Server closed by sending RST
tcpServerClosedByResetRate	Server close by sending RST rate
tcpServerCloseRate	Server close rate
totalAllowed	Total Strikes allowed
totalBlocked	Total Strikes blocked
totalErrored	Total Strikes errored
totalSkipped	Total Strikes skipped
totalStrikes	Total Strikes
avgLatency	Average frame latency
avgLatencyInst	Average instantaneous frame latency
flowExceptionCount	Flow Exception Count
udpTxFrames	UDP Frames transmitted
udpTxFrameRate	UDP frame transmit rate
udpRxFrames	UDP Frames received
udpRxFrameRate	UDP frame receive rate
tcpRxFrameDataRate	TCP data receive rate
tcpAvgSetupTime	Average TCP setup time
tcpAvgResponseTime	Average TCP response

Statistic	Description
tcpAvgCloseTime	Average TCP time to close
tcpAvgSessionDuration	Average duration
tcpClientEstablished	Client established
tcpClientEstablishRate	Client establish rate
tcpClientClosed	Client closed normally
tcpClientClosedByReset	Client closed by sending RST
tcpClientClosedByResetRate	Client close by sending RST rate
tcpClosedNormally	Aggregate closed normally
tcpClosedNormallyRate	Aggregate normal close rate
tcpClosedByReset	Aggregate closed by sending RST
tcpClosedByResetRate	Aggregate close by sending RST rate
tcpClientReceivedFin	Client received FIN
tcpServerReceivedFin	Server received FIN
tcpClientReceivedRst	Client received RST
tcpServerReceivedRst	Server received RST
tcpUnknownReceivedRst	Unknown/Closed flow received RST
tcpCorruptOptions	Corrupt TCP Options
tcpInvalidLength	Invalid TCP Header Length
tcpInvalidFlags	Invalid TCP Flag Combination
tcpClientCloseRate	Client close rate
tcpAttempted	Client attempted
tcpAttemptRate	Client attempt rate
tcpServerEstablished	Server established
tcpServerEstablishRate	Server establish rate

Statistic	Description
tcpServerClosed	Server closed normally
tcpServerClosedByReset	Server closed by sending RST
tcpServerClosedByResetRate	Server close by sending RST rate
tcpServerCloseRate	Server close rate
sctpRxFrameDataRate	SCTP data receive rate
sctpAvgSetupTime	Average SCTP setup time
sctpAvgResponseTime	Average SCTP response
sctpAvgCloseTime	Average SCTP time to close
sctpAvgSessionDuration	Average duration
sctpClientEstablished	Client established
sctpClientEstablishRate	Client establish rate
sctpClientClosed	Client closed
sctpClientClosedByAbort	Client closed by abort
sctpClientClosedByAbortRate	Client close by abort rate
sctpClosedNormally	Aggregate closed normally
sctpClosedNormallyRate	Aggregate normal close rate
sctpClosedByAbort	Aggregate closed by abort
sctpClosedByAbortRate	Aggregate close by abort rate
sctpClientReceivedShutdown	Client received shutdown
sctpServerReceivedShutdown	Server received shutdown
sctpClientReceivedAbort	Client received abort
sctpServerReceivedAbort	Server received abort
sctpClientCloseRate	Client close rate
sctpAttempted	Client attempted

Statistic	Description
sctpAttemptRate	Client attempt rate
sctpServerEstablished	Server established
sctpServerEstablishRate	Server establish rate
sctpServerClosed	Server closed
sctpServerClosedByAbort	Server closed by abort
sctpServerClosedByAbortRate	Server close by abort rate
sctpServerCloseRate	Server close rate
sctpSharedRxQueueFull	Shared Connection Receive Queue Full
sctpSharedRxFlowNotFound	Shared Connection Flow Not Found for Received Packet
sctpSharedTxNotClient	Shared Connection Transmit Not a Client
pingsSent	Pings Sent
pingsReceived	Pings Received
pingsTimeout	Pings Timed Out

Starting the Packet Trace

Use the `startPacketTrace` command to start capturing packets with the packet buffer.

Syntax

Use the following syntax to start the packet trace.

```
$testObject
startPacketTrace
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

```
$test1 run -async RunDoneProc when done after 2000; #wait 2 seconds
```

```
$test1 startPacketTrace; #starts collecting packets after 2 seconds
```

Stopping the Packet Trace

Use the `stopPacketTrace` command to stop the packet buffer from capturing packets during a test.

Syntax

Use the following syntax to stop the packet trace.

```
$testObject
stopPacketTrace
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

```
$test1 run -async RunDoneProc when done after 2000; #wait 2 seconds
```

```
$test1 stopPacketTrace; #stops collecting packets after 2 seconds
```

Listing the Components in a Test

You can use `getComponents` to return a list of all the components used by the test.

Syntax

Use the following syntax to get a list of components used by the test.

```
$testObject
getComponents
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 getComponents; #returns a list of components used by the test
```

Saving the Test

Use the `save` command to save the test.

Syntax

Use the following syntax to save the test.

```
$testObject
save
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

Canceling the Test Run

Use the `cancel` command to cancel the test. You can use the `-force` attribute to force any existing test with the same name into an idle state.

Syntax

Use the following syntax to cancel the test.

```
$testObject
cancel
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set c1 [$var getChassis]; #creates the chassis object
```

```
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
```

```
set test1 [$var createTest -template "appsim_enterprise" -name "myTest" ;
#creates the test object and a test called MyTest based on the AppSim test
```

```
$test1 configure -neighborhood Neighborhood1; #sets the Network Neighborhood
for the test to be Neighborhood 1
```

```
$test1 configure -dut Profile1; #sets the DUT Profile to Profile1
```

```
$test1 configure -description "this test is based on the default application
simulator quick test" ; #sets the description for the test
```

```
$test1 save; #saves the test
```

```
$test1 run; #runs the test
```

```
$test1 cancel -async ProcToPerformWhenRunIsDone; #cancels the test
```

Canceling a Running Test

Use the `cancelTest` command to cancel a test via the chassis object.

Syntax

```
$testObject
cancelTest
```

Example

```
set c [$bps getChassis]; #creates the chassis object
```

```
$c getTests; #returns a list of systems and the tests running on them
```

```
$c cancelTest test@192@127.0.0.1@AppSim@14; #cancels the specified test on the
chassis object
```

Exporting Test Results

Use the `exportReport` command to export test results in CSV, HTML, PDF, RTF, XLS, XML, and ZIP (CSV files). Identify which format you would like to export in by specifying the `-format` parameter along with the format value.

Note:

- Test reports exported using the .csv format will be downloaded as a single CSV file containing multiple tables. To export reports as a .zip file containing a separate file for each table, you must use the .zip format.
- Test reports exported using the html format will be downloaded in .zip format. To view the html report, you must first unzip the exported file.

Syntax

Use the following syntax to export a hard copy of test results.

```
$testObject exportReport-file
/../reportName.pdf
```

See for available options for the exportReport command.

Available Options for exportReport

Option	Description
-file value	output file <>
-channel value	output channel <>
-format value	Report format {csv, flowstats, html, pdf, rtf, xls, bpt, xml, zip} <pdf>
-sectionids	$filtername Generate a report using a list variable to filter desired sections **OR** {listSectionHierarchy} Select the sections you want to show up in the report in a hierarchy list Note: Ensure to add the entire list of sections hierarchy to the sections you want. Sections without the right path will not be found or appear in the report.
-help	Prints the list of commands with descriptions
-?	Prints the list of commands with descriptions

Example 1

```
$testObject exportReport -file
/temp/bitblasterresults.pdf
```

Example 2

```
set bbresult [open "|unzip" w]
$testobject exportReport -channel $bbresult -format csv; #opens a pipe to the
'unzip' system command and exports the report there directly.
```

Example 3

```
% $t exportReport -file /tmp/foo.zip -format flowstats; #exports the test
report in flowstats format.
```

Example 4

```
% $t exportReport -file /tmp/foo.xml -format xml; #exports the test report in
xml format.
```

Example 5

```
$testObject exportReport -testid [$t resultId] -file /tmp/foo.xml -format
xml
```

Searching Test Reports

Use the `listTestResults` command to return test report results. Use a prefix in the search string to search for anything the option list supports. The search dynamically filters the list per keystroke.

Syntax

Use the following syntax to search the test report for results.

```
$connectionObject listTestResults
value
```

See for available options for the `listTestResults` command. These options help to restrict the search results.

Available Options for listTestResults

Option	Description
`-class value`	Identifies the type of test results to list. Accepted values include single, resiliency, series, or multi.
`-dut value`	Returns tests results using this DUT only.
`-host value`	Returns test results from this host only.
`-internalid value`	Returns test results having this internal id only.
`-iteration value`	Returns test results having this iteration number only.

Option	Description
-limit value	Limits the results to this value.
-name value	Returns test results from this test only.
-network value	Returns tests results using this network only.
-offset value	Offset into results.
-result value	Returns tests that completed with this result only.
-sort value	Sorts column based on this value
-sortorder value	Sorts column order (ascending / descending).
-userid value	Returns test results from tests run by specified user only.

Example

```
$bps listTestResults appsim test; #Searches for any test result that mentions "appsim" and "test" somewhere in its metadata
```

```
$bps listTestResults -userid admin -iteration 2; #Lists test results that are from iteration 2 of any test run by user admin
```

```
$bps listTestResults userid:admin iteration:2; # An equivalent query using the search string rather than command options
```

Viewing Aggregate Statistics

You can call the aggregate statistics object to list all of the statistics available. For example, `$aggStatsObject` will return a list of all the statistics that are available for the aggregate statistics object.

Syntax

Use the following syntax to view the results from the aggregate statistics object.

```
$resultObjectName values aggStats
```

The following table lists descriptions for the available aggregate statistics.

Aggregate Statistics

Statistic	Description
cpu_usage	CPU Usage
ethAlignmentErrors	Ethernet alignment errors
ethDropEvents	Ethernet drop events
ethFCSErrors	Ethernet FCS errors
ethOversizedFrames	Ethernet oversize frames
ethRxErrors	Ethernet receive errors
ethRxFrameData	Ethernet bytes received. This includes L7 and all packet overhead, including L2, L3, L4 headers, ethernet CRC, and inter-packet gap (20 bytes per frame).
ethRxFrameDataRate	Ethernet receive rate. This includes L7 and all packet overhead, including L2, L3, L4 headers, ethernet CRC, and inter-packet gap (20 bytes per frame).
ethRxFrameRate	Ethernet frame receive rate
ethRxFrames	Ethernet frames received
ethRxPauseFrames	Ethernet pause frames received
ethTotalErrors	Total Errors
ethTxErrors	Ethernet transmit errors
ethTxFrameData	Ethernet bytes transmit. This includes L7 and all packet overhead, including L2, L3, L4 headers, ethernet CRC, and inter-packet gap (20 bytes per frame).
ethTxFrameDataRate	Ethernet transmit rate. This includes L7 and all packet overhead, including L2, L3, L4 headers, ethernet CRC, and inter-packet gap (20 bytes per frame).
ethTxFrameRate	Ethernet frame transmit rate
ethTxFrames	Ethernet frames transmitted
ethTxPauseFrames	Ethernet pause frames transmitted
ethUndersizedFrames	Ethernet undersize frames

Statistic	Description
linux mem_free_kb	Free memory on the System Controller
mem_total_kb	Total memory on the System Controller
mem_used_kb	Used memory
mount percent_used	The percent of disk spaced used on the disk partition
superFlowRate	Super Flow rate
superFlows	Aggregate Super Flows
superFlowsConcurrent	Concurrent Super Flows
tcpFlowRate	TCP Flow rate
tcpFlows	Aggregate TCP Flows
tcpFlowsConcurrent	Concurrent TCP Flows
timestamp	The time that the datapoint was taken (refers to the rest of the data that comes with it)
udpFlowRate	UDP Flow rate
udpFlows	Aggregate UDP Flows
udpFlowsConcurrent	Concurrent UDP Flows

Example

The following example displays the aggregate statics for the test.

```
% set var [bps::connect 10.10.10.10 joe passwd]
% set c1 [$var getChassis]; # creates the chassis object
% $c1 reservePort 1 0; #reserves ports 0 on slot 1
% $c1 reservePort 1 1; #reserves ports 1 on slot 1
% set t [$var createTest -template AppSim]; # creates a test object based on the AppSim test
% set c [$t get aggStats]; # stores the aggregate statistics for a test in an object
% $t run; # runs the test
```

```
% set r [$c result]; # creates the results object
```

```
% $r values aggStats; # retrieves the avaliable aggregate statistics of the
test from the results object
```

Listing Multi-box Tests

Use the `listMultiboxTests` command to display a list of multi-box tests currently on the system. This includes all user-created and BreakingPoint supplied tests.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display multi-box tests created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint multi-box tests, or `custom`, which will return a list of all user-created multi-box tests. Use the `-timeunit` and `-timeinterval` attributes to list multi-box tests by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

Syntax

Use the following syntax to list all multi-box tests on the system.

```
$connectionObject
listMultiboxTests
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listMultiboxTests; #returns a list of multi-box tests on the system
```

Creating a Multi-box Test

A multi-box test enables you concurrently run tests on up to five BreakingPoint systems. The system you create the multi-box test on will be the main system; this system will be used as the management interface for secondary systems. You must know the IP addresses and authentication information for the secondary systems in order to create a multi-box test. Additionally, all systems must be running the same build.

By default, when you create a multi-box test, the system will automatically use the following configuration:

- Test – AppSim
- DUT Profile – BreakingPoint Default
- Network Neighborhood – BreakingPoint Switching

For more information on multi-box testing, see the Multi-box Testing Overview on page 938section.

The connection object has a command called `createMultiboxTest` that you can use to create the multi-box test. Additionally, you can use the following attributes to create the multi-box test: `-name` and `-template`. The `-name` attribute enables you to name the multi-box test, and the `-template` attribute enables you to specify an existing multi-box test on which to base the multi-box test. If you do not specify a template, the system will create an empty multi-box test.

The recommended way to create is a multi-box test is create an object for it. The syntax and example below utilize this method.

Note: If you need to see a list of the multi-box tests that currently exist on the system, use the `listMultiboxTests` command (e.g. $connectionObject listMultiboxTests). This will return all multi-box tests that are currently on the system.

Syntax

Use the following syntax to create a multi-box test.

```
set multiboxObject [$connectionObject createMultiboxTest -template
"multiboxTest" -name "multiBoxTest name" ]
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

Configuring the Multi-box Test

The `configureTest` command enables you to modify the authentication information or the test used for the system. There are three attributes you can use with the `configureTest` command: `-name` to provide a new login ID, `-password` to provide a new password, and `-test` to assign a different test for the system to run.

Syntax

Use the following syntax to reconfigure a multi-box test.

Note: If you are configuring the main system, use localhost in place of an IP address (e.g. $mt1 configureTest localhost -username joe).

```
$multiboxTestObject configureTest IP -username "name" -password "password" -
test "test"
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "BitBlaster" "BreakingPoint Default" "BreakingPoint Switching" ; adds 10.10.10.11 to the multi-box test
```

```
$mt1 configureTest 10.10.10.11 -name admin -password admin; #changes the authentication information for the system
```

Adding Secondary Systems to the Multi-box Test

Each multi-box test can have up to 5 systems: one master system and four secondary systems. Each secondary system can run one test during a multi-box test and can have a different Network Neighborhood and DUT Profile assigned for it.

Note: You can only use the `addTest` command to add secondary systems. To modify the primary system's test or authentication information, use the `configureTest` command.

To add secondary systems to the multi-box test, use the syntax and follow the example provided below.

Syntax

Use the following syntax to add tests to a multi-box test.

```
$multiboxTestObject addTest systemIP userID password "test" "DUT Profile" "Network Neighborhood"
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default" "BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system will run the App Sim test and use the default BreakingPoint DUT Profile and BreakingPoint Routing Network Neighborhood
```

```
$mt1 addTest 10.10.10.12 john pswd "SessionSender" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.12 to the multi-box test, this system
will run the Session Sender test and use the default BreakingPoint DUT Profile
and BreakingPoint Routing Network Neighborhood
```

Listing the Tests in a Multi-box Test

To view all the systems and the tests in a multi-box test, use the `getTests` command.

Syntax

Use the following syntax to display a list of systems and the tests running on them.

```
$multiboxTestObject
getTests
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

```
$mt1 getTests; #returns a list of systems and the tests running on them
```

Removing Tests from the Multi-box Test

To remove a system and its test from the multi-box test, use the `removeTest` command.

Note: You cannot remove the primary system (localhost) from the multi-box test.

Syntax

Use the following syntax to remove secondary systems from the multi-box test.

```
$multiboxTestObject removeTest
IP
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

```
$mt1 removeTest 10.10.10.11; #removes this system from the multi-box test
```

Viewing the Multibox Configuration

Use the `getTests` command to return a list of systems and their tests and configurations.

Syntax

Use the following syntax to view the secondary systems and tests associated with the multi-box test.

```
$multiboxTestObject
getTests
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

```
$mt1 addTest 10.10.10.12 john pswd "SessionSender" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.12 to the multi-box test, this system
will run the Session Sender test and use the default BreakingPoint DUT Profile
and BreakingPoint Routing Network Neighborhood
```

Reserving Ports for Secondary Systems in a Multi-box Test

In order to run a multi-box test, you must reserve ports on each system. To do this, you will need to create chassis object for each secondary system. These procedures are the same as if you would create a chassis object for the primary system.

Syntax

Use the following syntax to create a chassis object for the secondary system and to reserve ports.

```
set chassisObject [$connectionObject getChassis
IP]
$chassisObject reserverPort slot# port#
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
set c1 [$var getChassis]; #creates chassis object for the primary system
```

```
$c1 reserve 1 0; #reserves slot 1/port 0 on the primary system
```

```
$c1 reserve 1 1; #reserves slot 1/port 1 on the primary system
```

```
set c2 [$var getChassis 10.10.10.11]; #creates chassis object for the secondary
system
```

```
$c2 reserve 1 0; #reserves slot 1/port 0 on the secondary system
```

```
$c2 reserve 1 1; #reserves slot 1/port 1 on the secondary system
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

Running a Multi-box Test

Use the `run` command to run the test.

Syntax

Use the following syntax to run a multibox test.

```
$multiboxObject
run
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
set c1 [$var getChassis]; #creates chassis object for the primary system
$c1 reserve 1 0; #reserves slot 1/port 0 on the primary system
$c1 reserve 1 1; #reserves slot 1/port 1 on the primary system
set c2 [$var getChassis 10.10.10.11]; #creates chassis object for the secondary system
$c2 reserve 1 0; #reserves slot 1/port 0 on the secondary system
$c2 reserve 1 1; #reserves slot 1/port 1 on the secondary system
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default" "BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system will run the App Sim test and use the default BreakingPoint DUT Profile and BreakingPoint Routing Network Neighborhood
$mt1 run; #runs the test
```

Canceling a Multi-box Test Run

To cancel a running multi-box test, use the `cancel` command. You can use the `-force` attribute to force any existing multi-box test with the same name into an idle state.

Syntax

Use the following syntax to cancel a running multibox test.

```
$multiboxObject
cancel
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the system
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox test called MTest1
set c1 [$var getChassis]; #creates chassis object for the primary system
$c1 reserve 1 0; #reserves slot 1/port 0 on the primary system
$c1 reserve 1 1; #reserves slot 1/port 1 on the primary system
set c2 [$var getChassis 10.10.10.11]; #creates chassis object for the secondary system
```

```
$c2 reserve 1 0; #reserves slot 1/port 0 on the secondary system
```

```
$c2 reserve 1 1; #reserves slot 1/port 1 on the secondary system
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

```
$mt1 run; #runs the test
```

```
$mt1 cancel; #cancels the test
```

Saving the Multi-box Test

To save the multi-box test, use the `save` command. This will store the multi-box test for later use. You can use the `-force` attribute to overwrite any existing multi-box test with the same name.

Syntax

Use the following syntax to save the multi-box test.

```
$multiboxTestObject save -
force
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set mt1 [$var createMultiboxTest -name "MTest1" ; #creates an empty multibox
test called MTest1
```

```
$mt1 addTest 10.10.10.11 john pswd "AppSim" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.11 to the multi-box test, this system
will run the App Sim test and use the default BreakingPoint DUT Profile and
BreakingPoint Routing Network Neighborhood
```

```
$mt1 addTest 10.10.10.12 john pswd "SessionSender" "BreakingPoint Default"
"BreakingPoint Routing" ; #adds 10.10.10.12 to the multi-box test, this system
will run the Session Sender test and use the default BreakingPoint DUT Profile
and BreakingPoint Routing Network Neighborhood
```

```
$mt1 save -force; #saves the test
```

Listing Test Series

Use the `listTestSeries` command to display a list of test series currently on the system. This includes all user-created and BreakingPoint supplied test series.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display test series created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint created test series, or `custom`, which will return a list of all user-created test series. Use the `-timeunit` and `-timeinterval` attributes to list test series by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

Syntax

Use the following syntax to list all test series on the system.

```
$connectionObject
listTestSeries
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listTestSeries; #returns a list of test series on the system
```

Creating a Test Series

A test series enables you to sequentially run a set of up to 20 tests. Since each test has its own Network Neighborhood and DUT Profile, you will not need to assign either of these for the test series. However, you will need to ensure that you have the necessary ports reserved to run the tests in the test series. For more information on creating tests, see the Creating Tests on page 1018 section. For more information on port reservations, see theReserving Ports on page 993 section.

By default, when you create a multi-box test, the system will automatically use the following configuration:

- Test – AppSim
- DUT Profile – BreakingPoint Default
- Network Neighborhood – BreakingPoint Switching

For more information on multi-box testing, see the Multi-box Testing Overview on page 938section.

The connection object has a command called `createTestSeries` that you can use to create the test series. Additionally, you can use the following attributes to create the test series: `-name` and `-template`. The `-name` attribute enables you to name the test series, and the `-template` attribute enables you to specify an existing test series on which to base the test. If you do not specify a template, the system will create an empty test series.

The recommended way to create is a test series is create an object for it. The syntax and example below utilize this method.

Syntax

Use the following syntax to create a test series.

```
set testSeriesObject [$connectionObject createTestSeries -template "testSeries"
-name "testSeries name" ]
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set ts1 [$var createTestSeries -template "appTestSeries1" -name
"appTestSeries2" ]; #creates a test series based on an existing template
```

Listing Existing Test Series on the System

To get a list of the tests that are on the system use the `listTestSeries` command. This will help you determine if there is a test series that already exists on which you would like to base your test series or to see what is already currently available on the system.

Syntax

Use the following syntax to create a test series.

```
$connectionObject
listTestSeries
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
$var listTestSeries; #returns a list of all test series on the system
```

Adding Tests to a Test Series

Use the `addTest` command to add a test to a test series. Each test series can have up to 20 tests. If you need to modify a test's configuration – such as it's Network Neighborhood, DUT Profile, or parameters – you will need to modify the individual test. For more information on modifying tests, see theConfiguring Test Components on page 1081section.

If you need to see a list of the tests that can be added to the test series, use the `listTests` command (e.g. $connectionObject listTests). This will return all tests that are currently on the system.

To add secondary systems to the test series, use the syntax and follow the example provided below.

Syntax

Use the following syntax to add tests to a test series.

```
$testSeriesObject addTest "test"
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the system
set c1 [$var getChassis]; #creates chassis object
$c1 reserve 1 0; #reserves slot 1/port 0
$c1 reserve 1 1; #reserves slot 1/port 1
set SS1 [$bps createTestSeries -name "Security Tests" ]; #creates an object called SS1 and a test series called Security Tests
$SS1 addTest "Security T1" ; #adds a test to the test series
```

Removing Tests from a Test Series

To remove a test from a test series, use the `removeTest` command.

Syntax

Use the following syntax to remove tests from a test series.

```
$testSeriesComponent removeTest
<indexNumber>
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the system
set c1 [$var getChassis]; #creates chassis object
$c1 reserve 1 0; #reserves slot 1/port 0
$c1 reserve 1 1; #reserves slot 1/port 1
set SS1 [$bps createTestSeries -name "Security Tests" ]; #creates an object called SS1 and a test series called Security Tests
$SS1 addTest "Security T1" ; #adds a test to the test series
$SS! removeTest 2; #removes the second test index from the test series. The numbering here is zero based, so the test with the second test index would be the first test.
$SS1 save; #saves the test series
```

Listing the Tests in a Test Series

To view all the tests in a test series, use the `getTests` command.

Syntax

Use the following syntax to display a list of tests in a test series.

```
$testSeriesObject
getTests
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set ts1 [$var createTestSeries -name "TSeries1" ; #creates an empty test
series
```

```
$ts1 addTest "AppSim" ; #adds a test to the test series
```

```
$ts1 getTests; #returns a list of tests in the test series
```

Running a Test Series

Use the `run` command to run a test series.

Syntax

Use the following syntax to run a test series.

```
$testSeriesObject
run
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set c1 [$var getChassis]; #creates chassis object
```

```
$c1 reserve 1 0; #reserves slot 1/port 0
```

```
$c1 reserve 1 1; #reserves slot 1/port 1
```

```
set SS1 [$bps createTestSeries -name "Security Tests" ]; #creates an object
called SS1 and a test series called Security Tests
```

```
$SS1 addTest "Security T1" ; #adds a test to the test series
```

```
$SS1 save; #saves the test series
$SS1 run; #runs the test series
```

Canceling a Test Series Run

To cancel a running test series, use the `cancel` command. You can use the `-force` attribute to force any existing test series with the same name into an idle state.

Syntax

Use the following syntax to cancel a running test series.

```
$testSeriesObject
cancel
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
set c1 [$var getChassis]; #creates chassis object
$c1 reserve 1 0; #reserves slot 1/port 0
$c1 reserve 1 1; #reserves slot 1/port 1
set SS1 [$bps createTestSeries -name "Security Tests" ]; #creates an object
called SS1 and a test series called Security Tests
$SS1 addTest "Security T1" ; #adds a test to the test series
$SS1 save; #saves the test series
$SS1 run -async {puts "Series Run Completed" }; #runs the test series in the
background
$SS1 cancel; #cancels the test series
```

Saving the Test Series

To save the test series, use the `save` command. This will store the test series for later use. You can use the `-force` attribute to overwrite any existing test series with the same name.

Syntax

Use the following syntax save the test series.

```
$testSeriesObject save -
force
```

Example

```
set var [bps::connect 10.10.10.10 john pswd]; #creates a connection to the
system
```

```
set c1 [$var getChassis]; #creates chassis object
```

```
$c1 reserve 1 0; #reserves slot 1/port 0
```

```
$c1 reserve 1 1; #reserves slot 1/port 1
```

```
set SS1 [$bps createTestSeries -name "Security Tests" ]; #creates an object
called SS1 and a test series called Security Tests
```

```
$SS1 addTest "Security T1" ; #adds a test to the test series
```

```
$SS1 save; #saves the test series
```

Creating a RFC 2544 Test

You will need to create an object for the RFC 2544 test. Within the object, you will use the `$bps createRFC2544Test` command to create the test.

There are a few things you must keep in mind before creating an RFC 2544 test:

- The MTU defined for the transmitting and receiving ports on the BreakingPoint system must be able to support the frame sizes defined for the RFC 2544 test. You should always check the MTU settings for each port before running the test to ensure that the port supports the frame sizes defined in the test.
- The system will send slow start packets in the reverse direction to the device under test for each iteration. This enables the DUT to determine the ports of the MAC addresses that the BreakingPoint system is using; therefore, in the Traffic Overview section of the RFC 2544 test report, you will see slow start packets listed for each data rate that was tested.
- The RFC 2544 test utilizes logical interface 1 as the transmitting interface and logical interface 2 as the receiving interface.

For more information on the Quick Test - RFC 2544, see Quick Test - RFC 2544 General Information on page 885.

Syntax

Use the `$bpscreateRFC2544Test` command to create the RFC 2544 test, as shown in the following example. Additionally, the example below creates an object for the RFC 2544 test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter.

```
set rfcObjectName [$connectionObjectName
createRFC2544Test]
```

You can also create an RFC 2544 test using a template.

Example

```
% set t [$bps createTest -template {RFC2544}]; #creates an RFC 2544 test based
on an existing RFC2544 template
```

RFC 2544 Test Commands

Once you have created an object for the RFC 2544 test, you can call the object to list all the commands that are available for the RFC 2544 test (e.g., `$rfcobjectName`).

Most of the commands are common to other features in the BreakingPoint System, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

Setting Parameters in an RFC 2544 Test

In order to list the parameters that you can configure for the RFC 2544 test, use the following syntax:

```
set rfcObjectName [$connectionObjectName
createRFC2544Test]

$rfcObjectName configure -option
```

The `configure` command will list all parameters available for the RFC 2544 test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

RFC 2544 Parameters

Parameter	Description	Valid Values
acceptableCorruptFrames	Defines the percentage of corrupt frames that is acceptable for the iteration to pass.	0 – 100
acceptableFrameLoss	Defines the percentage of frame loss that is acceptable for the iteration to pass.	0 – 100
binaryRateLower	The Rate Lower Limit	0 – 100
binaryRateUpper	The Rate Higher Limit	0 – 100

Parameter	Description	Valid Values
binaryResolution	The binary resolution	0 – 100
customPayload	This field is defined only if you have set the Payload to be **user defined**.	You can use standard hexadecimal notation to define a custom payload.
customSteps	Defines the frame sizes that will be tested.	Enter the frame sizes
dut	Defines the DUT Profile that will be used for the RFC 2544 test.	Any DUT Profile currently stored on the system.
frameSizeEnd	Defines the last frame size that will be tested in a step series.	64 – 9216
frameSizeInterval	Defines the interval at which the frame sizes are incremented; the frame size will start at `frameSizeStart`, and increment using `frameSizeInterval`, until it reaches `frameSizeEnd`.	1–128
frameSizeStart	Defines the first frame size that will be tested in a step series.	64 – 9216
loadApplication	Defines the maximum amount of throughput that will be tested.	0 – 10000 or total Total will use the maximum possible throughput
loadUnits	Defines the unit of measurement for the throughput.	mbps or gbps
mode	Defines the mode the test will use to search for the fastest frame rate.	binary, step, or combined

Parameter	Description	Valid Values
neighborhood	Defines the Network Neighborhood will be used for the RFC 2544 test.	Any Network Neighborhood currently stored on the system.
packetType	Sets the packet type for traffic on the wire.	ethernet, ip, udp, icmp, or tcp
payloadType	Establishes how the payload content is determined.	**0** – Payload is all 0s. **1** – Payload is all 1s. **random** – Payload is defined using random Hex values. **increment** – Payload is defined using ascending values starting at 0. **decrement** – Payload is defined using descending values starting at 0xff. **User-Defined** – Payload is defined using standard hexadecimal notation. If the payload is smaller than the packet size, then the Hex value will be repeated until it meets the packet size; however, if the payload is a user-defined Hex value that is greater than the packet size, the value will be truncated.
payloadWidth	Defines the width of the data (in bits) being inserted into the payload.	8, 16, or 32
seriesType	Establishes how the frame sizes are determined.	random, step, rfc, or custom
stepRate	Defines the rate at which the data rate is incremented; this value is used only if the mode is set to **step**.	0-100

Parameter	Description	Valid Values
stepduration	Defines the duration of each iteration.	1-1,000,000,000
stepdurationApplication	Establishes how the duration for the test is determined; you can either set the duration for each iteration (`periteration`) or set the duration for the entire test (`total`).	periteration or total
stepdurationunits	Defines the unit of measurement used for stepduration.	seconds, minutes, or hours

Creating a Session Sender Lab Test

You will need to create an object for the Session Sender Lab test. Within the object, you will use the `$bps createSessionLabTest` command to create the test.

For information on the Session Sender Lab test, see the Session Sender Lab General Information on page 892section.

Syntax

Use the `$bpscreateSessionLabTest` command to create the Session Sender Lab test, as shown in the following example. Additionally, the example below creates an object for the Session Sender Lab test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter.

```
set $sessionObjectName [$connectionObjectName
createSessionLabTest]
```

breaks down the elements of creating a Session Sender Lab test.

Creating a Session Sender Lab Test

Element	Description
`$sessionObjectName`	The object created for the Network Neighborhood.
`$connectionObjectName`	The name of the object created for the connection.
`createSessionLabTest`	The command to create a Session Sender Lab test.

Example

```
% set s [$bps createSessionLabTest]
::bps::BPSConnection::bPSConnection0::sessionLabClient0
% $s configure
{-aging {} {}} {-appProfile {} {}} {-dataType http } {-dstPortDist.max 1024
1024} {-dstPortDist.min 1 1} {-duration 00:01:00 00:01:00} {-dut
{BreakingPoint
Default} {BreakingPoint Default}} {-flowSize 4096 4096} {-maximumConcurrent
{}
{}} {-maximumConcurrentSession {} {}} {-maximumRate 5000 5000}
{-minimumConcurrent {} {}} {-minimumConcurrentSessions {} {}} {-minimumRate
10
10} {-neighborhood {BreakingPoint Switching} {BreakingPoint Switching}}
{-resetConnectionsBetweenTests {} {}}
{-retries {} {}} {-retry_quanta {} {}} {-srcPortDist.max 65535 65535}
{-srcPortDist.min 2049 2049} {-steadyBehavior {} {}} {-stepRate.num 10 10}
{-stepRate.type percent percent} {-stepdurationApplication periteration
periteration} {-testMode maxSustainedRate maxSustainedRate} {-testType
layer4
layer4}
% $s configure -aging 100 -srcPortDist.min 100
% $s save
% $s run
```

You can also create a Session Sender Lab test using a template.

Example

```
% set t [$bps createTest -template {Session Lab}]; #creates a Session Sender
Lab test based on an existing Session Lab template
```

Setting Session Sender Lab Test Commands

Once you have created an object for the Session Sender Lab test, you can call the object to list all the commands that are available for the Session Sender Lab test (e.g., `$sessionLabObjectName`).

Most of the commands are common to other features in the BreakingPoint System, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

Session Sender Lab Test Parameters

In order to list the parameters that you can configure for the Session Sender Lab test, use the following syntax:

```
set sessionLabObjectName [$connectionObjectName createsessionLabTest]

$sessionLabObjectName configure -option
```

The `configure` command will list all parameters available for the Session Sender Lab test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

Session Sender Lab Test Parameters

Parameter	Description	Valid Values
aging	The time, expressed in seconds, that an actively-closed TCP connection will remain in the flow table in the TIME_ WAIT state after closing.	0 – 120
appProfile	Sets the Application Profile that determines the mix of applications that will be used in the traffic.	A BreakingPoint Application Profile or a custom Application Profile
dataType	Selects the method used to determine the maximum TCP connection establishment rate through or with the DUT.	0, 1, Random, HTTP, User Defined
dstPortDist.max	Sets the maximum destination port number, if dstPortDist.type is Range or Random.	0 – 65,535
dstPortDist.min	Sets the minimum destination port number, if dstPortDist.type is Range or Random. Otherwise, this will be the value used for the destination port.	0 – 65,535
duration	Sets the length of the test.	1 – 120
dut	Searches for the device to be tested and its corresponding Network Neighborhood.	A BreakingPoint DUT or a custom DUT
maximumConcurrent	Sets the maximum number of simultaneous sessions that will exist concurrently during the test duration.	1 – 9, 223, 372, 036, 854, 775, 807
maximumRate	Limits the maximum connection establishment rate for the ramp up phase when not in Calculated mode.	1 – 9, 223, 372, 036, 854, 775, 807
minimumConcurrent	The number of sessions that must open to pass the test.	1 – 9, 223, 372, 036, 854, 775, 807

Parameter	Description	Valid Values
minimumRate	Specifies the connection establishment rate to be used at the start of the ramp up phase when not in Calculated mode. Must be less than or equal to Maximum Rate.	1 – 9, 223, 372, 036, 854, 775, 807
neighborhood	Searches for the device to be tested and its corresponding Network Neighborhood.	A BreakingPoint Network Neighborhood or a custom Network Neighborhood
raw_flag	Allows the specification of the TCP flags as decimal values.	-1 – 4095
resetConnectionsBetweenTests	Resets connections between test runs.	true or false
retries	Sets the number of times a connection is attempted before it is canceled.	0 – 7
retry_quanta	Sets the amount of time that elapses before a connection is retried.	100 – 2,000
srcPortDist.max	Sets the maximum source port number, if srcPortDist.type is range or random.	0 – 65,535
srcPortDist.min	Sets the minimum source port number, if srcPortDist.type is range or random. Otherwise, this will be the value used for the source port.	0 – 65,535
steadyBehavior	Defines the test behavior during the steady-state phase. Useful for examining connection tracking and timeout behavior of a device under test, as well as maintaining a steady load with a sustained number of simultaneous sessions.	cycle hold cycleRstClose cycleRst
stepRate.num	Defines the rate at which the data rate is incremented.	1 – 100
stepRate.type	Defines how the data rate is incremented.	literal percent

Parameter	Description	Valid Values
stepdurationApplication	Establishes how the duration for the test is determined; you can either set the duration for each iteration (periteration) or set the duration for the entire test (total).	periteration or total
testMode	The mode of the test being run.	maxSessionOpenRate maxSustainedRate maxConcurrentSessions
testType	The type of test being run.	layer2 layer3 layer4 stackscrambler appsim playback security_all aggregate_statistics clientsim sc_aggregate_statistics

Creating a Resiliency Score

You will need to create an object for the Resiliency Score. Within the object, you will use the `$bps createResiliencyTest` command to create the test.

For information on the Resiliency Score, see the Resiliency Score Lab General Information on page 900 section.

Syntax

Use the following syntax to run a Resiliency Score test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter.

```
set resiliencyTestObjectName [$connectionObjectName
createResiliencyTest]
```

breaks down the elements of creating a Resiliency Score test.

Creating a Resiliency Score Test

Element	Description
resiliencyTestObjectName	The name of the Resiliency Test object created for the test.
$connectionObjectName	The name of the object created for the connection.
createResiliencyTest	The command to create a Resiliency Score test.

Example

```
set t [$bps createResiliencyTest -name MyTest1]
$t configure
$t configure -deviceType Router
$t configure -runSecurity false
$t configure -speed 10000; # You can get a Network Neighborhood object, which
you can use to query how your device should be set up
set n [$t getNeighborhood]; # The run, cancel, and wait commands work just like
other tests, except you can also specify -runType to choose between Validate,
Quick, or Full.
proc echo {args} {puts $args}
$t run -runType Validate
$t run -progress "bps::textprogress stdout" -runType Quick
$t run -rtstats echo -async echo -runType Full
# rt stats gives the information shown in the dials of the UI as percentages:
# networkInterface deepPacketInspection sessionTable cpu routingEngine
# and threatDetectionEngine, as percentages
```

Setting Resiliency Score Test Commands

Once you have created an object for the Resiliency Score test, you can call the object to list all the commands that are available for the Resiliency Score test (e.g., `$resiliencyTestObjectName`).

Most of the commands are common to other features in the BreakingPoint system, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

lists the available Resiliency Score test commands and their descriptions.

Resiliency Score Test Commands

Command	Description
`cancel`	Cancels a test that has been running with the -async attribute.
`cget -option`	Retrieves the setting of an option.
`configure ?-option? ?value -option value...?`	Sets the value for a parameter.
`getNeighborhood`	Returns the Network Neighborhood used for the test context.
`isPending`	Returns the status of a job started using the -async option. Returns true if job is pending, returns false if job is not pending.
`resultId`	Returns the variable.
`run ?arg arg ...?`	Runs the test.
`wait`	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the -async attribute.

Resiliency Score Test Parameters

In order to list the parameters that you can configure for the Resiliency Score test, use the following syntax:

```
set resiliencyTestObjectName [$connectionObjectName createResiliencyTest]

$resiliencyTestObjectName configure -option
```

The `configure` command will list all parameters available for the Resiliency Score test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

Resiliency Score Test Parameters

Parameter	Description	Valid Values
speed	The target performance of the DUT. Test criteria such as offered bit rate and minimal performance criteria will be scaled automatically based on the claimed performance rate of the device.	100 – 1,000,000,000
runRobustness	Measure the ability of the device to correctly handle malformed traffic at different IP layers.	true or false

Parameter	Description	Valid Values
runSessionRate	Run traffic using realistic traffic engineered to stress the device's limits with respect to the rate of session churn.	true or false
runSecurity	Measure the ability of a device to correctly block exploit traffic.	true or false
name	Specify a name for a particular test.	Up to 256 alphanumeric and/or special characters can be used to define the name.
runThroughput	Measure the link speed of a device.	true or false
deviceType	Set the type of device to be tested.	Switch, Router, Firewall, Proxy, IPS, or UTM

Creating a Server Resiliency Score

You will need to create an object for the Server Resiliency Score. Within the object, you will use the `$bpscreateServerResiliencyTest` command to create the test.

For information on the Server Resiliency Score, see the Resiliency Score Lab General Information on page 900 section.

Syntax

Use the following syntax to run a Server Resiliency Score test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter

```
set serverResiliencyTestObjectName [$connectionObjectName
createServerResiliencyTest]
```

breaks down the elements of creating a Server Resiliency Score test.

Creating a Server Resiliency Score Test

Element	Description
`serverResiliencyTestObjectName`	The name of the Server Resiliency Test object created for the test.
`$connectionObjectName`	The name of the object created for the connection.
`createServerResiliencyTest`	The command to create a Resiliency Score test.

Example

```
set t [$bps createServerResiliencyTest -name MyTest1]
$t configure
{-neighborhood {} {}} {-numFileSystem 1 1} {-name {TCL Server Resiliency Test} MyTest1} {-numUsers 200 200} {-numWeb 1 1} {-numDb 1 1} {-numMail 1 1} {-deviceType {} Server}
$t listNeighborhoods
$t configure -neighborhood {Default App Server}
$t run -runType Validate -progress "bps::textprogress stdout" -async echo
$t run -runType AtLeast -progress "bps::textprogress stdout" -rtstats echo -async echo
$t run -runType Exactly
$t cancel
$t wait
```

Setting Server Resiliency Score Test Commands

Once you have created an object for the Server Resiliency Score test, you can call the object to list all the commands that are available for the Server Resiliency Score test (e.g., `$serverResiliencyTestObjectName`).

Most of the commands are common to other features in the BreakingPoint system, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

lists the available Resiliency Score test commands and their descriptions.

Server Resiliency Score Test Commands

Command	Description
`cancel`	Cancels a test that has been running with the -async attribute.
`cget -option`	Retrieves the setting of an option.
`configure ?-option? ?value -option value...?`	Sets the value for a parameter.
`isPending`	Returns the status of a job started using the -async option. Returns true if job is pending, returns false if job is not pending.

Command	Description
listNeighborhoods	Performs a search for Resiliency compatible neighborhoods.
resultId	Returns the variable.
run ?arg ...?	Runs the test.
wait	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the -async attribute.

Server Resiliency Score Test Parameters

In order to list the parameters that you can configure for the Resiliency Score test, use the following syntax:

```
set serverResiliencyTestObjectName [$connectionObjectName
createServerResiliencyTest]
$serverResiliencyTestObjectName configure -option
```

Creating a Lawful Intercept Test

You will need to create an object for the Lawful Intercept test. Within the object, you will use the `$bps createLawfulInterceptTest` command to create the test.

For more information on the Lawful Intercept test, see theLawful Intercept General Information on page 916section.

Syntax

Use the `createLawfulInterceptTest` command to create a Lawful Intercept test, as shown in the following example. Additionally, the example below creates an object for the Lawful Intercept test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter.

```
set lawfulInterceptObjectName [$connectionObjectName
createLawfulInterceptTest]
```

breaks down the elements of creating a Lawful Intercept test.

Creating a Lawful Intercept Test

Element	Description
lawfulInterceptObjectName	A name for the Lawful Intercept object.
$connectionObjectName	The name of the object created for the connection.

Element	Description
createLawfulInterceptTest	The command to create the Lawful Intercept test.

Example

```
% set t[$bps createLawfulInterceptTest]; #creates the connection object
::bps::BPSConnection::bPSConnection0::lawfulInterceptClient1
% $t configure
{-appProfile {BreakingPoint Enterprise} {BreakingPoint Enterprise}}
{-concurrentSessions 10000 10000} {-dataRate 200 200} {-duration 00:00:30
00:00:30} {-dut {BreakingPoint Default} {BreakingPoint Default}} {-
neighborhood
{BreakingPoint Switching} {BreakingPoint Switching}} {-sessionsPerSecond
1000
1000} {-target1.active true true} {-target1.fieldType phone phone}
{-target1.intervalType time time} {-target1.ipTrigger {} {}}
{-target1.quantityInterval {} {}} {-target1.superflowName {BreakingPoint
Gmail
(Lawful Intercept)} {BreakingPoint Gmail (Lawful Intercept)}}
{-target1.timeInterval 00:00:30 00:00:30} {-target2.active false false}
{-target2.fieldType {} {}} {-target2.intervalType quantity quantity}
{-target2.ipTrigger {} {}} {-target2.quantityInterval {} {}}
{-target2.superflowName {} {}} {-target2.timeInterval {} {}} {-
target3.active
false false} {-target3.fieldType {} {}} {-target3.intervalType quantity
quantity} {-target3.ipTrigger {} {}} {-target3.quantityInterval {} {}}
{-target3.superflowName {} {}} {-target3.timeInterval {} {}}

% $t configure-target1.active true
% $t listSuperflows
{BreakingPoint HTTP Request (Lawful Intercept)} {BreakingPoint SMTP Email
(Lawful Intercept)} {BreakingPoint IMAPv4-Advanced (Lawful Intercept)}
{BreakingPoint SIP/RTP Call (Lawful Intercept)} {BreakingPoint Windows Live
Messenger v15 (Lawful Intercept)} {BreakingPoint Gmail (Lawful Intercept)}
% $t listSuperflows HTTP
{BreakingPoint HTTP Request (Lawful Intercept)}

% $t run -progress {bps::textprogress stdout} -rtstats echo -async {echo
done}
```

Syntax

You can also create a Lawful Intercept test using the `createTest` command. Use the `-name` attribute to save an existing Lawful Intercept test under a new name. You can use the `-force` attribute to overwrite any existing Lawful Intercept test with the same name. If you do not name the Lawful Intercept test when you create it, the system will give it a default name (e.g., lawfulInterceptClient0).

Example

```
% set t [$bps createTest -template {Lawful Intercept Test}]; #creates a Lawful Intercept test based on an existing Lawful Intercept Test template
```

```
% $t configure -name myLawfulInterceptTest1; #renames test myLawfulInterceptTest
```

```
% $t save -name myLawfulInterceptTest1 -force; #saves the test under the name myLawfulInterceptTest1 and overwrites any existing Lawful Intercept test with the same name
```

```
% set t [$bps createLawfulInterceptTest -template myLawfulInterceptTest1]; #creates a new Lawful Intercept test based on the myLawfulInterceptTest1 test
```

Setting Lawful Intercept Test Commands

Once you have created an object for the Lawful Intercept test, you can call the object to list all the commands that are available for the Lawful Intercept test (e.g., `$lawfulInterceptObjectName`).

Most of the commands are common to other features in BreakingPoint, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

lists the available Lawful Intercept test commands and their descriptions.

Lawful Intercept Test Commands

Command	Description
`cancel`	Cancels a test that has been running with the `-async` attribute.
`cget option`	Retrieves the setting of an option.
`clearResults`	Clears the stored results of a test context.
`configure ?arg arg ...?`	Sets the value for a parameter.
`exportReport ?arg arg ...?`	Exports the report in PDF, XLS, ZIP, or HTML.
`getTargetParameters`	Takes a target number as an argument and returns a list of option name/default value pairs. The set of options depends on the fieldType of that target. The options that are returned can be set or returned by using the `cget` and `configure` commands.
`isPending`	Returns the status of a job started using the -async option. Returns true if job is pending, returns false if job is not pending.

Command	Description
`listSuperflows ?arg arg?`	Lists the Super Flows that are available.
`resultId`	Returns the variable.
`run ?arg arg ...?`	Runs the test.
`save ?arg arg ...?`	Saves the current test.
`validate ?arg arg ...?`	Allows you verify that your test has not exceeded the available bandwidth limitations and hardware resources.
`wait`	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the -async attribute.

Lawful Intercept Test Parameters

In order to list the parameters that you can configure for the Lawful Intercept test, use the following syntax:

```
set lawfulInterceptObjectName [$connectionObjectName createLawfulInterceptTest]

$lawfulInterceptObjectName configure -option
```

The `configure` command will list all parameters available for the Lawful Intercept test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

Lawful Intercept Parameters

Parameter	Description	Valid Values
appProfile	The Application Profile to be used in your test.	A valid App Profile
concurrentSessions	Sets the number of concurrent flows to be generated in your test.	1 – 20,000,000
dataRate	Sets the maximum speed (in Mbps) at which traffic is to be transmitted to the device for both background traffic and targeted Super Flows.	1 – 10,000
duration	Sets the length of the test.	Valid values in the form of hh:mm:ss
dut	The device to be tested.	A valid device under test

Parameter	Description	Valid Values
neighborhood	The Network Neighborhood to be used in your test.	A valid Network Neighborhood
sessionsPerSecond	Sets the number of flows per second for both background traffic and targeted Super Flows.	1 – 750,000
target1.active	Activates or deactivates target 1.	true or false
target1.fieldType	Sets the type of pattern to be searched for in the test.	phone, taxid, creditcard, pattern, fileentries, directentries
target1intervalType	Sets the type of interval to be used for the relative amount.	time or quantity
target1.ipTrigger	Sets the parameters of the trigger used in your test.	A domain name on interface 1 of the Network Neighborhood
target1.quantityInterval	Sets how frequently (in time) the pattern you are searching for appears in your test.	When intervalType is set to `quantity`, this value represents the number of flows between instances of the needle.
target1.superflowName	Sets the name of the Super Flow to be used in your test.	The name of the Super Flow to be used as the needle.
target1.timeInterval	Sets how frequently (in the number of flows) the pattern you are searching for appears in your test.	When intervalType is set to `time`, this value represents amount of time between needles. Can either be a number of seconds, or a time of the form hh:mm:ss.
target2.active	Activates or deactivates target 2.	true or false
target2.fieldType	Sets the type of pattern to be searched for in the test.	phone, taxid, creditcard, pattern, fileentries, directentries
target2.intervalType	Sets the type of interval to be used for the relative amount.	time or quantity
target2.ipTrigger	Sets the parameters of the trigger used in your test.	A domain name on interface 2 of the Network Neighborhood
target2.quantityInterval	Sets how frequently (in time) the pattern you are searching for appears in your test.	When intervalType is set to `quantity`, this value represents the number of flows between instances of the needle.

Parameter	Description	Valid Values
target2.superflowName	Sets the name of the Super Flow to be used in your test.	The name of the Super Flow to be used as the needle.
target2.timeInterval	Sets how frequently (in the number of flows) the pattern you are searching for appears in your test.	When intervalType is set to `time`, this value represents amount of time between needles. Can either be a number of seconds, or a time of the form hh:mm:ss.
target3.active	Activates or deactivates target 3.	true or false
target3.fieldType	Sets the type of pattern to be searched for in the test.	phone, taxid, creditcard, pattern, fileentries, directentries
target3.intervalType	Sets the type of interval to be used for the relative amount.	time or quantity
target3.ipTrigger	Sets the parameters of the trigger used in your test.	A domain name on interface 3 of the Network Neighborhood
target3.quantityInterval	Sets how frequently (in time) the pattern you are searching for appears in your test.	When intervalType is set to `quantity`, this value represents the number of flows between instances of the needle.
target3.superflowName	Sets the name of the Super Flow to be used in your test.	The name of the Super Flow to be used as the needle.
target3.timeInterval	Sets how frequently (in the number of flows) the pattern you are searching for appears in your test.	When intervalType is set to `time`, this value represents amount of time between needles. Can either be a number of seconds, or a time of the form hh:mm:ss.

Creating a Multicast Test

You will need to create an object for the Multicast test. Within the object, you will use the `$bpscreateMulticastTest` command to create the test.

For more information on the Multicast Test, see theMulticast General Information on page 921section.

Syntax

Use the `$bpscreateMulticastTest` command to create a Multicast test. Additionally, the example below creates an object for the Multicast test.

```
set multicastObjectName [$connectionObjectName
createMulticastTest]
```

breaks down the elements of creating a Multicast test.

Creating a Multicast Test

Element	Description
multicastObjectName	The name of the Multicast Test object created for the test.
$connectionObjectName	The name of the object created for the connection.
createMulticastTest	The command to create a Multicast test.

Example

```
% set t [$bps createMulticastTest]
::bps::BPSConnection::bPSConnection0::multicastClient0
% $t networkTypes
small {Subscriber IPs range from 10.10.2.1 - 10.10.18.254. Up to 16
simultaneous subscriber subnets will be used using a /24 netmask.} medium
{Subscriber IPs range from 10.10.2.1 - 10.10.18.254. Up to 256 simultaneous
subscriber subnets will be used using a /28 netmask.} large {Subscriber IPs
range from 10.10.2.1 - 10.10.66.254. Up to 1024 simultaneous subscriber
subnets
will be used using a /28 netmask.}

% $t configure
{-duration 00:00:30 00:00:30} {-networkType medium medium}
% $t configure -networkType small

% $t getSources
1 {ipAddress 10.1.1.2 groupAddress 224.0.0.1 rate 100} 2 {ipAddress 10.1.1.3
groupAddress 224.0.0.1 rate 1000} 3 {ipAddress 10.2.1.2 groupAddress
225.0.0.1
rate 10000}
% $t addSource -ipAddress 1.0.0.1 -groupAddress 224.0.0.1 -rate 100
4
% $t removeSource 3

% $t getSubscribers
1 {maxSubscribers 100 groupAddress 224.0.0.1 sourceSpecific true sources
{10.1.1.2 10.1.1.3}} 2 {maxSubscribers 1000 groupAddress 225.0.0.1
sourceSpecific false sources {}}
% $t addSubscribers -maxSubscribers 4 -groupAddress 224.0.0.1 \
-sourceSpecific true -sources {
1.2.3.4
}
3
% $t removeSubscribers 3
% $t run -progress {bps::textprogress stdout} -rtstats echo -async {echo
done}
```

You can also create a Multicast test using a template.

Example

```
% set t [$bps createTest -template {Multicast Test}]; #creates a Multicast test
based on an existing Multicast test template
```

Setting Multicast Test Commands

Once you have created an object for the Multicast test, you can call the object to list all the commands that are available for the Multicast test (e.g., `$multicastobjectName`).

Most of the commands are common to other features in the BreakingPoint system, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

The table below lists the available Multicast Test commands and their descriptions.

Multicast Test Commands

Command	Description
`addSource ?arg arg ...?`	Adds a source that will generate UDP multicast data streams to your test.
`addSubscribers ?arg arg ...?`	Allows you to define the subscriber (client) profiles to be used in your test.
`cancel`	Cancels a test that has been running with the `-async` attribute.
`cget -option`	Retrieves the setting of an option.
`clearResults`	Clears the stored results of a test context.
`configure ?arg arg ...?`	Sets the value for a parameter.
`exportReport ?arg arg ...?`	Exports the report in PDF, XLS, ZIP, or HTML.
`getSources`	Returns a list of sources used in the test.
`getSubscribers`	Returns a list of subscriber (client) profiles being used in your test.
`isPending`	Returns the status of a job started using the -async option. Returns true if job is pending, returns false if job is not pending.
`networkTypes`	Allows you to get to select a network type.
`removeSource index`	Removes a source that generates UDP multicast data streams from the test.
`removeSubscribers index`	Removes a subscriber (client) profile from your test.
`resultId`	Returns the variable.
`run ?arg arg ...?`	Runs the test.

Command	Description
`save ?arg arg ...?`	Saves the current test.
`validate ?arg arg ...?`	Allows you verify that your test has not exceeded the available bandwidth limitations and hardware resources.
`wait`	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the `-async` attribute.

Multicast Test Parameters

In order to list the parameters that you can configure for the Multicast test, use the following syntax:

```
set multicastObjectName [$connectionObjectName createMulticastTest]

$multicastObjectName configure -option
```

The `configure` command will list all parameters available for the Multicast test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

Multicast Parameters

Parameter	Description	Valid Values
duration	Sets the length of the test.	This time value can either be a number of seconds, or a time of the form hh:mm:ss
networkType	Sets the type of network used in the test.	small medium large

Creating an LTE Test

You will need to create an object for the LTE test. Within the object, you will use the `$bpscreateLTETest` command to create the test.

For more information on the LTE test, see Long Term Evolution General Information on page 931.

Syntax

Use the `createLTETest` command to create an LTE test. Additionally, the example below creates an object for the LTE test.

Please note that for all parameters you do not explicitly define, the test will use the pre-existing value defined for that parameter.

```
set LTEObjectName [$connectionObjectName createLTETest]
```

breaks down the elements of creating an LTE test.

Creating an LTE Test

Element	Description
LTEObjectName	The name of the LTE Test object created for the test.
$connectionObjectName	The name of the object created for the connection.
createLTETest	The command to create an LTE test.

Example

```
% set t [$bps createLTETest]
::bps::BPSConnection::bPSConnection0::lTEClient0
% $t configure
{-alloc_rate 2 2} {-apn internet internet} {-appProfile {BreakingPoint Mobile
User} {BreakingPoint Mobile User}} {-dataRate 1000 1000} {-dnsServerIP 10.0.1.3
10.0.1.3} {-domainName example.org example.org} {-duration 00:00:30 00:00:30}
{-gateway 10.0.1.1 10.0.1.1} {-imsi_base 240011234567000 240011234567000} {-
msisdn_base 001123456782319 001123456782319} {-netaddr 10.0.1.0 10.0.1.0} {-
netmask 24 24} {-numEnodeB 65 65} {-numUE 1 1} {-num_dedicated_bearers 2 2} {-
operatorVariant 8FB21E23AE9123923AE428F8FB3428EF
8FB21E23AE9123923AE428F8FB3428EF} {-pdn_gateway 10.0.1.1 10.0.1.1} {-pdn_
netaddr 10.0.1.0 10.0.1.0} {-pdn_netmask 24 24} {-pdn_numHosts 200 200} {-pdn_
router 10.0.1.2 10.0.1.2} {-pdn_startingIP 10.0.1.3 10.0.1.3} {-plmn_mcc 111
111} {-plmn_mnc 12 12} {-sctp_over_udp false false} {-sctp_sport 0 0} {-
secretKey 12FF98428EF13AE823AE9B23B23428EF 12FF98428EF13AE823AE9B23B23428EF} {-
startingIP 10.0.1.18 10.0.1.18}
% $t configure -numUE 100 -numEnodeB 4
% $t getMMEs
mmepool
% $t addMME -hostname foo
2
% $t getMMEs
mmepool foo
% $t removeMME 2
% $t save
% $t run
```

You can also create an LTE test using a template.

Example

```
% set t [$bps createTest -template {BreakingPoint LTE Lab}]; #creates an LTE
test based on an existing BreakingPoint LTE Lab test template
```

Setting LTE Test Commands

Once you have created an object for the LTE test, you can call the object to list all the commands that are available for the LTE test (e.g., `$LTEObjectName`).

Most of the commands are common to other features in the BreakingPoint System, such as creating tests, test series, and multi-box tests. For example, common commands include: `run`, `configure`, `exportReport`, `resultId`, `save`, and `wait`.

lists the available LTE test commands and their descriptions.

LTE Test Commands

Command	Description
`addMME ?arg arg ...?`	Adds an eNodeB/MME (Mobility Management Entity) client to a subnet.
cancel	Cancels a test that has been running with the `-async` attribute.
`cget option`	Retrieves the setting of an option.
`clearResults`	Clears the stored results of a test context.
`configure ?arg arg ...?`	Sets the value for a parameter.
`exportReport ?arg arg ...?`	Exports the report in PDF, XLS, ZIP, or HTML.
`getComponents`	Returns the components used by the test.
`getMMEs`	Returns the MMEs used by the test.
`isPending`	Returns the status of a job started using the -async option. Returns true if job is pending, returns false if job is not pending.
`removeMME index`	Removes an LTE eNodeB/MME (Mobility Management Entity) client from a subnet.
`resultId`	Returns the variable.
`run ?arg arg ...?`	Runs the test.
`save ?arg arg ...?`	Saves the current test.
`validate ?arg arg ...?`	Allows you verify that your test has not exceeded the available bandwidth limitations and hardware resources

Command	Description
`wait`	Waits for that test to complete before continuing execution. This command is typically used after running a test that uses the `-async` attribute.

LTE Test Parameters

In order to list the parameters that you can configure for the LTE test, use the following syntax:

```
set LTEObjectName [$connectionObjectName createLTETest]

$LTEObjectName configure -option
```

The `configure` command will list all parameters available for the LTE test. Additionally, it will list the default setting and the current setting for each parameter. The following tablelists the parameters and their descriptions.

LTE Parameters

Parameter	Description	Valid Values
alloc_rate	Sets the rate at which UE bandwidth is allocated in the test	1 – 9,223,372,036,854,775,807
apn	The type of network connection to create	Up to 256 alphanumeric and/or special characters can be used to define the apn
appProfile	This parameter defines the Application Profile that will be used in the test	A valid Application Profile
dataRate	This parameter defines the bandwidth for the UEs in the test	1 – 4,294,967,295
dnsServerIP	The address of the DNS to use when resolving hostnames	A valid IPv4 address
domainName	A name for the domain	Up to 256 alphanumeric and/or special characters can be used to define the domain name
duration	The duration of the test	This time value can either be a number of seconds, or a time of the form hh:mm:ss
gateway	The default gateway that each eNodeB will be configured with	A valid IPv4 address

Parameter	Description	Valid Values
imsi_base	Identifies the SIM card of each device	May be left blank or contain 11 to 15 digits
msisdn_base	A secondary unique identifier for each device, This number identifies a subscription in the UMTS network	May be left blank or contain 11 to 15 digits
netaddr	Defines a 32-bit or 128-bit base network address	A valid IPv4 address
netmask	Defines the subnet mask for the Network Address	A valid IPv4 address
numEnodeB	Sets the number of eNodeB clients to be used in the test.	1 – 4096
numUE	The total number of devices to simulate	0 – 6,000,000
num_ dedicated_ bearers	Specifies the number of UE dedicated bearers to use in the test.	0 – 10
operatorVariant	Specifies a unique value originally assigned by the UE manufacturer. The operator variant is usually unique to each brand of UE.	A 32-character hexadecimal
pdn_gateway	The point of exit and entry of traffic for the UE	A valid IPv4 address
pdn_netaddr	The base pdn network address	A valid IPv4 address
pdn_netmask	The netmask for the network address	A valid IPv4 address
pdn_numHosts	The total number of separate simulated hosts that will be used to provide Internet services	A numeric value that is less than the number of hosts defined by the netmask (256)
pdn_router	The PDN IP address of the BreakingPoint Storm	A valid IPv4 address
pdn_startingIP	The first IP address that the Internet services will use	Must fall within the network defined by the network address and netmask
plmn_mcc	The mobile country code of the PLMN	A mobile country code consisting of 3 numeric characters

Parameter	Description	Valid Values
plmn_mnc	The mobile network code of the PLMN	A valid mobile network code consisting of 2 or 3 numeric characters
sctp_over_udp	Enables or disables the tunneling of SCTP over UDP.	true or false
secretKey	The base value for a secret key that is generated for each UE.	A 32-character hexadecimal
startingIP	The first IP address that the eNodeBs will be given.	A valid IPv4 address

Validating Test Lab Tests

The `validate` command allows you verify that your test has not exceeded the available bandwidth limitations and hardware resources.

Syntax

Use the following syntax to validate your test lab tests:

```
%testObject
validate
```

Example

```
% set t [$bps createLTETest]; #creates an LTE test
object.
% $t validate; #validates the LTE test object.
```

Canceling a Test Lab Test

Use the `cancel` command to cancel a test lab test. You can use the `-force` attribute to force an existing test lab test with the same name into an idle state.

Syntax

Use the following syntax to cancel the test.

```
$testObject
cancel
```

Load Profiles

The following commands are described in this section:

Listing Load Profiles

Use the `listLoadProfiles` command to display a list of all the Load Profiles that are available on the system. The `listLoadProfiles` command by itself will retrieve a list of all App Profiles.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display Super Flows created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint-created Super Flows, or `custom`, which will return a list of all user-created Super Flows. You will use the `-timeunit` and `-timeinterval` attributes to list Super Flows by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

Syntax

Use the following syntax to display a list of existing Load Profiles.

```
$connectionObject
listLoadProfiles
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listLoadProfiles; #displays a list of Load Profiles on the system
```

Creating Load Profiles

You will need to create an object to store the Load Profile and use the `createLoadProfile` command to create a Load Profile.

When creating the Load Profile, you can use the `-name` optional attribute to name it. If you do not name the Load Profile when you create it, the system will give it a default name (e.g., LoadProfileClient0). Additionally, you can used the `-template` attribute to choose an existing Load Profile on which to base the Load Profile. If you do not base your Load Profile on a template, then the Load Profile will not contain any phases.

Once you create the Load Profile object, you can use its commands to create Load Profiles and phases.

Note:
- You must save the Load Profile so that it will be stored on the system for later use.
- If you do not use a template to create your Load Profile, the system will automatically create one based on BreakingPoint Default.

Syntax

Use the following syntax to create a Load Profile.

```
set loadProfileObjectName [$connectionObject createLoadProfile-name
loadProfileName -template {Load Profile Name}
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
set z [$var createLoadProfile -template {BreakingPoint Maximum Sessions per
second} -name profile2; #creates a Load Profile called profile 2 based on the
BreakingPoint Maximum Sessions per second profile
```

Listing Phases in a Load Profile

Use the `getPhases` command to get a list of phases in a Load Profile. The `getPhases` command will return a list of all the phases in the Load Profile and the parameter configurations for each phase.

Each phase in a Load Profile is assigned a flow ID; this value is based on the order in which the phase was added. For example, the first phase will be phase '1', the second phase will be phase '2', and so forth.

Note: If you remove a phase from a Load Profile, then the flows will be resequenced to the flow ID the preceding flow ID (e.g., flow '3' will become flow '2').

Syntax

Use the following syntax to display a list of phases in a Load Profile.

```
$loadProfileObjectName
getPhases
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y getPhases; #displays a list of phases and their configurations
```

Adding Phases to a Load Profile

Use the `addPhase` command to add a phase to a Load Profile. All Load Profiles will start at phase '0', which is the ramp up phase. The last phase in a Load Profile is the ramp down phase. All phases in between are steady-state phases. If phases are deleted, they will be resequenced to the previous phase value.

To add a phase before the ramp up phase, you can use the special index 'end' (e.g., $loadProfileObjectName addPhase end). This will place the new phase before the ramp up phase. The system will reassign phase IDs to the ramp down phase and the new phase accordingly.

Most of the time, you will want to use 'end' to add phases to the Load Profile. This convenient indexer automatically places the phase before the ramp down phase, so you do not have to manually track all the phase IDs.

When you add a phase to a Load Profile, each phase will automatically be assigned a phase ID. This value is based on the position at which the phase was added. For example, the first phase will be phase '1', the second phase will be phase '2', and so forth.

Note: If you add a phase that uses the same phase ID as an existing phase, then the system will resequence the phase to the following phase ID. For example, if you add phase '3' to a Load Profile, the current phase '3' will be resequenced to phase '4', and so forth.

Note: If you remove a phase from a Load Profile, then the flows will be resequenced to the flow ID the preceding phase ID (e.g., phase '3' will become phase '2').

When you add a phase to a Load Profile, you can specify the phase's parameter configurations. The parameters that you can set for a Load Profile include the phase duration, maximum number of simultaneous sessions, session rate, data rate scope, data rate, and data rate unit. For more information on the Tcl equivalent for these parameters, Session Sender Parameters on page 1116.

Note: If you do not specify any parameter configurations for a phase, the system will assign the parameters their default values.

Syntax

Use the following syntax to add a phase to a Load Profile.

```
$loadProfileObjectName addPhase phase# -parameter value
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y getPhases; #displays a list of phases and their configurations
```

```
$y addPhase end; #adds a phase before the ramp down phase
```

```
$y addPhase 2; #adds phase '2'
```

```
$y addPhase 3 -rateDist.unit fps; #adds phase '3' and assigns the data rate
unit to fps
```

Modifying Phases

Use the `modifyPhase` command to modify the parameter configurations for a phase. To see the current values for a phase, use the `getPhases` command; this will return a list of all phases in a Load Profile, and the parameter configurations for each phase.

Syntax

Use the following syntax to modify a phase.

```
$loadProfileObjectName modifyPhase phaseID -parameter
value
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y addPhase end -rateDist.unit fps; #adds a phase before the ramp down phase
and assigns the data rate unit to fps
```

Removing Phases from a Load Profile

Use the `removePhase` command to remove a phase from a Load Profile.

Note:
- You cannot remove the ramp up or ramp down phases.
- If you remove a phase from a Load Profile, then the flows will be resequenced to the flow ID the preceding phase ID (e.g., phase '3' will become phase '2').

Syntax

Use the following syntax to remove an phase from a Load Profile.

```
$loadProfileObjectName removePhase
phaseID
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y addPhase 4 -rateDist.unit fps; #adds phase 4 and assigns the data rate unit
to fps
```

```
$y removePhase 4; #removes phase 4 from the Load Profile
```

Saving a Load Profile As...

Use the `save` command and the `-name` attribute to save an existing Load Profile under a new name.

Note:
- The original Load Profile will still remain in the system.
- You can use -force true to overwite any Load Profiles with the same name.
- You must save the Load Profile before you can run it in a test.

Syntax

```
$loadProfileObjectName save-name
newLoadProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y addPhase 4 -rateDist.unit fps; #adds phase 4 and assigns the data rate unit
to fps
```

```
$y save; #saves the Load Profile
```

Deleting Load Profiles

Use the `deleteLoadProfile` command to delete a Load Profile from the system.

Syntax

Use the following syntax to remove a Load Profile from the system.

```
$connectionObject deleteLoadProfile
loadProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set y [$var createLoadProfile -name profile1]; #creates a Load Profile called
profile1
```

```
$y addPhase 4 -rateDist.unit fps; #adds phase 4 and assigns the data rate unit
to fps
```

```
$y save; #saves the Load Profile
```

```
$var deleteLoadProfile profile1; #deletes profile1 from the system
```

Deleting the Load Profile Object

Use the `itcl::delete` command to delete the Load Profile object.

Syntax

Use the following syntax to delete the Load Profile object.

```
itcl::deleteobject
$loadProfileObjectName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createLoadProfile -name LP1]; #creates a Load Profile called LP1
and a Load Profile object called 'x'
```

```
$x save; #saves the Load Profile
```

```
itcl::delete object $x; #deletes the Load Profile object
```

Evasion Profiles

The following commands are described in this section:

Listing Evasion Profiles

Use the `listEvasionProfiles` command to display a list of the Evasion Profiles that are contained within a Security Component.

Syntax

Use the following syntax to list the Evasion Profiles that are in a Security Component.

```
$connectionObjectName
listEvasionProfiles
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listEvasionProfiles; #returns a list of Evasion Profiles
```

Creating an Evasion Profile

Use the `createEvasionProfile` command to create an Evasion Profile.

Syntax

```
$evasionProfileObjectName createEvasionProfile
evasionProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
% set ep [$var createEvasionProfile]; #creates an Evasion Profile object named
'ep'
```

```
% set ep [$var createEvasionProfile -name "My Evasion Profile"]; # creates a
new Evasion Profile named "My Evasion Profile"
```

```
% set ep [$bps createEvasionProfile -name "My Evasion Profile" -template "An
Existing Evasion Profile"]; # creates an Evasion Profile named "My Evasion
Profile" using a template based on an existing Evasion Profile
```

Adding an Evasion Profile to a Security Component

Use the `createComponent` and `configure` commands to add an Evasion Profile to a Security component.

Example

```
set var [bps::connect 10..10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
% set t [$bps createTest -name "My Security Test"]; # creates an empty test
```

```
% set security [$t createComponent security "My Security Component" 1 2]; #
creates the security component for the test
```

```
% $security configure -attackPlan "My Strikelist" -attackProfile "My Evasion
Profile; # adds the Evasion Profile named "Default evasion settings" to the
security component
```

Deleting an Evasion Profile from a Security Component

Use the `deleteEvasionProfile` command to delete an Evasion Profile.

Syntax

```
$connectionObjectName deleteEvasionProfile
evasionProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
% $bps deleteEvasionProfile "My Evasion Profile"; # deletes the Evasion Profile
```

```
% $bps deleteEvasionProfile -force "My Evasion Profile"; # forces the deletion
```

Renaming an Evasion Profile

To rename an existing Evasion Profile, you must create a copy of it, save it under a new name, and delete the original version.

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set ep [$var createEvasionProfile -template "original"]; #creates a copy of the
original Evasion Profile
```

```
$ep save -name "new"; #names the saved Evasion Profile
```

```
$var deleteEvasionProfile "original"; #deletes the original version of the
Evasion Profile
```

Listing Evasion Profile Parameters

Each Evasion Profile has a set of evasion parameters. The evasion parameters that are available depend on the Strikes that the Evasion Profile contains; therefore, you must add Strikes to an Evasion Profile before you can assign evasion parameters to it.

Syntax

Use the following syntax to view the evasion options that are configurable for an Evasion Profile. The system will return a list of Evasion Profile Settings and their corresponding descriptions.

```
$evasionProfileObjectName getParameters
evasionProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set ep [$bps createEvasionProfile]; #creates an Evasion Profile object called
'ep'
```

```
set ep [$bps createEvasionProfile -template "An Existing Evasion Profile"]; #
creates an Evasion Profile from an existing template
```

```
$ep getParameters; #lists the parameters for the Evasion Profile
```

Adding a Parameter to an Evasion Profile

Use the `configure` command to add a parameter to an Evasion Profile.

Syntax

Use the following syntax to add an Evasion Profile parameter to an Evasion Profile.

```
$evasionProfileObjectName configure -evasionParameter true
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set ep [$var createEvasionProfile -name Series1]; #creates an Evasion Profile
called Series 1 and an Evasion Profile object called 'ep'
```

```
% $ep configure -SMTP.PadCommandWhitespace true; # adds the Evasion Profile
parameter -SMTP.PadCommandWhitespace to the Evasion Profile called Series 1
```

Removing an Existing Evasion Profile Parameter

Use the `unset` command to remove an existing Evasion Profile parameter.

Syntax

Use the following syntax to remove an existing Evasion Profile parameter from an Evasion Profile.

```
$evasionProfileObjectName unset -
evasionParameter
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set ep [$var createEvasionProfile -name Series1]; #creates an Evasion Profile
called Series 1 and an Evasion Profile object called 'ep'
```

```
% $ep configure -SMTP.PadCommandWhitespace true; # adds the Evasion Profile
parameter -SMTP.PadCommandWhitespace to the Evasion Profile called Series 1
```

```
% $ep unset -SMTP.PadCommandWhitespace; # removes the -
SMTP.PadCommandWhitespace parameter from the Evasion Profile
```

Application Profiles

The following commands are described in this section:

Listing App Profiles

Use the `listAppProfiles` command to display a list of all the App Profiles that are available on the system. This command is useful if you want to see which App Profiles you can modify or select for an App Sim test. The `listAppProfiles` by itself will retrieve a listing of all App Profiles; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display App Profiles created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint created App Profiles, or `custom`, which will return a list of all user-created App Profiles. Use the `-timeunit` and `-timeinterval` attributes to list App Profiles by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

This command also accepts a Google-formatted search string as a final argument.

Syntax

Use the following syntax to view a list of available App Profiles; this includes all canned and custom App Profiles.

```
$connectionObject
listAppProfiles
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listAppProfiles; #returns a list of all the App Profiles stored on the
system
```

```
$var listAppProfiles -userid admin; #returns a list of all App Profiles created
by the admin
```

```
$var listAppProfiles -class canned; #returns a list of all default App Profiles
```

```
$var listAppProfiles-timeunit day -timeinterval 2; #returns a list of all App
Profiles created two days ago
```

Creating App Profiles

You will need to create an object to store the App Profile and use the `createAppProfile` command to create the App Profile.

When you create an App Profile, you can choose to either use an existing App Profile as a template, or you can create an empty App Profile. Empty App Profiles will not contain any Super Flows. App Profiles that are created using a template will be a clone of the template used.

When creating the App Profile, you can use the `-name` attribute to name it. If you do not name the App Profile when you create it, the system will give it a default name (e.g., appProfileClient0).

Once you create the App Profile object, you can use its commands to create hosts and flows and add Super Flows.

Note: You must save the App Profile so that it will be stored on the system for later use.

Syntax

Use the following syntax to create an App Profile based on a template.

```
set appProfileObjectName [$connectionObject createAppProfile-template {App
Profile Name} -name appProfileName
```

Use the following syntax to create an empty App Profile.

```
set appProfileObjectName [$connectionObject createAppProfile -name
appProfileName]
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name httpProfile]; #creates an empty App Profile
called httpProfile and an App Profile object called 'x'
```

```
set y [$var createAppProfile -template {BreakingPoint Enterprise} -name
enterpriseProfile; #creates an App Profile called enterpriseProfile using a
canned App Profile as a template
```

```
$x save; #saves the App Profile for this object
```

```
$y save; #saves the App Profile for this object
```

Saving an App Profile As...

Use the `configure` command and the `-name` attribute to save an existing App Profile under a new name.

Note: The original App Profile will still remain in the system.

Syntax

```
$appProfileObjectName configure-name
newAppProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name httpProfile]; #creates an empty AppProfile
called httpProfile
```

```
$x configure -name webTraffic; renames httpProfile to webTraffic
```

```
$x save; saves the App Profile
```

Deleting an App Profile

Use the `deleteAppProfile` command to remove an App Profile from the system.

Syntax

Use the following syntax to delete an App Profile from the system.

```
$connectionObject deleteAppProfile
appProfileName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var deleteAppProfile httpProfile; #removes httpProfile
```

Removing the App Profile Object

Use the `itcl::delete` command to delete the App Profile object.

Syntax

Use the following syntax to delete the App Profile object.

```
itcl::delete object
$appProfileObjectName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name httpProfile]; #creates an App Profile called
httpProfile and an App Profile object called 'x'
```

```
$x save
```

```
itcl::delete object $x
```

Flows and Super Flows

The commands are described in this section:

Listing Super Flows

The `listSuperflows` command by itself will retrieve a listing of all Super Flows stored on the system; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display Super Flows created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint-created Super Flows, or `custom`, which will return a list of all user-created Super Flows. You will use the `-timeunit` and `-timeinterval` attributes to list Super Flows by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

This command also accepts a Google-formatted search string as a final argument.

Syntax

Use the following syntax to view a list of available Super Flows; this includes all canned and custom Super Flows.

```
$connectionObject
listSuperflows
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listSuperflows; #returns a list of all the Super Flows stored on the
system
```

```
$var listSuperflows -userid admin; #returns a list of all Super Flows created
by the admin
```

```
$var listSuperflows -class canned; #returns a list of all default Super Flows
```

```
$var listSuperflows -timeunit day -timeinterval 2; #returns a list of all Super
Flows created two days ago
```

```
$var listSuperflows "needle:true" ; #returns a list of all Super Flows that
contain needles
```

Creating Super Flows

You will need to create an object to store the Super Flow and use the `createSuperflow` command to create a new Super Flow.

When you create a Super Flow, you can choose to either use an existing Super Flow as a template, or you can create an empty Super Flow. Empty Super Flows will not contain any flows; however, there will

be a client and a server host definition. App Profiles that are created using a template will be a clone of the template used.

When creating the Super Flow, you can use the `-name` attribute to name it. If you do not name the Super Flow when you create it, the system will give it a default name (e.g., superflowClient0). You can additionally use the `-template` attribute to base the Super Flow on an existing Super Flow.

Once you create the Super Flow object, you can use its commands to create and set up flows and configure the host definitions. After you've created a complete Super Flow, you can add it to an App Profile.

Note: You must save the Super Flow so that it will be stored on the system for later use.

Syntax

Use the following syntax to create a Super Flow based on a template.

```
set superflowObjectName [$connectionObject createSuperflow-template
{superFlowName} -name superflowName
```

Use the following syntax to create an empty Super Flow.

```
set superflowObjectName [$connectionObject createSuperflow-name
superflowName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name ftpFlow]; #creates an empty Super Flow called
ftpFlow and a Super Flow object called 'x'
```

```
set y [$var createSuperflow -name httpFlow]; #creates an empty Super Flow
called httpFlow and a Super Flow object called 'y'
```

```
$x save; #saves httpTraffic
```

```
$y save; #saves ftpTraffic
```

Saving the Super Flow As...

Use the `configure` command and the `-name` attribute to save an existing Super Flow under a new name.

Note: The original Super Flow will still remain in the system.

Syntax

```
$superFlowObjectName configure -name
newSuperflowName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name httpProfile]; #creates an empty AppProfile
called httpProfile
```

```
$x configure -name webTraffic; renames httpProfile to webTraffic
```

```
$x save; saves the App Profile
```

Setting the Weight of a Super Flow

Use the `weightType` command to set whether the weight of a Super Flow determines its proportion in the traffic by flow count, or by bandwidth.

Syntax

Use the following syntax to set the weight of a Super Flow.

```
$a configure -weightType bandwidth,
flows
```

Example

```
set a [$bps createAppProfile]
::bps::BPSConnection::bPSConnection0::appProfileClient0
```

```
$a configure
{-name appProfileClient0 appProfileClient0} {-weightType bandwidth
bandwidth}
```

```
$a configure -weightType asdf
workingAppProfileModify:weightType:'asdf':must be one of ["bandwidth",
"flows"]
```

```
$a configure -weightType flows
```

```
$a cget -weightType
```

```
flows
```

Adding Super Flows to an App Profile

Use the `addSuperflow` command to add a Super Flow to an App Profile. When adding a Super Flow to an App Profile, you must specify the Super Flow name, weight distribution, and random seed.

Note: The weight distribution determines the frequency at which Super Flow will be selected for the application traffic. The random seed enables you to control whether static or dynamic content will be generated. Setting the random seed to '0' will generate dynamic content.

Syntax

Use the following syntax to add a Super Flow to an App Profile.

```
$appProfileObjectName addSuperflow superflowName weight seed
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name webTraffic]; #creates an empty App Profile
called webTraffic and an App Profile object called 'x'
```

```
set y [$var createSuperflow -name httpFlow]; #creates an empty Super Flow
called httpFlow and a Super Flow object called 'y'
```

```
$x addSuperflow httpFlow 10 415; #adds httpFlow with a weight of 10 and a
random seed of 415 to the webTraffic App Profile
```

Listing Super Flows in an App Profile

Use the `getSuperFlows` command (command of the App Profile object) to get a list of Super Flows that are in an App Profile. Additionally, you can retrieve the weight and seed of each Super Flow.

Syntax

Use the following syntax to add a Super Flow to an App Profile.

```
$appProfileObjectName
getSuperFlows
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createAppProfile -name webTraffic]; #creates an empty App Profile called webTraffic and an App Profile object called 'x'
```

```
set y [$var createSuperflow -name httpFlow]; #creates an empty Super Flow called httpFlow and a Super Flow object called 'y'
```

```
$x addSuperflow httpFlow 10; #adds httpFlow with a weight of 10 to the webTraffic App Profile
```

```
$x getSuperFlows; # returns a list of Super Flows and their weights and random seeds
```

Removing a Super Flow from an App Profile

Use the `removeSuperflow` command to remove a Super Flow from an App Profile.

Syntax

Use the following syntax to remove a Super Flow from an App Profile.

```
$appProfileObjectName removeSuperflow superflowName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true]; #creates the connection object
```

```
set x [$var createAppProfile -name httpTraffic; #creates an App Profile called httpTraffic
```

```
$x addSuperflow httpFlow 10; #adds a Super Flow with a weight of 10 to the App Profile
```

```
$x removeSuperFlow httpFlow; #removes the Super Flow from the App Profile
```

Deleting a Super Flow from the System

Use the `deleteSuperflow` command to delete a Super Flow from the system.

Syntax

Use the following syntax to remove a Super Flow from an App Profile.

```
$connectionObject removeSuperflow superflowName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow; #creates a Super Flow called
httpFlow
```

```
$var deleteSuperflow httpFlow; #removes the Super Flow from the system
```

Deleting the Super Flow Object

Use the `itcl::delete` command to delete the Super Flow object.

Syntax

Use the following syntax to delete the App Profile object.

```
itcl::deleteobject
$appProfileObjectName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x save; #saves the Super Flows in $x
```

```
itcl::delete object $x; #deletes the Super Flow object
```

Listing Hosts

Use the `getHosts` command to get a list of hosts that are available for a Super Flow.

Additionally, you can use the Tcl `dict` command to get more information on a particular host – including the host's interface (target or origin) and DNS name. The system will list this information using the `iface` and `dnsname` tags.

Syntax

Use the following syntax to get a list of hosts that are available in a Super Flow.

```
$superflowObjectName
getHosts
```

Use the following syntax to get more information about a particular host.

```
dict get [$superflowObjectName getHosts]
hostName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x getHosts; #returns a list of hosts that are available
```

```
Client {iface origin dnsname client%n} Server {iface target dnsname server%n};
#example of the information the system will return
```

```
dict get [$x getHosts] Client; #get more information on the Client
```

```
iface origin dnsname client%n; #example of the dns name and interface
information the system returns
```

Adding Hosts to the Origin Interface

Use the `addHost` command to add a host to the origin interface (or also known as the client interface).

Syntax

Use the following syntax to add a host to the Origin interface.

```
$superflowObjectName addHost hostNickname origin hostName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addHost {DNS Server} origin dns%n; #adds a host called DNS Server that uses
the client interface and has a dns name of dns%n to the Super Flow
```

Adding Hosts to the Target Interface

Use the `addHost` command to add a host to the target interface (or also known as the server interface).

Syntax

Use the following syntax to add a host to the Target interface.

```
$superflowObjectName addHost hostNickname target hostName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addHost {DNS Server} target dns%n; #adds a host called DNS Server that uses
the server interface and has a dns name of dns%n to the Super Flow
```

Modifying Hosts

Use the `modifyHost` command and the `-iface` and `-dnsname` attributes to modify a host's attributes.

Note: You cannot modify the host's nickname.

Syntax

Use the following syntax to modify the interface and host name.

```
$superflowObjectName modifyHost hostNickname -iface interface -dnsname
hostName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addHost {DNS Server} target dns%n; #adds a host called DNS Server that uses
the server interface and has a dns name of dns%n to the Super Flow
```

```
$x modifyHost {DNS Server} -iface origin -dnsname server%n; #changes the
interface to 'origin' and the host name to 'server%n'
```

Removing a Host from a Super Flow

Use the `removeHost` command to remove a host from a Super Flow.

Syntax

Use the following syntax to delete a host from a Super Flow.

```
$superflowObjectName removeHost
hostNickname
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addHost {DNS Server} target dns%n; #adds a host called DNS Server that uses
the server interface and has a dns name of dns%n to the Super Flow
```

```
$x removeHost {DNS Server}; deletes the DNS Server host from the Super Flow
```

Listing Protocols

Use the `listProtocols` command to see a list of all protocols that are available to use for flows.

Syntax

Use the following syntax to get a list of the protocols you can use to create flows.

```
$connectionObject
listProtocols
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var listProtocols; #returns a list of all protocols
```

Specifying an Uploaded File in the Super Flow

Use the `getActions` command to specify an uploaded file within your Super Flow.

Syntax

Use the following syntax to specify an uploaded file within your HTTP Super Flow.

```
% dict get [$s
getActions]
```

Example

```
% set s [$bps createSuperflow];# creates a Super Flow
% $s addFlow httpadv Client Server;# adds a flow
```

```
% $s getActionChoices 1;# returns a list of actions that can be used for the
specified flow
```

```
% $s addAction 1 client post_uri
% dict get [$s getActions] 1 post-data-URL
% $s getActions
% $s getActionParameters 1;# adds the action to get a list of the parameters
that it supports
```

```
% $s modifyAction 1 -post-data-URL cs-smtp-simple-message-body.txt;# adds the
path relative to the /resources directory on the machine
```

```
% dict get [$s getActions] 1 post-data-URL
cs-smtp-simple-message-body.txt
```

Note: If you do not know which values the parameter accepts, try any value. If the value you try is invalid, you will receive an error message that provides you with valid values.

Example

```
% $s modifyAction 1 -post-data-URL cs-smtp-simple-mes
invalid value "cs-smtp-simple-mes" for "post-data-URL", must be one of:
cs-smtp-simple-message-body.txt testlink.htm attachment.txt bpsh-
chassis.exe.bak URL.htm Trackweb.asp.htm Network Management Software _
Enterprise Performance NetQoS NetQoS.htm {Network Management Software _
Enterprise Network Performance _ NetQoS NetQoS.htm} it_works.html it_doesnt_
work.html page-1.html bigfile.pcap cannedapp.xml
```

Finding Flows

Use the `getFlows` command to get a list of flows that are available for a Super Flow. When you use only the `getFlows` command, and no optional attributes, the system will return the following information:

- Protocol on which the flow is based
- The direction of the flow (i.e., from the client to the server)
- Configurations for the protocol parameters

Additionally, you can use the Tcl `dict` command to retrieve the protocol on which a specific flow is based.

Syntax

Use the following syntax to get a list of flows that are available in a Super Flow.

```
$superflowObjectName
getFlows
```

Use the following syntax to return the protocol on which the flow is based.

```
dict get [$superflowObjectName getFlows] flowName protocol
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x getFlows; #returns a list of flows that are available
```

```
dict get [$x getFlows] flow1 protocol; #see what protocol flow1 uses
```

Adding Flows

Use the `addFlow` command to add a flow to a Super Flow. When you add a flow, you can specify whether the flow goes from the client to the server, or from the server to the client. Additionally, you can specify the flow's protocol parameters when you create it.

Note: When you add a flow to a Super Flow, the system will automatically name the flow for you. Each flow will be named based on the order in which it was added. For example, the first flow added to a Super Flow will be called '1'; the second flow will be called '2', and so forth.

Note: There can be up to 16 flows in a Super Flow.

Syntax

Use the following syntax to add a flow that goes from the client to the server.

```
$superflowObjectName addFlowprotocol Client Server
```

Use the following syntax to add a flow that goes from the server to the client.

```
$superflowObjectName addFlow protocol Server Client
```

Use the following syntax to add a flow that goes from the server to the client and specifies its protocol parameters.

```
$superflowObjectName addFlow protocol Server Client -protocolParameter value
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds an http-based flow
```

Listing Flow Parameters

Use the `listFlowParameters` command to get a list of configurable flow parameters for a flow. You must specify the flowID for the flow whose Flow Parameters you would like to see.

Syntax

Use the following syntax to return a list of Flow Parameters that are available for a specific flow.

```
$superflowObjectName getFlowParameters
flowID
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x getFlowParameters 1; #returns a list of Flow Parameters for flow 1
```

Removing Flows from Super Flows

Use the `removeFlow` command to remove a flow from a Super Flow.

Note:

- If you delete a flow, the other existing flows will be resequenced. For example, deleting flow 4 will resequence flow 5 to 4, and flow 6 to 5.
- Deleting a flow will remove all references to it, including all actions that are based on that flow.

Syntax

Use the following syntax to remove a flow from a Super Flow.

```
$superflowObjectName removeFlow
flow#
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called x
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x removeFlow 1; #removes flow 1 from the Super Flow
```

Listing Protocol Parameters for Flows

Use the `getFlowParameters` command to get a list of protocol parameters that are configurable for a specific flow.

Syntax

Use the following syntax to list the configurable protocol parameters for a flow.

```
$superflowObjectName getFlowParameters
flow#
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called x
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x getFlowParameters; #returns a list of protocol parameters for the http-based
flow
```

Configuring Protocol Parameters for Flows

Use the `modifyFlow` command to configure the protocol parameters for a flow. Additionally, you can use the `modifyFlow` command and the optional attributes `-to` and `-from` to modify the hosts on the target and origin interfaces (e.g., -to Client -from Server or -to Server -from Client).

Note: To see the configurable protocol parameters for a flow, use the `getFlowParameters` command.

Syntax

Use the following syntax to configure the protocol parameters for a flow.

```
$superflowObjectName modifyFlow flow# -protocolParameter
value
```

Use the following syntax to configure the protocol parameters for a flow and use the `-to` and `-from` attributes to set the direction of the flow.

```
$superflowObjectName modifyFlow flow# -to interfaceName -from intefaceName -
protocolParameter value
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called x
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x getFlowParameters; #returns a list of protocol parameters for the http-based
flow
```

```
$x modifyFlow 1 -to Client -from Server -client-profile ie7; #changes the hosts
for the flow and sets the client profile to IE 7
```

Unsetting Protocol Parameters

Use the `unsetFlowParameter` command to unset the value for a protocol parameter. When a protocol parameter is 'unset', the system will reset the protocol parameter to its default value; in some cases, the protocol parameter may have been empty. If a protocol parameter's default value is empty, the system may generate random values for the parameter.

Syntax

Use the following syntax to 'unset' or reset a protocol parameter to its default value.

```
$superflowObjectName unsetFlowParameter flow# -
protcolParameter
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called x
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x getFlowParameters; #returns a list of protocol parameters for the http-based
flow
```

```
$x modifyFlow 1 -to Client -from Server -client-profile ie7; #changes the
direction of the flow and sets the client profile to IE 7
```

```
$x unsetFlowParameter 1 -client-profile; #unsets the client profile protocol
parameter
```

Listing Actions

Use the `getActionChoices` command to return a list of actions for a specific flow. All actions are listed by flow ID and will include the source (i.e., client or server) and the action type (e.g., GET, PUT, POST, etc.).

Syntax

Use the following syntax to retrieve a list of available actions for the Super Flow.

```
$superflowObjectName getActionChoices
flow#
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpadvFlow]; #creates a Super Flow called
httpadvFlow and a Super Flow object called x
```

```
$x addFlow httpadv Server Client; #adds flow 1 to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow 1
```

Adding Actions to a Super Flow

Use the `addAction` command to add an action to a Super Flow. Each action that is added to a Super Flow will be assigned an action ID; this value is based on the order in which the action was added. For example, the first action will be action '1', the second action will be action '2', and so forth.

Note: If you remove an action from a Super Flow, then the actions will be resequenced to the action ID the preceding action ID (e.g., action '3' will become action '2').

When you add an action to the Super Flow, you will need to specify the flow ID on which the action will be based, the source of the action (i.e., client or server), the action type (e.g., get, post, put, etc.), and any action parameters that you want to configure.

Note: Any action parameters that you do not specify will use the system's default value. If the action parameter's default value is blank, then the system will generate a random value for the action parameter.

Note: Use the `getActionChoices` command to display a list of available actions for a specific flow.

Syntax

Use the following syntax to add an action to a Super Flow.

```
$superflowObjectName addAction flowID source actionType -actionParameter value
```

Example

```
set var [bps::connect 10..10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called x'
```

```
$x addFlow http Server Client; #adds flow 1 to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow 1
```

```
$x addAction 1 client get -request-size 45; #adds a get request from the client
with a request size of 45 bytes
```

Configuring Action Parameters

Use the `modifyAction` command to configure the action parameters for a specific action.

Syntax

Use the following syntax to configure action parameters for a specific action.

```
$superflowObjectName modifyAction actionID -actionParameter
value
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow
'1'
```

```
$x addAction 1 client get -request-size 45; #adds action '1', which is a get
request from the client with a request size of 45 bytes
```

```
$x modifyAction 1 -request-size 2; #changes the request size for action '1' to
2 bytes
```

Listing Action Parameters

Use the `getActionParameters` command to list the Action Parameters for each Action. You will do this by referencing the Action ID assigned to the action. Action IDs are automatically and sequentially assigned to Actions as they are added to a Super Flow.

Syntax

Use the following syntax to view a list of actions that are used in a Super Flow.

```
$superflowObjectName getActionParameters
actionID
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x addAction 1 client get; #adds the GET action to flow '1', this will be the
first action so it will have an action ID of '1'
```

```
$x getActionParameters 1; returns a list of Action Parameters for the action ID
specified
```

Listing Actions in a Super Flow

Use the `getActions` command to get a list of all actions that are in a specific Super Flow.

Syntax

Use the following syntax to view a list of actions that are used in a Super Flow.

```
$superflowObjectName
getActions
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow
'1'
```

```
$x addAction 1 client get -request-size 45; #adds a get request from the client
with a request size of 45 bytes
```

```
$x save; #saves the Super Flow
```

```
$x getActions; #returns a list of actions used by the Super Flow
```

Unsetting Action Parameters

Use the `unsetActionParameter` command to unset the value for an action parameter. When a protocol parameter is 'unset', the system will reset the action parameter to its default value; in some cases, the action parameter may have been empty. If a protocol parameter's default value is empty, the system will generate random values for the parameter.

Syntax

Use the following syntax to 'unset' or reset an action parameter to its default value.

```
$superflowObjectName unsetActionParameter action# -
actionParameter
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow
'1'
```

```
$x addAction 1 client get -request-size 45; #adds a get request from the client with a request size of 45 bytes
```

```
$x unsetActionParameter 1 -request-size; #resets the request size to its default value
```

Deleting Actions from a Super Flow

Use the `removeAction` command to remove an action from a Super Flow.

Note: If you delete an action, the remaining actions will be resequenced. For example, deleting action '4' will resequence action '5' to '4', and action '6' to '5'.

Syntax

Use the following syntax to remove an action from a Super Flow.

```
$superflowObjectName removeAction action#
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true]; #creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x getActionChoices 1; #returns a list of actions that are available for flow '1'
```

```
$x addAction 1 client get -request-size 45; #adds a get request from the client with a request size of 45 bytes
```

```
$x removeAction 1; #removes action '1' from the Super Flow
```

Adding Conditional Requests to a Super Flow

Conditional Requests enable you to set up several matches for a packet; these matches represent the expected responses (i.e., strings or patterns) from the device under test.

Note: The maximum number of matches for Conditional Request actions is 63 starting with BPS version 8.01. The maximum number of matches for versions prior to BPS 8.01 is three. The matches for Conditional Requests can be defined individually or a Multi-Match Response 200 OK action may be a more efficient method for you to define multiple match conditions.

The system will process each match listed in the Conditional Request in the order in which it is listed. Additionally, you can define one mismatch for the Conditional Request; this occurs when there is no response from the DUT.

For each match, you will need to specify the string the system should look for (e.g., 200 OK). If the string matches, then the system will respond with the Action you have specified for that string (e.g., Server: Response 200 (OK)). When specifying the Action for the match, you can configure the Action Parameters as you normally would.

Before creating a Conditional Request, please review the following restrictions and guidelines:

- There can only be one flow per Super Flow, if it uses Conditional Responses.
- You can only use simple expressions to define the match string.
- You can specify up to three matches and one mismatch for the Conditional Request.
- Conditional Requests are typically only used for string based protocols, such as HTTP, SMTP, FTP, etc.

To add a Conditional Request to a Super Flow, you will need to use the `addAction` command and use the keyword `expect` as the Action. Each match is defined as an attribute of the `addAction` command (e.g., `-match1`, `-match2`, and `-match3`, and `-nomatch.timeout` [for mismatches]).

Syntax

Use the following syntax to add a Conditional Request to a Super Flow. The value 'n' represents the amount of time that should elapse before a timeout occurs.

```
$superFlowObjectName addAction flowID source expect -match1 {matchName} -match2
{matchName} -match3 {matchName} -nomatch.timeout n
```

Example

```
set var [bps::connect 10..10. john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x addAction $flowid client expect -match1 {200 OK} -match2 {301 Moved} -match3
{404 Not} -nomatch.timeout 3; adds a conditional request that has 3 matches
with the specified names as well as a timeout of 3 seconds in cases of
mismatches
```

Adding Match Actions to a Match

After you add a Conditional Request to a Super Flow, you can begin adding Match Actions to the Conditional Request. Match Actions are the same as Actions; it is the term BreakingPoint uses to reference the Actions that are used within the Matches.

Syntax

Use the following syntax to add a Match Action to a Match.

```
$superFlowObjectName addMatchAction actionID matchID actionMatchID source
matchAction
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x addAction $flowid client expect -match1 {200 OK} -match2 {301 Moved} -match3
{404 Not} -nomatch.timeout 3; #adds a conditional request that has 3 matches
with the specified names as well as a timeout of 3 seconds in cases of
mismatches
```

```
$x addMatchAction 1 1 1 client get_uri -URL /match1.html
```

Viewing Match Action Parameters

Use the `getMatchActionParameters` command to retrieve a list of parameters that are available for a Match Action.

Syntax

Use the following syntax to view the parameters for a Match Action.

```
$superFlowObjectName getMatchActionParamters actionID matchID
matchActionID
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createSuperflow -name httpFlow]; #creates a Super Flow called
httpFlow and a Super Flow object called 'x'
```

```
$x addFlow http Server Client; #adds flow '1' to the Super Flow
```

```
$x addAction $flowid client expect -match1 {200 OK} -match2 {301 Moved} -match3 {404 Not} -nomatch.timeout 3; adds a conditional request that has 3 matches with the specified names as well as a timeout of 3 seconds in cases of mismatches
```

```
$x addMatchAction 1 1 1 client get_uri -URL /match1.html; #adds a match action to match 1
```

```
$x getMatchActionParameters 1 1 1; #returns a list of action parameters
```

Adding Goto Actions

You can use a Goto action to perform a group of actions multiple times without having to manually re-enter them multiple times.

Syntax

Use the following syntax to activate the Goto action.

Note: Before an action is referenced by a `goto -actionId`, the action must first be defined.

```
$sfSQL addMatchAction $act4 1 1 client goto -actionId 5 -loop 0
```

Example

```
set bps [bps::connect 10.10.11.202 admin admin]; #creates the connection object
```

```
set sfSQL [$bps createSuperflow -name QA_clientsim_SQL]; # defines the Super Flow
```

```
$sfSQL removeHost {Server}; # removes the old host
```

```
$sfSQL addHost {MySQL Server} target server%n; # defines the new host
```

```
set flowID_1 [$sfSQL addFlow mysql Client {MySQL Server}]; # defines the flow
```

```
$sfSQL modifyFlow $flowID_1 -client-port 0 -server-port 3306; # modifies the flow
```

```
set act1 [$sfSQL addAction $flowID_1 client login -transflag startend \ -username testuser1 -password ]; # adds actions
```

```
set act2 [$sfSQL addAction $flowID_1 client expect -match1IsRE true \
-match1 {\x07\x00\x00\x02\x00\x00\x00\x02\x00\x00\x00}]
set act3 [$sfSQL addAction $flowID_1 server delay \
-transflag continue -delay 1]
set act4 [$sfSQL addAction $flowID_1 client expect -match1IsRE true \
-match1 {[\x20-\x7f]\x05\x00\x00.\xfe\x00\x00.\x00}]
set act5 [$sfSQL addAction $flowID_1 client quit -transflag end]; # defines
conditional request using expect and regex
```

```
$sfSQL addMatchAction $act2 none $flowID_1 client goto \
-actionId 5 -loop 0; # a non-match action specified by 'none' that goes to
action 5
```

```
$sfSQL addMatchAction $act2 1 1 client use_database -database_name mysql
```

```
$sfSQL addMatchAction $act2 1 1 client query \
-transflag start -sql_statement {select * from user}
```

```
$sfSQL addMatchAction $act4 1 1 client goto -actionId 5 -loop 0
$sfSQL save -force
```

Strike Lists

The following commands are described in this section:

Listing Strike Lists

Use the `searchStrikeLists` command to display a list of all the Strike Lists that are available on the system. This command is useful if you want to see which Strike List you can modify or select for a Security test. The `searchStrikeLists` by itself will retrieve a listing of all Strike Lists; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-userid`, `-class`, `-timeunit`, `-timeinterval`, and `-limit`. The `-userid` attribute allows you to display Strike Lists created by a specific user. The `-class` attribute can either be defined as `canned`, which will return a list of all BreakingPoint-created Strike Lists, or `custom`, which will return a list of all user-created Strike Lists. Use the `-timeunit` and `-timeinterval` attributes to list Strike Lists by the date they were created. You can specify `-timeunit` as day or week, and you can specify any integer value between 1-500 for `-timeinterval`. The `-limit` attribute limits the number of results that are returned.

Syntax

Use the following syntax to view a list of available Strike Lists; this includes all canned and custom Strike Lists.

```
$connectionObject
searchStrikeLists
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var searchStrikeLists; #returns a list of all the Strike List stored on the
system
```

```
$var searchStrikeLists -userid admin; #returns a list of all Strike List
created by the admin
```

```
$var searchStrikeLists -class canned; #returns a list of all default Strike
List
```

```
$var searchStrikeLists -timeunit day -timeinterval 2; #returns a list of all
Strike List created two days ago
```

Creating a Strike List

You will need to create an object to store the Strike List and use the `createStrikeList` command to create a Strike List.

When you create a Strike List, you can choose to either use an existing Strike List as a template, or you can create an empty Strike List. A Strike List that is created using a template will be a clone of the template used.

When creating the Strike List, you can use the `-name` attribute to name it. If you do not name the Strike List when you create it, the system will give it a default name (e.g., strikeListClient0).

WARNING!: Running tests that contain malware Strikes will send potentially infectious malware through the device under test. After running tests that contain malware Strikes, the device under test should be considered an infected system and treated as such.

Note: You must save the Strike List so that it will be stored on the system for later use.

Syntax

Use the following syntax to create a Strike List based on a template.

```
set strikeListObjectName $connectionObject createStrikeList-template
{strikeListName} -name {strikeListName}]
```

Use the following syntax to create an empty Strike List.

```
set strikeListObjectName [$connectionObject createStrikeList -name
{strikeListName}]
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set x [$var createStrikeList -name zeroAttacks]; #creates an empty Strike List
called zeroAttacks and a Strike List object called 'x'
set y [$var createStrikeList -template {FTP Strikes} -name allFTP; #creates a
Strike List called allFTP using FTP Strikes as a template
$x save; #saves the Strike List for this object
$y save; #saves the Strike List for this object
```

Saving a Strike List As

Use the `configure` command and the `-name` attribute to save an existing Strike List under a new name.

Note: The original Strike List will still remain in the system.

Syntax

```
$strikeListObjectName configure -name
newStrikeListName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set x [$var createStrikeList -name zeroAttacks]; #creates an empty Strike List
called zeroAttacks
$x configure -name noAttacks; #renames zeroAttacks to noAttacks
```

```
$x save; #saves the Strike List$
```

Searching the Strike List

Use the `searchStrikes` command to display a list of all the Strikes that are available on the system. The `searchStrikes` command by itself will retrieve a listing of all Strikes; however, you can customize your query by using the following attributes.

The optional attributes you can add to your query include: `-offset`, `-limit`, and `-data`. The `-offset` attribute allows you to determine the starting point for your search; the `-limit` attribute limits the number of results that are returned; and the `-data` attribute allows you to include or not include certain strike data in your results. You can use these attributes separately, or use them together for a more refined search.

Strikes will be listed hierarchically, or based on their directory listing (e.g., /strikes/denial/misc/osx_quickdraw_getsrcbits32argb_imap4_download.xml).

The `searchStrikes` command allows you to search for strikes based on details such as protocol, strike, direction, run ID, model name, keyword, path ID, or a category ID. To narrow your search, you can enter more than one item into your search string.

The default search capability (no keywords) will search for a string anywhere in the description. For example, if you search for HTTP, you will receive results for strikes against other protocols if HTTP is anywhere in the description. If you only want strikes against the HTTP protocol, use the protocol:http search operation.

contains some of the query strings that can be used to search for specific types of strikes contained in your tests. Enter these query strings to narrow your search.

Strike List Query Strings

Query Type	Description	Query String	Example
runid	Lists strikes that were blocked, errored, or allowed in the specified test.	runid:Blocked:internal ID runid:Errored:internal ID runid:Allowed:internal ID*	runid:Blocked:684
protocol	Lists specified strikes contained in the test that include the specifed protocol.	protocol:protocol	protocol:http

Query Type	Description	Query String	Example
keyword	Lists strikes that contain the keyword you specify.	keyword:keyword	keyword:ms_2010-07
direction	Lists strikes that contain the directionality (c2s – meaning client to server, s2c – meaning server to client, etc.) that you specify.	direction:direction	direction:c2s
name	Lists strikes that contain the details that you specify.	name:name	name:ActiveX
category	Lists strikes that belong to the category that you specify.	category:category	category:Exploits: Web Application Cookie
categoryid	Lists strikes that contain the details that you specify.	categoryid:categoryid	categoryid:/strikes/exploits/ftp/ categoryid: Exploits
pathid	Lists strikes included in the path that you specify.	pathid:path	pathid:/strikes/denial/browser/aol_activex_cookie.xml
reftype	List strikes that contain the reference id number that you specify.	reftype:reference id number	reftype:BPS 2010-0001
modelname	Lists strikes associated with the test name that you specify.	modelname:test name	modelname:0-sc
* The internal ID can be found at the end of the test report URL.			

Keywords

The following list contains all of the keywords that you can use to search for Strikes.

KeyWord	Type	Description
browser	AttackTarget	Strikes targeting Web Browsers (includes Internet Explorer, Edge, Safari, Firefox, Chrome, Opera, …).
database	AttackTarget	Strikes targeting databases (includes MySQL, NoSQL, MS-SQL, Postgres, …).
document	AttackTarget	Strikes targeting all document files (pdf, office, .doc, .xlsx); does not include .zip or other binary file types.
ics	AttackTarget	Strikes targeting Industrial Controls Systems (includes SCADA, …).
iot	AttackTarget	Strikes targeting Internet of Things (IoT).
mobile	AttackTarget	Strikes targeting mobile, includes Andrid, iOS, Windows Mobile…
network_service	AttackTarget	Strikes targeting network services (like TCP, BGP, RIP, DNS, ICMP, Radius…).
operating_system	AttackTarget	Strikes targeting Operating System itself.
web_application	AttackTarget	Strikes targeting only Web applications.
authentication_bypass	AttackVector	Any kind of strikes to bypass requirement for authentication (vulnerability or logic); does not include default credentials.
backdoor	AttackVector	Kind of malware intended of accessing system; does not include default credentials or authentication bypass.
buffer_overflow	AttackVector	Strikes sending too much/large data (includes Integer Overflow/Underflow, Stack Overflow, Heap Overflow ...).
context_escape	AttackVector	Strikes bypassing security logic to execute malicious code out of the initial environment (context).
cross_site_request_ forgery	AttackVector	Strikes targeting Cross-Site Request Forgery (CSRF or Sea-Surf) attack.
cross_site_scripting	AttackVector	Strikes targeting Cross-Site Scripting (or XSS) attacks.

KeyWord	Type	Description
design_flaw	AttackVector	Strikes targeting an error into the target or a logic vulnerability.
diagnostic	AttackVector	Diagnostic tool (kind of reconnaissance).
directory_traversal	AttackVector	Strikes targeting Directory Traversal attack against Web application. Includes "path traversal" and "local file inclusion" (LFI).
exception_handling	AttackVector	Strikes leading to an exception to allow attacker to collect information (memory dump), bypass security context or crash the target. Includes Null Pointer Dereferences.
file_format_vulnerability	AttackVector	Including attacks based in malformed file (.mp3, .avi, .docx, .pdf, …) and targeting "file reader" like Adobe Reader, Word, Excel…
file_upload	AttackVector	Strikes targeting File Upload attack against Web application to upload a malicious script and then to execute it. Includes "File Upload", "File Include" and "remote file inclusion" (RFI).
format_string	AttackVector	Strikes using "format string" vulnerability to insert string which will be evaluated as a command on the targeted system.
hijack	AttackVector	Strikes hijacking a resource (includes DLL Hijack).
invalid_value	AttackVector	Strikes inserting an expected value on targeted system leading to a Denial of Service (DoS) attack.
malware	AttackVector	Malware.
memory_corruption	AttackVector	Strikes changing data or corrupting the memory (includes Memory Corruption, Use-After-Free, …).
misconfiguration	AttackVector	Strikes abusing rules of system (includes default credentials).

KeyWord	Type	Description
recon	AttackVector	Not necessarily bad in-and-of-itself; typically, a precursor to attack (information gathering).
shellcode	AttackVector	Payload to exploit vulnerability.
social_engineering	AttackVector	Strikes using "Social Engineering" to lead victim to take part of the attack by executing malicious content.
sql_injection	AttackVector	Strikes targeting SQL injection (SQLi) attack.
unspecified_vulnerability	AttackVector	Strikes for which the vulnerability is not specified on public resources (typically NVD/NIST).
worm	AttackVector	Worms (kind of malware).
denial_of_service	AttackResult	Strikes with a Denial of Service (DoS) result.
information_disclosure	AttackResult	Strikes allowing to disclose data from the target. Can be the result of memory leakage, SQL injection, file dump, …
information_gathering	AttackResult	Strikes allowing to gather information about the target (from reconnaissance, diagnostic, …).
privilege_escalation	AttackResult	Strikes allowing to gain elevated access to resources normally protected.
remote_code_execution	AttackResult	Strike running code on the targeted system. Most of the time an injected code from Cross-Site Scripting, Buffer Overflow attack, File Format Vulnerability, …
resource_exhaustion	AttackResult	Strikes targeting to exhaust resources on the targeted systems. Leading most of the time to a Denial of Service.
0_day	AttackRelevance	No patch was available at the time the strike was released.

KeyWord	Type	Description
ati_rapsheet	AttackRelevance	Exploits targeting the vulnerability identified by ATI researchers in the wild (child of in_the_wild).
attacker_tools	AttackRelevance	Exploits for this strike available as part of well-known attacker tools (as might be used by script-kiddies).
default_over_ssl	AttackRelevance	Strikes running over SSL by default.
false_positive	AttackRelevance	All strikes can send 'almost-exploit' to test overly aggressive DUT signatures or cache-based detection.
fuzzer	AttackRelevance	Strikes primarily intended to test applications and robustness.
in_body	AttackRelevance	Exploit is in the protocol body (HTTP, SIP). This does not include query (POST). Generally, it's a Server to Client attack.
in_header	AttackRelevance	Exploit is in the protocol header (HTTP, SIP). This does not include query (GET). Generally, it's a Client to Server attack.
in_query	AttackRelevance	Exploit is in the query (accounts for both GET and POST). Generally, it's a Client to Server attack.
in_the_wild	AttackRelevance	Attacks targeting vulnerability which is actively exploited in the wild.
no_nat	AttackRelevance	Strike will not run correctly when used with DNAT/SNAT Network Neighborhoods.
one_arm	AttackRelevance	Strike can exploit the vulnerability against a live system (includes former one-way strikes).
private_poc	AttackRelevance	An exploit (or Proof-of-Concept) for the targeted vulnerability is known to exist but is not generally available publicly.
public_details	AttackRelevance	Details required to construct an exploit for the vulnerability is generally available, but no PoC is known to be available.

KeyWord	Type	Description
public_poc	AttackRelevance	An exploit (or Proof-of-Concept) for the targeted vulnerability is available from public resources (Metasploit, Project Zero, Packetstorm, Exploitdb, …).
verified	AttackRelevance	Flag to identify strikes which have been verified (from ATI) to crash or execute on target system.

Syntax

Use the following syntax to view a list of all Strikes available on the system.

```
$connectionObject searchStrikes
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
$var searchStrikes; #returns a list of all the Strikes stored on the system
```

```
[$var searchStrikes -limit 10 "protocol:http"]; #returns a list of the Strikes
that target HTTP
```

```
[$var searchStrikes -limit 10 "protocol:http direction:c2s"]; #returns a list
of the clientside Strikes that target HTTP
```

```
[$var searchStrikes -limit 10 "runid:653protocol:http"]; #returns the Strikes
that are http-based for test 653. To search for particular types of strikes
within a specific test, always include the runid query in conjuction with the
other queries.
```

Adding Strikes to a Strike List

Use the `addStrike` command to add a Strike to a Strike List.

Syntax

Use the following syntax to add a Strike to an Strike List.

```
$strikeListObjectName addStrike -name fullStrikeName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createStrikeList -name Series1]; #creates a Strike List called
Series 1 and a Strike List object called 'x'
```

```
$x addStrike -name; #adds a strike to the Strike List
```

Listing Strikes in a Strike List

Use the `getStrikes` command to get a list of Strikes that are contained within a Strike List.

Syntax

Use the following syntax to list the Strikes contained within a specific Strike List.

```
$strikeListObjectName getStrikes
strikeListName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createStrikeList -name Series1]; #creates a Strike List called
Series 1 and a Strike List object called 'x'
```

```
$x addStrike /strikes/generic/tftp/tftp_octet_long_put_256.xml; #adds a strike
to the Strike List
```

```
$x getStrikes; # returns the strikes in the Strike List
```

Deleting Strikes from a Strike List

Use the `removeStrike` command to delete a Strike from a Strike List.

Syntax

Use the following syntax to remove a Strike from a Strike List.

```
$strikeListObjectName removeStrike
fullStrikeName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createStrikeList -name Series1]; #creates a Strike List called
Series 1 and a Strike List object called 'x'
```

```
$x addStrike Strikes /strikes/generic/tftp/tftp_octet_long_put_256.xml; #adds a
Strike to the Strike List
```

```
$x removeStrike /strikes/generic/tftp/tftp_octet_long_put_256.xml; # removes
the strike from the Strike List
```

Deleting the Strike List Object

Use the `itcl::delete` command to delete the Strike List object.

Syntax

Use the following syntax to delete the Strike List object.

```
itcl::delete object
$attackSeriesObjectName
```

Example

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
```

```
set x [$var createStrikeList -name Series1]; #creates a Strike List called
Series 1 and a Strike List object called 'x'
```

```
$x save
```

```
itcl::delete object $x
```

Creating a Smart Strike List

Use the `setQuery` command to create a query that the Smart Strike List will use to locate Strikes. The resulting list will be updated when ATI Updates adds new Strikes that match the query. Use the `getQuery` command to display the query being used by the Smart Strike List. Use the `setStrikesFromQuery` command to create a copy of a Smart Strike List. Lists created using the `setStrikesFromQuery` command are static and will not be updated when ATI Updates adds new Strikes that match the query.

Syntax

Use the following syntax to define the contents of a Smart Strike List.

```
$strikeListObjectName
setQuery
```

Example

```
$s setQuery "keyword:iis"; #creates a Smart Strike List using the keyword "iis"
```

```
% $s getQuery
keyword:iis
```

```
% $s getStrikes
```

```
/strikes/worms/codered_a.xml /strikes/exploits/httpd/cve_2012_0007_ms_windows_
anti-xss_library.xml ,,,
```

Syntax

Use the following syntax to create a copy of a Smart Strike List.

```
$strikeListObjectName
setStrikesFromQuery
```

Example

```
% $s setStrikesFromQuery "keyword:iis"
```

```
% $s getQuery; # while no saved query exists, the list of strikes is the same
```

```
% $s getStrikes
```

```
/strikes/worms/codered_a.xml /strikes/exploits/httpd/cve_2012_0007_ms_windows_
anti-xss_library.xml ,,,
```

Capture

The following commands are described in this section:

Exporting the Packet Buffer

The chassis object has a command called `exportPacketTrace` that enables you to export the packet buffers for the specified slot(s)/port(s).

Note: Packet buffers can only be exported for ports that you have reserved.

In order to use the `exportPacketTrace` command, you must know:

- The location to which you would like to export the packet trace
- The type of traffic you would like exported from the buffer (values can be `both`, `rx`, or `tx`)
- The slot/port numbers whose packet buffers you would like to export

By default, packet traces are exported as a Zip (.gz) file. However, you can choose to export the packet traces as a single PCAP file or a Zip (.gz) file. You can name the file by adding it as part of the file location. Once the packet buffer has been exported, you can unzip the file to see the individual packet traces.

Note: Packet buffer export operations may be lengthy and export approximately at the rate of between 1.4 MB and 2.3 MB per second.

Syntax

Use the following syntax to export the packet trace for a slot/port on the BreakingPoint Storm.

```
$chassisObject exportPacketTrace directory? options? $slot $port
$direction
```

See

Available Options for exportPacetTrace

Option	Description
`-async value`	Specified as an attribute to the `run` command. This attribute runs the test in the background, and executes the command specified.
`compress value`	Returns the data in Zipped (.gz) compressed pcap format when set to true
`-force value`	Specified as an attribute to any command that creates or modifies an object. This attribute allows you to force the system to override the existing object. If you do not specify true or false after the statement, the system will automatically assume that the value is true.
`-help`	Prints the list of commands with descriptions
`-progress value`	Specified as an attribute to the `run` command. This attribute lets you specify a Tcl script that will be called periodically while the test runs. The test name and a percentage of completion will be appended to the script you provide via the 'concat' command. The default value is the empty string, which means that no command will run to show the test progress.
`-rxfilter value`	BPF filter string to limit the data returned for received packets

Option	Description
-rxsnaplen value	Truncates received packets larger than specified length
-size value	Specifies the size value of the data to be returned
-sizetype value	Specifies the size value of the export {megabytes, frames}
-start value	Designates a starting point for the export
-startType value	Specifies the start value of the export {megabytes, packets}
-txfilter value	BPF filter string to limit the data returned for transmitted packets
-txsnaplen value	Truncates transmitted packets larger than specified length
-?	Prints the list of commands with descriptions

for available options for the exportPacketTrace command.

Example

The following example exports packet traces from slot 1/port 0, slot 1/port1, and slot 1/port 2.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
set c1 [$var getChassis]; #creates the chassis object
$c1 reservePort 1 0; #reserves port 0 on slot 1
$c1 reservePort 1 1; #reserves port 1 on slot 1
$c1 reservePort 1 2; #reserves port 2 on slot 1
$c1 reservePort 1 3; #reserves port 3 slot 1
$c1 exportPacketTrace /tmp 1 0 both; # exports the transmitted and received
traffic from the packet buffer on slot 1/port 0
$c1 exportPacketTrace /tmp 1 1 rx; # exports the received traffic from the
packet buffer on slot 1/port 1
$c1 exportPacketTrace /tmp 1 2 tx; # exports the transmitted traffic from the
packet buffer on slot 1/port 2
$cl exportPacketTrace /tmp -compress true 1 0 both; # returns the data in a
compressed pcap file on slot 1/port 0
$cl exportPacketTrace /tmp -compress false 1 0 both; # returns the data in an
uncompressed pcap file on slot 1/port 0
$cl exportPacketTrace /tmp -txsnaplen 60 -rxsnaplen 60 1 0 both; # truncates
transmitted and received packets that are larger than 60 bytes on slot 1/port 0
$cl exportPacketTrace /tmp -txfilter "host 10.1.0.254" 1 0 both; # limits the
data transmitted to packets returned from specified host on slot 1/port 0
$cl exportPacketTrace /tmp -rxfilter "host 10.1.0.254" 1 0 both; # limits the
data received to packets returned from specified host on slot 1/port 0
$cl exportPacketTrace /tmp -starttype frames -start 1000 -size 3000 1 0 both; #
specifies the point and size at which to start capture on slot 1/port 0
proc showProgress { slot interface progressPercentage } {
puts "$slot $interface $ progressPercentage"
}
$cl exportPacketTrace /tmp -progress showProgress 1 0 both; # shows progress of
the capture on slot 1/port 0
proc notifyMeWhenDone {args}{puts "I finished!!! $args" }
$cl exportPacketTrace /tmp -async notifyMeWhenDone 1 0 both; # runs a given
script when the export is complete
```

Importing PCAP Files

Use the `importPcap` command to import a PCAP file into the system. Additionally, you can use the `-force` attribute to overwrite any PCAP file with the same name.

Note: All imported PCAP files can be used with the Recreate component. To reference the PCAP file, use the `-file` parameter when creating the Recreate component (e.g., $recreateObjectName configure -file httptraffic.pcap).

Syntax

Use the following syntax to import a PCAP file from a file location.

```
$bps importPcap <filename> -file /location/name.pcap -
force
```

Use the following syntax to import a PCAP file from a URL.

```
$bps importPcap -url http://www.google.com/name.pcap
```

Use the following syntax to import a PCAP and force it to overwrite an existing PCAP with the same name. You can set the `-force` attribute to `true` to overwrite an existing file, or set it to `false` if you do not want to overwrite an existing file.

```
$bps importPcap <filename> -file /location/name.pcap -force true -progress
"bps::textprogress stdout"
```

breaks down the elements of importing a PCAP.

Importing a PCAP

Option	Description
`-file` value	References the name of the update file.
`-url` value	References the location of the update file.
`-force` false	Allows you to force the system to override the existing object. If you do not specify true or false after the statement, the system will automatically assume that the value is true.
`-bpffilter` value	BPF formatted filter to apply when importing. Only packets that match the filter will be imported into our internal file format.
`-exportsize` value	Designate a size for the export. Only packets that are under this limit will be imported into our internal file format. The export size can be limited by the number of frames or megabytes.
`-exportunit` value	The type size limit given in megabytes or frames.
`-progress` value	The script to run with progress notifications.

Example

The following example imports a file called httptraffic.pcap from the temp location.

```
set var [bps::connect 10.10.10.10 john passwd -onclose exit -shortcuts true];
#creates the connection object
$var importPcap httptraffic -file /temp/httptraffic.pcap -force; #imports
httptraffic.pcap and overwrites any file with that same name
```

Exporting Captured Packets to a File

Use the following syntax to export the packet buffer from slot 0, port 1.

```
username@storm (group:1)% pcap export
0/1
```

You can also specify a range of frames/data to export, using an `f` (to specify frames) or an `m` (to specify megabytes).

Example

To export 5Mb of data starting at the 5th Mb on slot 2, ports 0 and 1:

```
username@storm (group:1)% pcap export 2/0-1 range 5-
10m
```

To export the first 10 frames from the packet capture buffer on slot 1, port 0:

```
username@storm (group:1)% pcap export 1/0 range 10f
```

Statistics

The following commands are described in this section:

Tcl Stats Per Component

The following sections will list the stats that can be queried for each component. When you query a stat, the system will return the value stored for it. This is useful because these stats can also be used to set up pass/fail criteria for the test. See the example below.

Pass/Fail Test Criteria Example

```
#set the results for the test
set rslt1 [ss1 result]
set rslt2 [ss2 result]
#set the variables for txFrames and rxFrames
set txF1 [$rslt1 get txFrames]
set rxF1 [$rslt1 get rxFrames]
set txF2 [$rslt2 get txFrames]
set rxF2 [$rslt2 get rxFrames]
puts "\n-CHECKING TX AND RX FRAMES-"; # makes sure txFrames matches
rxFrames
set totalTxFrames [expr ($txF1+$txF2)]
set totalRxFrames [expr ($rxF1+$rxF2)]
if {$totalTxFrames == $totalRxFrames} {
puts "PASSED: TxFrames equals RxFrames"
} else {
puts "FAILED: TxFrames not equal to RxFrames"
}
```

Bit Blaster

The following table lists the Bit Blaster statistics that you can query.

Bit Blaster Statistics

Statistic	Description
avgLatency	The average latency for all frames transmitted and received by the component over the course of the test.
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
latency_total	The total latency for all frames transmitted and received by the component
result	The result of the test (i.e., passed or failed)

Statistic	Description
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxFramesCorrupt	The total numbers of frames that were received by the component; this stat only includes tracks frames that encountered a CRC error.
rxFramesDuplicate	The total number of duplicate frames
rxFramesOos	The total number of Out-of-Sequence frames received by the component
rxFramesSlowStart	The total number of slow start frames received by the component
rxFramesUnknown	The total number of frames received by the component that did not come from the system
rxFramesWrongPort	The total number of frames that were not received on the correct port
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.

Statistic	Description
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
txFramesSlowStart	The total number of slow start frames sent by the component
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Routing Robot

The following table lists the Routing Robot statistics that you can query.

Routing Robot Statistics

Statistic	Description
avgLatency	The average latency for all frames transmitted and received by the component over the course of the test
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds

Statistic	Description
latency_total	The total latency for all frames transmitted and received by the component
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxFramesCorrupt	The total numbers of frames that were received by the component; this stat only includes tracks frames that encountered a CRC error.
rxFramesDuplicate	The total number of duplicate frames
rxFramesOos	The total number of Out-of-Sequence frames received by the component
rxFramesSlowStart	The total number of slow start frames received by the component
rxFramesUnknown	The total number of frames received by the component that did not come from the system
rxFramesWrongPort	The total number of frames that were not received on the correct port
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.

Statistic	Description
txFrameData	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
txFramesSlowStart	The total number of slow start frames sent by the component
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Session Sender

The following tablelists the Session Sender statistics you can query.

Session Sender Statistics

Statistic	Description
avgLatency	The average latency for all frames transmitted and received by the component over the course of the test
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
ipRxFrameData	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (IP frames only)
ipRxFrameDataRate	The rate (Mbps) at which IP bytes are received.

Statistic	Description
ipRxFrameRate	The rate (fps) at which IP frames are received.
ipRxFrames	The total number of TCP frames received by the component
ipTxFrameData	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (IP frames only
ipTxFrameDataRate	The rate (Mbps) at which IP bytes are transmitted.
ipTxFrameRate	The rate (fps) at which IP frames are transmitted.
ipTxFrames	The total number of TCP frames received by the component
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
latency_total	The total latency for all frames transmitted and received by the component
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of received frames that were between 512 – 1023 bytes

Statistic	Description
rxFrameSize_127	The total number of received frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of received frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of received frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of received frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxMaxFrameDataRate	The maximum rate (in fps) at which frames are received by the component
rxMaxFrameRate	The maximum rate (in Mbps) at which data is received by the component
tcpAttemptRate	The maximum number of sessions that were attempted per second
tcpAttempted	The total number of TCP connections attempted by the client
tcpAvgCloseTime	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgResponseTime	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSessionDuration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.

Statistic	Description
tcpAvgSetupTime	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpClientCloseRate	The rate at which TCP connections are closed by the client
tcpClientClosed	The total number of TCP connections closed by the client
tcpClientConcurrent	The total number of TCP connections concurrently opened by the client
tcpClientEstablishRate	The rate at which TCP sessions are established by the client
tcpClientEstablished	The total number of TCP connections established by the client
tcpClientStateCloseWait	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
tcpClientStateClosing	The total number of TCP connections on the client's side that were in the CLOSING state
tcpClientStateEstablished	The total number of TCP connections on the client's side that were in the ESTABLISHED state
tcpClientStateFinWait1	The total number of TCP connections on the client's side that were in the FIN-WAIT1 stat
tcpClientStateFinWait2	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
tcpClientStateLastAck	The total number of TCP connections on the client's side that were in the LAST_ACK state
tcpClientStateListen	The total number of TCP connections on the server's side that were in the LISTEN state
tcpClientStateSynReceived	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
tcpClientStateSynSent	The total number of TCP connections on the server's side that were in the SYN-SENT state
tcpClientStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state

Statistic	Description
tcpCloseTime_10	The total number of sessions that had a close time of 0 – 10 ms
tcpCloseTime_100	The total number of sessions that had a close time of 11 – 100 ms
tcpCloseTime_1000	The total number of sessions that had a close time of 101 – 1000 ms
tcpCloseTime_10000	The total number of sessions that had a close time of 1001 – 10000 ms
tcpCloseTime_high	The total number of sessions that had a close time of more than 10000 ms
tcpCloseTime_total	The total number of sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, and more than 10000 ms
tcpFlowsConcurrent	The maximum number of TCP flows concurrently open at any given time
tcpMaxClientConcurrent	The maximum number of TCP sessions concurrently opened by the client
tcpMaxClientEstablishRate	The maximum rate at which the client establishes TCP connections
tcpMaxServerConcurrent	The maximum number of TCP sessions concurrently opened by the server
tcpResponseTime_10	The total number of sessions that had a response time of 0 – 10 ms
tcpResponseTime_100	The total number of sessions that had a response time of 11 – 100 ms
tcpResponseTime_1000	The total number of sessions that had a response time of 101 – 1000 ms
tcpResponseTime_10000	The total number of sessions that had a response time of 1001 – 10000 ms
tcpResponseTime_high	The total number of sessions that had a response time of more than 10000 ms
tcpResponseTime_total	The total number of sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, and more than 10000 ms
tcpRxFrameData	The total number of bytes received by the component
tcpRxFrameDataRate	The maximum rate (in Mbps) at which TCP data is received by the component

Statistic	Description
tcpRxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpRxFrames	The total number of TCP frames received
tcpServerCloseRate	The rate at which TCP sessions are closed by the server
tcpServerClosed	The total number of TCP sessions closed by the server
tcpServerConcurrent	The total number of TCP connections concurrently opened by the server
tcpServerEstablishRate	The rate at which TCP sessions are established by the server
tcpServerEstablished	The total number of TCP connections established by the server
tcpServerStateCloseWait	The total number of TCP connections on the server's side that were in the CLOSE-WAIT state
tcpServerStateClosing	The total number of TCP connections on the server's side that were in the CLOSING state
tcpServerStateEstablished	The total number of TCP connections on the server's side that were in the ESTABLISHED state
tcpServerStateFinWait1	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
tcpServerStateFinWait2	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
tcpServerStateLastAck	The total number of TCP connections on the server's side that were in the LAST-ACK state
tcpServerStateListen	The total number of TCP connections on the server's side that were in the LISTEN state
tcpServerStateSynReceived	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
tcpServerStateSynSent	The total number of TCP connections on the server's side that were in the SYN-SENT state
tcpServerStateTimeWait	The total number of TCP connections on the server's side that were in the TIME-WAIT state
tcpSessionDuration_10	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state

Statistic	Description
tcpSessionDuration_100	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
tcpSessionDuration_1000	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
tcpSessionDuration_10000	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
tcpSessionDuration_high	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
tcpSessionDuration_total	The total number of sessions that had session duration of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, and more than 10000 ms
tcpSetupTime_10	The number of sessions that had a duration between 0 – 10 ms to establish a connection
tcpSetupTime_100	The number of sessions that had a duration between 11 – 100 ms to establish a connection
tcpSetupTime_1000	The number of sessions that had a duration between 101 – 1000 ms to establish a connection
tcpSetupTime_10000	The number of sessions that had a duration between 1001 – 10000 ms to establish a connection
tcpSetupTime_high	The number of sessions that had a duration of more than 10000 ms to establish a connection
tcpSetupTime_total	The total number of sessions that had setup time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, and more than 10000 ms
tcpTxFrameData	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameDataRate	The rate (in Mbps) at which TCP data is transmitted by the component
	The rate (in fps) at which TCP frames are transmitted by the component
tcpTxFrames	The total number of TCP frames transmitted by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.

Statistic	Description
txFrameData	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap.
txFrameDataRate	The maximum rate at which data was transmitted (in Mbps) by the component
txFrameRate	The maximum rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that are between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
udpFlowsConcurrent	The maximum number of UDP flows that were open at any given time during the test

Application Simulator

The following tablelists the Application Simulator stats you can query.

Application Simulator Statistics

Statistic	Description
aggregateAppFlows	The total number of flows opened for all application protocols

Statistic	Description
appAttempted	The total number of application flows attempted
appAttemptedRate	The number of new application flows that are attempted by the component per second; this value accounts for all flows that have sent a Transaction Start packet.
appAvgResponseTime	The average response time for an application flow; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appFlowRate	The number of new application flows that are opened per second
appResponseTime_10	The number of transactions that had a response time that lasted between 0 – 10 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_100	The number of transactions that had a response time that lasted between 11 – 100 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_1000	The number of transactions that had a response time that lasted between 101 – 1000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_10000	The number of transactions that had a response time that lasted between 1001 – 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_high	The number of transactions that had a response time that lasted longer than 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_total	The total amount of response time across all application protocols
appRxFrameData	The aggregate total of bytes received by the component for all application protocols
appRxFrameDataRate	The rate (in Mbps) at which Layer 7 traffic is received by the component; this value only includes application traffic.

Statistic	Description
appRxFrameRate	The rate (in fps) at which frames are received by the component; this value only includes application traffic.
appRxFrames	The aggregate total of bytes received by the component for all application protocols
appSuccessful	The total number of applications flows that were completed
appSuccessfulRate	The number of application flows that are successfully established per second; this value accounts for all flows that have sent a Transaction End packet.
appTxFrameData	The aggregate total of bytes transmitted by the component for all application protocols
appTxFrameDataRate	The rate (in Mbps) at which Layer 7 traffic is transmitted by the component; this value only includes application traffic.
appTxFrameRate	The rate (in fps) at which frames are transmitted by the component; this value only includes application traffic.
appTxFrames	The aggregate total of frames transmitted by the component for all application protocols
appUnsuccessful	The total number of applications flows that did not complete
appUnsuccessfulRate	The rate at which application flows fail; this value accounts for all flows that have sent a Transaction Start packet, but no Transaction End packet.
avgLatency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
concurrentAppFlows	The maximum number of concurrent application flows reached by the system
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds

Statistic	Description
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
latency_total	The total latency for all frames transmitted and received by the component
maxAppFlowRate	The maximum rate at which application flows were opened
maxConcurrentAppFlows	The maximum number of concurrent application flows reached by the system
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component

Statistic	Description
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
tcpAttemptRate	The rate at which TCP connections are attempted by the client
tcpAttempted	The total number of TCP connections attempted by the client
tcpAvgCloseTime	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgResponseTime	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSessionDuration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSetupTime	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpClientCloseRate	The rate at which TCP sessions are closed by the client
tcpClientClosed	The total number of TCP connections closed by the client
tcpClientConcurrent	The total number of TCP connections concurrently opened by the client
tcpClientEstablishRate	The rate at which TCP sessions are established by the client
tcpClientEstablished	The total number of TCP connections established by the client

Statistic	Description
tcpClientStateCloseWait	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
tcpClientStateClosing	The total number of TCP connections on the client's side that were in the CLOSING state
tcpClientStateEstablished	The total number of TCP connections on the client's side that were in the ESTABLISHED state
tcpClientStateFinWait1	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state
tcpClientStateFinWait2	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
tcpClientStateLastAck	The total number of TCP connections on the client's side that were in the LAST_ACK state
tcpClientStateListen	The total number of TCP connections on the client's side that were in the LISTEN state
tcpClientStateSynReceived	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
tcpClientStateSynSent	The total number of TCP connections on the client's side that were in the SYN-SENT state
tcpClientStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state
tcpCloseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpFlowsConcurrent	The maximum number of TCP flows concurrently open at any given time

Statistic	Description
tcpMaxClientConcurrent	The maximum number of TCP sessions concurrently opened by the client
tcpMaxClientEstablishRate	The maximum rate at which the client establishes TCP connections
tcpMaxServerConcurrent	The maximum number of TCP sessions concurrently opened by the server
tcpResponseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpRxFrameData	The total number of bytes received by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
tcpRxFrameDataRate	The maximum rate (in Mbps) at which TCP data is received by the component
tcpRxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpRxFrames	The total number of TCP frames received by the component
tcpServerCloseRate	The rate at which TCP sessions are closed by the server
tcpServerClosed	The total number of TCP connections closed by the server
tcpServerConcurrent	The total number of TCP sessions closed by the server
tcpServerEstablishRate	The rate at which TCP sessions are established by the server
tcpServerEstablished	The total number of TCP connections established by the server
tcpServerStateCloseWait	The total number of TCP connections on the server's side that were in the CLOSE-WAIT state

Statistic	Description
tcpServerStateClosing	The total number of TCP connections on the server's side that were in the CLOSING state
tcpServerStateEstablished	The total number of TCP connections on the server's side that were in the ESTABLISHED state
tcpServerStateFinWait1	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
tcpServerStateFinWait2	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
tcpServerStateLastAck	The total number of TCP connections on the server's side that were in the LAST-ACK state
tcpServerStateListen	The total number of TCP connections on the server's side that were in the LISTEN state
tcpServerStateSynReceived	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
tcpServerStateSynSent	The total number of TCP connections on the server's side that were in the SYN-SENT state
tcpServerStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state
tcpSessionDuration_10	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state
tcpSessionDuration_100	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
tcpSessionDuration_1000	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
tcpSessionDuration_10000	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
tcpSessionDuration_high	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
tcpSetupTime_10	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
tcpSetupTime_100	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection

Statistic	Description
tcpSetupTime_1000	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
tcpSetupTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
tcpSetupTime_high	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection
tcpTxFrameData	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameDataRate	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpTxFrames	The total number of TCP frames transmitted by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component

Statistic	Description
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Security

The following tablelists the Security stats you can query.

Security Statistics

Statistic	Description
avgLatency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
category	The Strike category to which the Strike belongs
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes

Statistic	Description
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
strikeresult	The result of a Strike (i.e., blocked, passed, errored)
strike_id	A Strike's ID
strikesBlocked	The total number of Strikes blocked by the DUT
strikesErrored	The total number of Strikes that encountered an error
strikesPassed	The total number of Strikes that were not blocked by the DUT
strikesTotal	The total number of Strikes sent to the DUT
totalAllowed	The aggregate number of Strikes not blocked by the DUT; this stat is measured across all Security components in a test.
totalBlocked	The aggregate number of Strikes blocked by the DUT; this stat is measured across all Security components in a test.
totalErrored	The aggregate number of Strikes that encountered an error; this stat is measured across all Security components in a test.
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this is the aggregate total for all types of traffic – including IP, TCP, UDP, application, and non-system generated traffic.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes

Statistic	Description
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Stack Scrambler

The following tablelists the Stack Scrambler stats you can query.

Stack Scrambler Statistics

Statistic	Description
avgLatency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted.
finalPingsRecv	The total number of pings received at the end of the test
finalPingsSent	The total number of pings sent at the end of the test
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
pingsReceived	The total number of pings sent by the component
pingsSent	The total number of pings received by the component

Statistic	Description
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_ 1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_ high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this includes all packet overhead – including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_ 1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes

Statistic	Description
txFrameSize_ high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component

Client Simulator

The following tablelists the Client Simulator stats you can query.

Client Simulator Statistics

Statistic	Description
aggregateAppFlows	The total number of flows opened for all application protocols
appAttempted	The total number of application flows attempted
appAttemptedRate	The number of new application flows that are attempted by the component per second; this value accounts for all flows that have sent a Transaction Start packet.
appAvgResponseTime	The average response time for an application flow; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appFlowRate	The number of new application flows that are opened per second
appResponseTime_10	The number of transactions that had a response time that lasted between 0 – 10 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_100	The number of transactions that had a response time that lasted between 11 – 100 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_1000	The number of transactions that had a response time that lasted between 101 – 1000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_10000	The number of transactions that had a response time that lasted between 1001 – 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.

Statistic	Description
appResponseTime_high	The number of transactions that had a response time that lasted longer than 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_total	The total amount of response time across all application protocols
appRxFrameData	The aggregate total of bytes received by the component for all application protocols
appRxFrameDataRate	The rate (in Mbps) at which layer 7 traffic is received by the component; this value only includes application traffic.
appRxFrameRate	The rate (in fps) at which frames are received by the component; this value only includes application traffic.
appRxFrames	The aggregate total of bytes received by the component for all application protocols
appSuccessful	The total number of applications flows that were completed
appSuccessfulRate	The number of application flows that are successfully established per second; this value accounts for all flows that have sent a Transaction End packet.
appTxFrameData	The aggregate total of bytes transmitted by the component for all application protocols
appTxFrameDataRate	The rate (in Mbps) at which layer 7 traffic is transmitted by the component; this value only includes application traffic.
appTxFrameRate	The rate (in fps) at which frames are transmitted by the component; this value only includes application traffic.
appTxFrames	The aggregate total of frames transmitted by the component for all application protocols
appUnsuccessful	The total number of applications flows that did not complete
appUnsuccessfulRate	The rate at which application flows fail; this value accounts for all flows that have sent a Transaction Start packet, but no Transaction End packet.

Statistic	Description
avgLatency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
concurrentAppFlows	The maximum number of concurrent application flows reached by the system
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
latency_total	The total latency for all frames transmitted and received by the component
maxAppFlowRate	The maximum rate at which application flows were opened
maxConcurrentAppFlows	The maximum number of concurrent application flows reached by the system
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component

Statistic	Description
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
tcpAttemptRate	The rate at which TCP connections are attempted by the client
tcpAttempted	The total number of TCP connections attempted by the client
tcpAvgCloseTime	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgResponseTime	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSessionDuration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.

Statistic	Description
tcpAvgSetupTime	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpClientCloseRate	The rate at which TCP sessions are closed by the client
tcpClientClosed	The total number of TCP connections closed by the client
tcpClientConcurrent	The total number of TCP connections concurrently opened by the client
tcpClientEstablishRate	The rate at which TCP sessions are established by the client
tcpClientEstablished	The total number of TCP connections established by the client
tcpClientStateCloseWait	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
tcpClientStateClosing	The total number of TCP connections on the client's side that were in the CLOSING state
tcpClientStateEstablished	The total number of TCP connections on the client's side that were in the ESTABLISHED state
tcpClientStateFinWait1	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state
tcpClientStateFinWait2	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
tcpClientStateLastAck	The total number of TCP connections on the client's side that were in the LAST_ACK state
tcpClientStateListen	The total number of TCP connections on the client's side that were in the LISTEN state
tcpClientStateSynReceived	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
tcpClientStateSynSent	The total number of TCP connections on the client's side that were in the SYN-SENT state
tcpClientStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state

Statistic	Description
tcpCloseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpFlowsConcurrent	The maximum number of TCP flows concurrently open at any given time
tcpMaxClientConcurrent	The maximum number of TCP sessions concurrently opened by the client
tcpMaxClientEstablishRate	The maximum rate at which the client establishes TCP connections
tcpMaxServerConcurrent	The maximum number of TCP sessions concurrently opened by the server
tcpResponseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpRxFrameData	The total number of bytes received by the component; this includes all packet overhead — including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
tcpRxFrameDataRate	The maximum rate (in Mbps) at which TCP data is received by the component

Statistic	Description
tcpRxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpRxFrames	The total number of TCP frames received by the component
tcpServerCloseRate	The rate at which TCP sessions are closed by the server
tcpServerClosed	The total number of TCP connections closed by the server
tcpServerConcurrent	The total number of TCP sessions closed by the server
tcpServerEstablishRate	The rate at which TCP sessions are established by the server
tcpServerEstablished	The total number of TCP connections established by the server
tcpServerStateCloseWait	The total number of TCP connections on the server's side that were in the CLOSE-WAIT state
tcpServerStateClosing	The total number of TCP connections on the server's side that were in the CLOSING state
tcpServerStateEstablished	The total number of TCP connections on the server's side that were in the ESTABLISHED state
tcpServerStateFinWait1	The total number of TCP connections on the server's side that were in the FIN-WAIT1 state
tcpServerStateFinWait2	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
tcpServerStateLastAck	The total number of TCP connections on the server's side that were in the LAST-ACK state
tcpServerStateListen	The total number of TCP connections on the server's side that were in the LISTEN state
tcpServerStateSynReceived	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
tcpServerStateSynSent	The total number of TCP connections on the server's side that were in the SYN-SENT state
tcpServerStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state
tcpSessionDuration_10	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state

Statistic	Description
tcpSessionDuration_100	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
tcpSessionDuration_1000	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
tcpSessionDuration_10000	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
tcpSessionDuration_high	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
tcpSetupTime_10	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
tcpSetupTime_100	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection
tcpSetupTime_1000	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
tcpSetupTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
tcpSetupTime_high	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection
tcpTxFrameData	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameDataRate	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpTxFrames	The total number of TCP frames transmitted by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this includes all packet overhead — including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component

Statistic	Description
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Recreate

Recreate Statistics

The following tablelists the Recreate stats you can query.

Recreate Statistics

Statistic	Description
aggregateAppFlows	The total number of flows opened for all application protocols
appAttempted	The total number of application flows attempted
appAttemptedRate	The number of new application flows that are attempted by the component per second; this value accounts for all flows that have sent a Transaction Start packet.
appAvgResponseTime	The average response time for an application flow; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appFlowRate	The number of new application flows that are opened per second

Statistic	Description
appResponseTime_10	The number of transactions that had a response time that lasted between 0 – 10 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_100	The number of transactions that had a response time that lasted between 11 – 100 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_1000	The number of transactions that had a response time that lasted between 101 – 1000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_10000	The number of transactions that had a response time that lasted between 1001 – 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_high	The number of transactions that had a response time that lasted longer than 10000 ms; the response time is measured from when the Transaction Start packet is sent to when the Transaction End packet is sent.
appResponseTime_total	The total amount of response time across all application protocols
appRxFrameData	The aggregate total of bytes received by the component for all application protocols
appRxFrameDataRate	The rate (in Mbps) at which layer 7 traffic is received by the component; this value only includes application traffic.
appRxFrameRate	The rate (in fps) at which frames are received by the component; this value only includes application traffic.
appRxFrames	The aggregate total of bytes received by the component for all application protocols
appSuccessful	The total number of applications flows that were completed
appSuccessfulRate	The number of application flows that are successfully established per second; this value accounts for all flows that have sent a Transaction End packet.

Statistic	Description
appTxFrameData	The aggregate total of bytes transmitted by the component for all application protocols
appTxFrameDataRate	The rate (in Mbps) at which layer 7 traffic is transmitted by the component; this value only includes application traffic.
appTxFrameRate	The rate (in fps) at which frames are transmitted by the component; this value only includes application traffic.
appTxFrames	The aggregate total of frames transmitted by the component for all application protocols
appUnsuccessful	The total number of applications flows that did not complete
appUnsuccessfulRate	The rate at which application flows fail; this value accounts for all flows that have sent a Transaction Start packet, but no Transaction End packet.
avgLatency	The average amount of latency for all received frames. This value is computed by adding the latencies for all frames and dividing that number by the total number of frames that use the stats: 0 – 10 us latency, 11 – 100 us latency, 1001 – 10000 us latency, and over 10000 us latency.
concurrentAppFlows	The maximum number of concurrent application flows reached by the system
droppedFrames	The total number of frames received by the component but were dropped because they were malformed or misrouted
latency_10	The number of frames that had a latency between 0 – 10 microseconds
latency_100	The number of frames that had a latency between 11 – 100 microseconds
latency_1000	The number of frames that had a latency between 101 – 1000 microseconds
latency_10000	The number of frames that had a latency between 1001 – 10000 microseconds
latency_high	The number of frames that had a latency of more than 10000 microseconds
latency_total	The total latency for all frames transmitted and received by the component

Statistic	Description
maxAppFlowRate	The maximum rate at which application flows were opened
maxConcurrentAppFlows	The maximum number of concurrent application flows reached by the system
result	The result of the test (i.e., passed or failed)
rxAvgFrameSize	The average size of frames received by the component (in bytes). This value is computed by taking the receiving data rate and dividing it by the total number of received frames.
rxFrameData	The total number of bytes received by the component
rxFrameDataRate	The rate at which data was received (in Mbps) by the component
rxFrameRate	The rate at which frames were received (in fps) by the component
rxFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
rxFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
rxFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
rxFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
rxFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
rxFrames	The total number of frames received by the component
rxMaxFrameDataRate	The maximum rate (in Mbps) at which data is received by the component
rxMaxFrameRate	The maximum rate (in fps) at which frames are received by the component
tcpAttemptRate	The rate at which TCP connections are attempted by the client
tcpAttempted	The total number of TCP connections attempted by the client

Statistic	Description
tcpAvgCloseTime	The average amount of time it takes from the first FIN-ACK to the last ACK. This value is computed by taking the total amount of time it takes for all TCP sessions to close time and dividing it by the total number of TCP sessions that had a close time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgResponseTime	The average amount of time it takes to send the first SYN and to receive a SYN-ACK. This value is computed by taking the total response time for all TCP sessions and dividing it by the total number of TCP sessions that had a response time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSessionDuration	The average amount of time TCP sessions remained in the ESTABLISHED state. This value is computed by taking the total duration time for all TCP sessions and dividing it by the total number of TCP sessions that had a duration time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpAvgSetupTime	The average amount of time it takes from when the first SYN is sent to when the connection has been established. This value is computed by taking the total amount of set up time for all TCP sessions and dividing it by the total number of TCP sessions that had a set up time of 0 – 10 ms, 11 – 100 ms, 101 – 1000 ms, 1001 – 10000 ms, or more than 10000 ms.
tcpClientCloseRate	The rate at which TCP sessions are closed by the client
tcpClientClosed	The total number of TCP connections closed by the client
tcpClientConcurrent	The total number of TCP connections concurrently opened by the client
tcpClientEstablishRate	The rate at which TCP sessions are established by the client
tcpClientEstablished	The total number of TCP connections established by the client
tcpClientStateCloseWait	The total number of TCP connections on the client's side that were in the CLOSE-WAIT state
tcpClientStateClosing	The total number of TCP connections on the client's side that were in the CLOSING state
tcpClientStateEstablished	The total number of TCP connections on the client's side that were in the ESTABLISHED state

Statistic	Description
tcpClientStateFinWait1	The total number of TCP connections on the client's side that were in the FIN-WAIT1 state
tcpClientStateFinWait2	The total number of TCP connections on the client's side that were in the FIN-WAIT2 state
tcpClientStateLastAck	The total number of TCP connections on the client's side that were in the LAST_ACK state
tcpClientStateListen	The total number of TCP connections on the client's side that were in the LISTEN state
tcpClientStateSynReceived	The total number of TCP connections on the client's side that were in the SYN-RECEIVED state
tcpClientStateSynSent	The total number of TCP connections on the client's side that were in the SYN-SENT state
tcpClientStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state
tcpCloseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpCloseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpFlowsConcurrent	The maximum number of TCP flows concurrently open at any given time
tcpMaxClientConcurrent	The maximum number of TCP sessions concurrently opened by the client
tcpMaxClientEstablishRate	The maximum rate at which the client establishes TCP connections
tcpMaxServerConcurrent	The maximum number of TCP sessions concurrently opened by the server

Statistic	Description
tcpResponseTime_10	The number of TCP sessions that took between 0 – 10 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_100	The number of TCP sessions that took between 11 – 100 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_1000	The number of TCP sessions that took between 101 – 1000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to go from the first FIN-ACK to the last ACK
tcpResponseTime_high	The number of TCP sessions that took longer than 10000 ms to go from the first FIN-ACK to the last ACK
tcpRxFrameData	The total number of bytes received by the component; this includes all packet overhead — including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap. (TCP frames only)
tcpRxFrameDataRate	The maximum rate (in Mbps) at which TCP data is received by the component
tcpRxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpRxFrames	The total number of TCP frames received by the component
tcpServerCloseRate	The rate at which TCP sessions are closed by the server
tcpServerClosed	The total number of TCP connections closed by the server
tcpServerConcurrent	The total number of TCP sessions closed by the server
tcpServerEstablishRate	The rate at which TCP sessions are established by the server
tcpServerEstablished	The total number of TCP connections established by the server
tcpServerStateCloseWait	The total number of TCP connections on the server’s side that were in the CLOSE-WAIT state
tcpServerStateClosing	The total number of TCP connections on the server’s side that were in the CLOSING state
tcpServerStateEstablished	The total number of TCP connections on the server’s side that were in the ESTABLISHED state
tcpServerStateFinWait1	The total number of TCP connections on the server’s side that were in the FIN-WAIT1 state

Statistic	Description
tcpServerStateFinWait2	The total number of TCP connections on the server's side that were in the FIN-WAIT-2 state
tcpServerStateLastAck	The total number of TCP connections on the server's side that were in the LAST-ACK state
tcpServerStateListen	The total number of TCP connections on the server's side that were in the LISTEN state
tcpServerStateSynReceived	The total number of TCP connections on the server's side that were in the SYN-RECEIVED state
tcpServerStateSynSent	The total number of TCP connections on the server's side that were in the SYN-SENT state
tcpServerStateTimeWait	The total number of TCP connections on the client's side that were in the TIME-WAIT state
tcpSessionDuration_10	The number of sessions that had a duration between 0 – 10 ms in the ESTABLISHED state
tcpSessionDuration_100	The number of sessions that had a duration between 11 – 100 ms in the ESTABLISHED state
tcpSessionDuration_1000	The number of sessions that had a duration between 101 – 1000 ms in the ESTABLISHED state
tcpSessionDuration_10000	The number of sessions that had a duration between 1001 – 10000 ms in the ESTABLISHED state
tcpSessionDuration_high	The number of sessions that had a duration of more than 10000 ms in the ESTABLISHED state
tcpSetupTime_10	The number of TCP sessions that took between 0 – 10 ms to send the first SYN and establish a connection
tcpSetupTime_100	The number of TCP sessions that took between 11 – 100 ms to send the first SYN and establish a connection
tcpSetupTime_1000	The number of TCP sessions that took between 101 – 1000 ms to send the first SYN and establish a connection
tcpSetupTime_10000	The number of TCP sessions that took between 1001 – 10000 ms to send the first SYN and establish a connection
tcpSetupTime_high	The number of TCP sessions that took longer than 10000 ms to send the first SYN and establish a connection

Statistic	Description
tcpTxFrameData	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameDataRate	The rate (in Mbps) at which TCP data is transmitted by the component
tcpTxFrameRate	The rate (in fps) at which TCP frames are received by the component
tcpTxFrames	The total number of TCP frames transmitted by the component
txAvgFrameSize	The average size of frames transmitted by the component (in bytes). This value is computed by taking the transmitting data rate and dividing it by the total number of transmitted frames.
txFrameData	The total number of bytes transmitted by the component; this includes all packet overhead — including L2, L3, and L4 headers, Ethernet CRC, and inter-packet gap.
txFrameDataRate	The rate at which data was transmitted (in Mbps) by the component
txFrameRate	The rate at which frames were transmitted (in fps) by the component
txFrameSize_1023	The total number of transmitted frames that were between 512 – 1023 bytes
txFrameSize_127	The total number of transmitted frames that were between 64 – 127 bytes
txFrameSize_255	The total number of transmitted frames that were between 128 – 255 bytes
txFrameSize_511	The total number of transmitted frames that were between 256 – 511 bytes
txFrameSize_high	The total number of transmitted frames that were larger than 1024 bytes
txFrames	The total number of frames transmitted by the component
txMaxFrameDataRate	The maximum rate (in Mbps) at which data is transmitted by the component
txMaxFrameRate	The maximum rate (in fps) at which frames are transmitted by the component

Sample Script

```
set bps [bps::connect 127.0.0.1 admin admin -onclose exit]
```

```
# create a new neighborhood

set n [$bps createNeighborhood -template existingNeighborhood -name "Script
Neighborhood"]

$n setVlanEtherType 1 88a8

$n setVlanEtherType 2 9100


# create an external domain called 'webserver', with a host we're going to refer to
later

$n addDomain external webserver

$n addSubnet external webserver {

 ranges {

 {external 1.0.1.1 1.0.1.1}

 }

}


# add a domain on interface 1 that uses VLANs

$n addDomain 1 trunk

$n addSubnet 1 trunk {

 netaddr 16.0.0.0

 netmask 16

 gateway 16.0.0.1

 behind_snapt false

 ranges {

 {hosts 16.0.0.1 16.0.0.254 00:00:10:00:00:00}

 }

 innervlan 16

}

$n addSubnet 1 trunk {

 netaddr 17.0.0.0

 netmask 16

 gateway 17.0.0.1
```

```
 behind_snapt false
 ranges {
 {hosts 17.0.0.1 17.0.0.254 00:00:10:00:00:00}
 }
 innervlan 17
}
# this VLAN uses Q-in-Q routing
$n addSubnet 1 trunk {
 netaddr 18.0.0.0
 netmask 16
 gateway 18.0.0.1
 behind_snapt false
 ranges {
 {hosts 18.0.0.1 18.0.0.254 00:00:10:00:00:00}
 }
 innervlan 45
 outervlan 18
}
$n save

$bps setNeighborhood "Script Neighborhood"

bitblaster bitblaster-1
bitblaster-1 setDomain client 1 default
bitblaster-1 setDomain server 2 default
bitblaster-1 configure -advanced.ethTypeVal CAFE \
 -duration.type frames -duration.value 1000000

appsim as 3 4
stackscrambler stks 4 3
recreate p 1 2
```

```
routingrobot rr 3 4


# create a sessionsender test that will send over all the VLANs on the trunk port
connected to interface 1

sessionsender ss

ss setDomain client 1 trunk

ss setDomain server 2 default


# create a security component that will target the webserver configured earlier

security sec

sec setDomain client 3 default

sec setDomain server external webserver


$bps run -progress "bps::textprogress stdout"


# save a test report

$bps exportReport -file /tmp/report.pdf


# check something in our sessionsender's results

set r [ss result]

foreach val [$r values] {

 puts "$val: [$r get $val]"

}
```

Filtering

Filtering adds the ability to filter `getAll` results by interface, container, outer VLAN, inner VLAN, or tag. The filters can be combined to filter for a particular VLAN ID on a particular interface.

Example

```
$n getAll ip_static_hosts -interface 1
```

```
$n getAll ip_static_hosts -tags {a b}
```

```
$n getAll ip_static_hosts -inner_vlan 11
$n getAll ip_static_hosts -inner_vlan 11 -interface 1
$n getAll ip_static_hosts -outer_vlan 0
$n getAll ip_static_hosts -container {Virtual Router i1_
default}
```

Transaction Validation

If several operations are to be performed in sequence, there is a performance optimization available. All of the validation with the BPS device can be deferred until all operations are complete. For example, if you need to create 4094 VLANs, you can do it in a loop and validate the results with the server only after it is completely ready. This can significantly reduce the amount of time taken to do the overall change.

To do so, start a transaction by calling "begin" on the network object.

```
% $n
begin
```

All subsequent operations will receive only limited client-side validation from that point until the transaction is either committed or reverted.

```
$n configure -name "Any kind of config
change"
$n commit ;# the change is validated here
$n revert
```

The `commit` command will validate the change and make it permanent, while `revert` will discard the change.

All server validation that has been deferred will happen when the `commit` command is called. Therefore, any error that might have been thrown by a configuration change during the transaction will now be thrown by the `commit`. If multiple errors exist, any one of them could be thrown by the `commit`.

If any of the client-side validation throws an error, the entire transaction will automatically be reverted.

Any removals that occur during the transaction will immediately delete the associated subobject. If the transaction is later reverted, the network entity will return, but the user will need to re-obtain a new object. Any subobjects that are created as part of a transaction will be deleted automatically if the transaction is reverted.

Transactions cannot be nested.

Example

```
% $n begin
% $n begin
% $n commit ;# The entire transaction is committed here
% $n commit ;# This has no effect because the transaction is
over
```

Configuration Command Options

The table below lists the available `config` options along with their descriptions and their default values.

Configuration Command Options

Variable	Description	Default Value
`cleanup_on_load`	Automatically removes temporary items each time a connection to BreakingPoint is established.	Yes
`group_autochange`	Automatically changes the current test group number to the lowest available group number.	Yes
`pcap_dir`	Identifies the directory that packet capture files are saved to after an export.	$HOME/pcaps
`port_force_reserve`	Unreserves selected ports if they are currently being used by another user.	Yes
`port_mtu`	Specifies the default MTU that will be used when reserving ports.	9198
`report_dir`	Identifies the directory where reports will be saved after an export.	$HOME/reports
`report_format`	Identifies the default report format used for export.	.pdf
`test_dir`	Identifies the directory where tests will be saved after an export.	$HOME/tests
`test_mode`	Indicates the default mode that tests will run in.	async

Listing all Configuration Command Options

Use the following syntax to list all options available for the `config` command.

Syntax

```
username@storm (group:1)%
config
```

Example

```
username@storm (group:1)% config

VARIABLE DEF VALUE
------------------ --- --------------------
cleanup_on_load yes yes
group_autochange yes yes
pcap_dir yes /home/username/pcaps
port_force_reserve yes yes
port_mtu yes 9198
prompt_color yes bold yellow
report_dir yes /home/username/reports
report_format yes pdf
test_dir yes /home/username/tests
test_mode yes async
```

Listing Specific Configuration Options

To show the options relating to a particular `esh` command, pass a pattern as the first argument to `config`:

```
username@storm (group:1)% config ^port

VARIABLE DEF VALUE
------------------ --- --------------------
port_force_reserve yes yes
port_mtu yes 9198
```

To look for a single option:

```
username@storm (group:1)% config port_mtu

VARIABLE DEF VALUE
------------------ --- --------------------
port_mtu yes 9198
```

Setting Configuration Options

To change the value of a `config` option, add the value to the end of the command string:

```
username@storm (group:1)% config port_mtu 1500
[+] setting 'port_mtu' to '1500'
username@storm (group:1)% config port_mtu

VARIABLE DEF VALUE
---------------- --- --------------------
port_mtu no 1500
```

Resetting Configuration Options

To reset the value of a `config` option to the default value:

```
username@storm (group:1)% config port_mtu default
[+] resetting 'port_mtu' to '9198'
```

To reset all `config` options to their default values:

```
username@storm (group:1)% config all default
[+] resetting configuration to defaults
[+] using the default configuration settings
```

CHAPTER 22 System Administration

The following commands are described in this section:

Administering the System

You can use the following commands to administer the system: `getSystemType`, `getBuildId`, `getStrikepackId`, `installStrikepack`, `installUpdate`, `factoryRevert`, `previousRevert`, and `reboot`. See the following tablefor their descriptions.

Installing Firmware Updates

Use the `installUpdate` command to install firmware updates. When installing updates to the firmware, be sure to identify the slot you are updating. To install firmware updates, use the following syntax.

```
$connectionObject installUpdate -slot2 -file* /../update-123.bps
```

Note: *-file can be replaced with -url to reference a URL rather than a fie location.

Example

```
% $bps installUpdate -slot2 -file
/home/kraney/workspace/distro/packages/updates/update-66329-79167.bps; #
Updates firmware on slot 2. If no slot is specified, the system defaults to
slot 0.
```

Note: To complete the firmware update, you must click the Start Update button on the Firmware Update screen in the user interface.

Installing ATI Updates

To install ATI Updates, use the following syntax.

```
$connectionObject installStrikePack -file* /../strike-123.bps
```

Note: *-file can be replaced with -url to reference a URL rather than a fie location.

The `installStrikepack` command accepts several arguments. provides descriptions for these arguments.

installStrikePack Arguments

Argument	Description
–file	Update file
–url	Url for update file
–progress	Script to run with progress notifications <bps::textprogressstdout>
–async	Return immediately, then call the given script upon completion

Argument	Description
–slot2	Apply update to slot 2
–slot1	Apply update to slot 1
–slot0	apply update to slot 0
–help	Print this message
–?	Print this message

Detecting System Errors

Use the following callback function to detect when a system error has occurred. The `-onsystemerror` attribute enables you to get callbacks when the system has encountered an error.

```
set chassisObjectName [$connectionObjectgetChassis-onsystemerror "echo
SYSERROR"]
```

Use the `-getSystemErrorLog` attribute to obtain information on the system error.

```
global chs exitOnSysErr
set stamp [GetTimeStamp]
puts "\nSYSTEM ERROR at $stamp"
puts [$chs getSystemErrorLog]
if {[catch {$chs getDiags -file /tmp/diags_$stamp.tgz}
err]} {
puts "getDiags ERROR: $err"
}
puts "\nDiagnostic log saved /tmp/diags_$stamp.tgz"
if { $exitOnSysErr } {
puts "exit test due to system error"
exit
} ; # end if
} ; # end proc
```

Note: This information is also available from the Control Center interface.

Viewing Resource Allocation

The chassis object has a command called `getResourceAllocation` that enables you to determine which resources on the BreakingPoint Storm are available and which resources are reserved prior to creating a new test.

Syntax

Use the following syntax to view the availability of the resources of the BreakingPoint Storm.

```
$chassisObject getResourceAllocation $slotNumber -group
$groupNumber
```

Example

The following example displays the availability of the resources of the BreakingPoint Storm.

```
set c [$bps getChassis]; # takes a slot as an
argument
$c getResourceAllocation 1
0
$c reservePort 1 2
$c getResourceAllocation 1
25
$c reservePort 1 3
$c getResourceAllocation 1
50; # also takes an optional group specification
$c getResourceAllocation 1 -group 2
$c reservePort 1 0 -group 2
$c getResourceAllocation 1 -group 2
25
$c reservePort 1 1 -group 2
$c getResourceAllocation 1 -group 2
50
```

Performing a Backup

You can use the `backup` command to perform a backup of system files. You can backup files to a USB or external hard drive, or to an NFS-based network drive.

Notes on backing up:

- Backups may take a long time (sometimes more than four hours) , so plan accordingly.
- Tests cannot be running while a backup is in progress.

- During a System Backup, only files related to the core application are backed up. Middleware files are not backed up. Also, files related to the chassis operating system or other applications installed on the chassis are not backed up.
- You can backup to a USB drive (flash drive or disc drive connected over USB), or to an NFS network drive.
- When backing up to NFS, it is recommended to have at least a 1 Gbps link to the NFS Server before beginning the Backup or Restore procedures.
- PerfectStorm and PerfectStorm One only support NFS backup.
- To backup to an NFS drive, the drive must be mountable without user credentials. There is no way to supply NFS user credentials through the system.
- Ixia recommends that you back up to FAT32 or EXT3-formatted drives. You cannot backup to FATor NTFS-formatted drives.
- The backup drive must support long file names.
- The first partition on the backup drive must be one of the supported file system types (such as FAT32 or EXT3).
- If the backup process prompts you to select the partition table type and the choices are GUID, Apple, BSD, or Master Boot Record (MBR), select MBR.

Syntax

Use the following syntax to backup files to a USB or external hard drive.

```
$connectionObject backup -
useExternal
```

Syntax

Use the following syntax to backup files to an NFS-based partition at a specific IP address.

```
$connectionObject backup -nfsIP
value
```

Syntax

Use the following syntax to backup files to an NFS-based network drive.

```
$connectionObject backup -nfsPath
value
```

breaks down the elements of performing a system backup.

Backing Up System Files

Element	Description
connectionObject	The object created for the connection.
backup	The command to create a backup of the files.

Element	Description
-useExternal	Backs up files to a USB or an external hard drive.
-nfsIP	Backs up files to an NFS-based partition located at a given IP address.
-nfsPath	Backs up files to an NFS-based partition located at a given path.
value	The name of the files being backed up.

Listing Backup Files

You can use the `listBackups` command to view a list of available system backup files. You can list files located on a USB or external hard drive, or on an NFS-based network drive.

breaks down the elements of listing available system backup files.

Listing System Backup Files

Element	Description
-useExternal	Lists files located on a USB or an external hard drive.
-nfsIP	Lists files located on an NFS-based partition at a given IP address.
-nfsPath	Lists files located on an NFS-based partition at a given path.
value	The destination of the files listed.

Syntax

Use the following syntax to list available system backup files on a USB or external hard drive.

```
$connectionObject listBackups -
useExternal
```

Syntax

Use the following syntax to list available system backup files on an NFS-based partition at a specific IP address.

```
$connectionObject listBackups -nfsIP
value
```

Syntax

Use the following syntax to list available system backup files on an NFS-based network drive.

```
$connectionObject listBackups -nfsPath
value
```

Restoring Backup Files

You can use the `restorBackup` command to restore system backup files. You can restore files located on a USB or external hard drive, or on an NFS-based network drive.

Note: After calling the `restoreBackup` command, you will need to explicitly call the `reboot` command to initiate the `restore`. You will be disconnected by the `restore` once it has been initiated.

Note: After a `restore` is complete, you will need to download a new Tcl shell before continuing.

breaks down the elements of restoring system backup files.

Restoring System Backup Files

Element	Description
`-useExternal`	Restores files located on a USB or an external hard drive.
`-nfsIP`	Restores files located on an NFS-based partition at a given IP address.
`-nfsPath`	Restores files located on an NFS-based partition at a given path.
value	The destination of the files being restored.

Syntax

Use the following syntax to restore backup files located on a USB or external hard drive.

```
$connectionObject restoreBackup -
useExternal
```

Syntax

Use the following syntax to restore backup files located on an NFS-based partition at a specific IP address.

```
$connectionObject restoreBackup -nfsIP
value
```

Syntax

Use the following syntax to restore backup files located on an NFS-based network drive.

```
$connectionObject restoreBackup -nfsPath
value
```

CHAPTER 23 RESTful API

This section covers:

RESTful API Overview and Notes

This section describes the Ixia BreakingPoint Representational State Transfer (REST) API functionality.

REST API root URL

The Breaking Point Controller REST API root URL can be accessed at the same location with the Web Interface IP (System Controller IP) with the suffix: "/bps/api/v1"

https://<System Controller IP>/bps/api/v1

REST API HTTP Methods

The Breaking Point Controller REST API works by issuing HTTPS Requests to specific URIs. The following HTTPS Requests are supported.

Verb	Description
GET	Return the resource at the specified URI. Does not require a body.
POST	Replace /Modify/ Append a resource from the specified URI. Usually contains a JSON body
DELETE	Delete a resource from the specified list element URI.

REST API HTTP Authentication and Security

- All HTTP Requests must use HTTPS. HTTP is not supported.
- Except the login procedure (see Login method description below) all other request need to be executed on a authenticated session.
- On a successful login the response will include an 'apiKey' value in the json body. This value needs to added on the future requests from the same session in the header : 'X-API-KEY'
- An example can be seen in the Python Wrapper Library included in the build. To get you started with the python wrapper see the Python REST Overview and Examples on page 1303.

REST API HTTP Responses

HTTP Status Code Conventions

Status Code	Reason
200 OK	The operation was successful and the response is expected to be free of errors.
201 Created	The correct response to a POST to a list.
202 Accepted	The response to a POST to an operation that does not return its result immediately it will refer to a status link that can be queried.
204 No Content	Servers should return a 204 No Content response if no content is included in the response.
400 Bad Request	Any parameters were in an incorrect format. Encompasses parameters in the JSON body. Body must contain more information.
401 Unauthorized	Attempt to access a URI without the proper credentials.
403 Forbidden	Attempt to execute an illegal operation. The most common case would be for a PUT/PATCH/POST to a read-only element or list.
404 Not Found	The URI was not found. This should be returned for any unknown URI, including those with URI parameters corresponding to non-existent resources.
405 Method Not allowed	Attempt to use an illegal HTTP operation on a URI, e.g. POST on a non-list URI.
406 Not Acceptable	Attempt to request a content type (e.g. in the Accept header) that is not available on the server.
500 Internal Server Error	An uncaught internal exception occurred. We should strive to avoid these as the frequency of this response is inversely proportional to code quality. Body must contain more information.

REST API Examples:

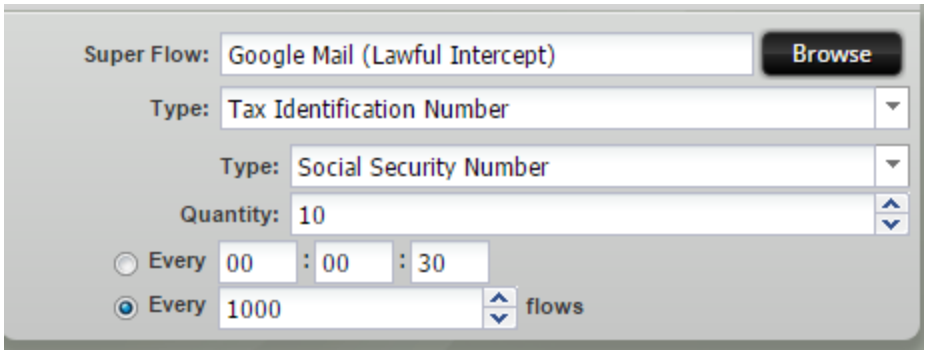
Most of the supported commands are exemplified in the Python REST Example Wrapper Description on page 1304. The methods in the examples can be adapted in other scripting languages or extended to match specific requirements. The Python libraries and a few example scripts for administering and running tests can be downloaded from the BPS system.

Note: The following notes are some of the less obvious requirements for working with Breaking Point REST API.

1. The JSON objects sent in a payload must start with lower-case letters.
2. The JSON parameters which accept Boolean values should only by passed “true” or “false” (case-sensitive).

3. If an operation fails or an error is encountered, a corresponding error code (400, 500, etc.) will be returned. Errors should be rectified and then the series of commands should be restarted in order to ensure that the final results are accurate.
4. Currently, attempts to use a non-existent PATCH parameter will not return an error code although a status code of 204 is returned.
5. When modifying an Evasion Profile, be aware that validation only occurs when you attempt to save the Evasion Profile.
6. When modifying a Super Flow name with REST, ensure to pass the Super Flow name that is displayed in the Super Flow Manager and not the name displayed in your Browse results.

For example, when you configure a Lawful Intercept Lab, the results of browsing for and selecting a Google Mail Super Flow is shown in the image below. The displayed name should NOT be passed in a REST command.



The name that should be passed in a REST command can be found in the Super Flow manager. From the Admin page select, **Managers** > **Super Flows** and then search for "Google Mail". From the list that is displayed, double-click "Google Mail (Lawful Intercept)". The name that must be used in a REST command is displayed in the title of the window that appears as shown in the image below.

Python REST Overview and Examples

Python REST Example Wrapper Description

The Python REST wrapper is a set of python libraries that can be downloaded from the BreakingPoint System which were created to help customers understand how to use the BreakingPoint REST API. The libraries contain Python functions that exercise almost all of the REST APIs. See Getting Started with Python REST Calls on the facing page for information on how to download and setup these library files.

This topic describes the Python REST wrapper methods for execution of BPS tests and applications. The methods are wrappers over the REST APIs that will be described in detail later in this section.

bpsRest.py

This example has a single class called "BPS". To initialize an object from this class the following are required:

- IP address of the BPS system
- BPS username and password login credentials

For example, bps = BPS('10.10.x.x', 'admin', 'admin')

Using the BPS object, various operations can be performed. For example, editing network settings and configurations, reserving ports, and exporting reports in PDF format, etc.

bpsAdminRest.py

This library contains 2 classes, "BPS_Storage" and "BPS_Updates". The "BPS_Updates" class can be used to update the BPS system and content. The "BPS_Storage" class can be used for storage related options (ex. Backup , Purge, etc.).

bpsVEAdminRest.py

This library contains 1 class. The "BPSVEAdmin" class can be used for BPS Virtual Edition deployments to create, add or remove slots.

Note: The Web Platform administrative REST URIs used for bpsAdminRest.py and bpsVEAdminRest.py are documented in a Services document that can be accessed using 2 different options:

1. 'https://' + self.ipstr + '/api/v1/admin/
2. **BPS Administration** >  > **REST API** > **Services**.

.

Getting Started with Python REST Calls

BPS comes with sample libraries that contain wrappers for the BPS REST API. These libraries can be used as a starting point for understanding the REST API and automating specific use cases.

Download Python libraries

The Python library files can be downloaded from the BPS system.

- **bpsRest.py**
- **bpsAdminRest.py**
- **bpsVEAdminRest.py**

To download the files, access BPS **Administration** >  > **REST API** > **BPS REST API**. Click the **REST API** link to begin the download.

Requirements for the Python wrapper libraries:

To run he REST examples you will need to install the following:

• Python 2.7

• Python module **Requests** – required to send HTTP Requests and evaluate the HTTP Responses

• Python module **JSON** - converts the JSON strings into python objects

Verify wrapper setup

You can verify the environment by trying to import the Requests and JSON libraries in Python as shown in the image below. If there are no errors, your setup is ready to run the Python Example Wrapper.

```
[root@bps-pythar-ca ~]# python
Python 2.7.5 (default, Nov  6 2016, 00:28:07)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-11)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>> import json
>>>
```

A set of example scripts are provided that demonstrate some basic BPS functionality. See Running a Basic BPS Test using REST API from Python below, Backup and Update System Example on page 1308, Restore System Example on page 1309.

Running a Basic BPS Test using REST API from Python

Unpack the archive downloaded from the BPS chassis REST API section. It consist of 2 python libraries containing wrappers for the BPS REST API.

- **bpsRest.py**
- **bpsAdminRest.py**

Three simple example Python scripts that use methods defined in the bpsRest.py and bpsAdminRest.py libraries can also be found in the downloaded archive. The "sample_script.py" script shown below, will help you setup a simple configuration.

This sample imports the BPS class from the bpsRest.py file. Calling the REST wrapper methods, this simple example:

1. Logs into the BPS system using the **admin** user account and password.
2. Checks and reserves ports 0 and 1 from slot 1.
3. Loads and executes an AppSim test configuration that already exists on the BPS system.
4. During test execution the progress is checked. When the test is finished, the overall result status of the execution is printed.
5. Logs out of the BPS system.

sample_script.py

```
from bpsRest import *

# Demo 1 Script. This script will perform the following steps:

# 1. Login
```

```
# 2. Show current port reservation state and then reserve the ports
# 3. Run a canned test - 'AppSim'
# 3.a Show progress and test statistics
# 3.b After test completion, show the test result (Pass/Fail)
# 5. Unreserve the ports
# 6. Logout
bps = BPS('10.10.x.x', 'admin', 'admin')
# login
bps.login()
# showing current port reservation state
bps.portsState()
# reserving the ports.
bps.reservePorts(slot = 1, portList = [0,1], group = 1, force = True)
# running the canned test 'DDoS Ping Flood Attack' using group 1
# please note the runid generated. It will be used for many more functionalities
runid = bps.runTest(modelname = 'AppSim', group = 1)
# showing progress and current statistics
progress = 0
while (progress < 100):
    progress = bps.getRTS(runid)
    time.sleep(1)
# showing the test result (Pass/Fail)
# a sleep is put here because we do not immediately get the test results.
# inserting a sleep to allow for the data to be stored in the database before retrieval
time.sleep(1)
bps.getTestResult(runid)
# logging out
bps.logout()
```

To run the script shown above using your BPS system:

1. Change the IP address and username and password displayed in the line, "<bps = BPS ('10.10.x.x', 'admin', 'admin')>", to values appropriate for your system.
2. Change the slot and ports in displayed the line, "< bps.reservePorts(slot = 1, portList = [0,1], group = 1, force = True) > line", to values appropriate for your system.
3. Save the file.
4. Run the command, "`python sample_script.py`".

For an example on how to update and backup the system, see Backup and Update System Example below.

Backup and Update System Example

This sample imports the methods and classes from bpsRest.py file and bpsAdminRest.py. Calling the REST wrapper methods this example does the following:

1. Logs into the BPS system using a BPS object. The user in the example is admin and password is also admin.
2. Backs up the current user data from the system.
3. Gets information on the latest available online versions of:
   a. ATI packages
   b. Monthly updates for Malware and Evergreen
   c. Daily Malware updates
4. From the list shown in step 3, installs the two most recent available online updates.

sample_bpsadmin_update.py

```
import json

import time

import os

import bpsRest

from bpsAdminRest import *

bps = bpsRest.BPS('10.10.x.x', 'admin', 'admin')

bps.login()

#optionally it is recomended to backup database and user settings before update

bpsstorage = BPS_Storrage(bps)

location = r'C:\Users\Public\Downloads'
```

```
bpsstorage.backup()
backupfile= bpsstorage.downloadBackup(location)
#####Uptdate ATI with latest 2 available missing packages ###############
#get the list of available on-line updates and update
bpswp = BPS_Updates(bps)
updateList = bpswp.getLatestAvailableUpdates()
installedpackages = bpswp.getInstalledPackages()
#create a list of possible upgrades : keys - ati-dailymalware-bps, ati-malware-bps, ati-evergreen-bps
missingupdates = createMissingUpdateList (updateList, installedpackages)
#pick the last 2 updates available from ati-dailymalware-bps
myupdatetype = "ati-dailymalware-bps"
update1 = missingupdates[myupdatetype][-1]
update2 = missingupdates[myupdatetype][-2]
print "Install ", update1, update2
mydailyupdates = [{"id": myupdatetype , "items" : [ update1, update2] }]
#install latest 2 daily ati updates
bpswp.installCloudUpdates(mydailyupdates)
#after installation is complete bps will be restarted.
print "Done"
```

To run the script shown above using your BPS system:

1. Change the IP address, username, and password displayed in the line, "<bps = BPS('10.10.x.x', 'admin', 'admin')>", to values appropriate for your system.
2. Change the desired location of the backup displayed in the line, "location = r'C:\Users\Public\Downloads' ".
3. Save the file.
4. Run the command, "`python sample_script.py`".

Restore System Example

This sample imports the methods and classes from bpsRest.py file and bpsAdminRest.py. Calling the REST wrapper methods, this example does the following:

1. Logs into the BPS system using a BPS object. The user in the example is admin and password is also admin.
2. Restores the Custom Data and Results database from a previously saved export.

sample_bpsadmin_restore.py

```
import json
import time
import os
import bpsRest
from bpsAdminRest import *
bps = bpsRest.BPS('10.10.x.x', 'admin', 'admin')
bps.login()
#backup database and user settings
bpsstorage = BPS_Storrage(bps)
backupfile = r'C:\Users\Public\Downloads\Ixia_WAF_Backup_2018_06_13_12_26_39'
print ("Attempting to restore : %s " % backupfile )
bpsstorage.restoreBackupUserData(backupfile)
print "Done"
```

To run the script shown above using your BPS system:

1. Change the IP address, username, and password displayed in the line, "<bps = BPS('10.10.x.x', 'admin', 'admin')>", to values appropriate for your system.

2. Change the desired location of the backup file displayed in the line, "backupfile = r'C:\Users\Public\Downloads\Ixia_WAF_Backup_2018_06_13_12_26_39'".

3. Save the file.

4. Run the command, `python sample_bpsadmin_restore.py`.

BPS Virtual Edition – Manage Load Modules (slots)

This sample imports the methods and classes from bpsRest.py and bpsAdminVERest.py.

The bpsAdminVERest.py can be used to create slots from scratch on EXS WMWare or KVM hypervisors and attach/detach slot operations for any kind of virtual deployment. Calling the REST wrapper methods, this example does the following:

1. Logs into the BPS system using a BPS object. The user in the example is admin and password is also admin.
2. List the hypervisor datastore and networks.
3. Set a template for deployment.
4. Create and attach one slot according to the template.

sample_bpsveadmin_addslot.py

```
import json

import time

import os

import bpsRest

from bpsVEAdminRest import *

#connect to the bps system

bps = bpsRest.BPS('BPS_IP_OR_FQDN', 'admin', 'admin')

bps.login()

#create an instance of the bpsadmin class

bpsveadmin = BPSVEAdmin(bps)

slotsEmptyStatus = bpsveadmin.getEmptySlots()

mgmtNetwork = bpsveadmin.getControllerIPSettings()

newSlotName = "VirtualBladeREST"

#hyervisor ip/hostname , credentials and type - kvm (kQEMU) or esx (kVMWare)

hypervisor = {"hostName":"Hypervisor_IP_OR_
FQDN","hostUsername":"root","hostPassword":"1x1ac0m.c0m","hostType":"kVMWare"}

#use the 1st datastore name listed by hypervisor

datastoreList = bpsveadmin.getHypervisorDatastores(hypervisor['hostName'], hypervisor
['hostUsername'],

hypervisor['hostPassword'],

hypervisor['hostType'])
```

```
datastoreName = str (datastoreList[0]['name'] )

#get the networks names listed by hypervisor

networkList = bpsveadmin.getHypervisorNetworks(hypervisor['hostName'], hypervisor
['hostUsername'],

hypervisor['hostPassword'],

hypervisor['hostType'])

#assume 1st network is management (this network assignment is subjective to the deployment)

mgmtNetworkName = networkList[0]['name']

#choose test interface NICS

testNetwork_1_Name = networkList[1]['name']

testNetwork_2_Name = networkList[2]['name']

testNetwork1_Config = {"adapter":"Network Adapter 1","network": testNetwork_1_Name}

testNetwork2_Config = {"adapter":"Network Adapter 2","network": testNetwork_2_Name}

#select the number of VMS/slots to be deployed with this template.

slotCount = 1

#configure the slot management static ip address. Use an empty array for DHCP or static address as in
the example below

staticMgmIpSettings = []

#staticMgmIpSettings =
[{"ipAddress":"10.12.12.23","mask":"255.255.252.0","gateway":"10.12.12.1","identifier":"mgmtNic_
%s" % newSlotName,"networkConfigurationType":"kStatic"}]

#the setting for the new slot

newSlotVMSettings = {"hostInfo": hypervisor ,"defaultVmName": newSlotName,"vmNo":
slotCount,"datastore": datastoreName,"mngmNetwork":mgmtNetworkName,"testNetworks":
[testNetwork1_Config, testNetwork2_Config],"ipConfigs": staticMgmIpSettings}

#create deployment template the slot with the settings configured above

if bpsveadmin.prepareSlotVMDeployment(newSlotVMSettings):

    #start deployment operations

    bpsveadmin.createVMSlots()
```

To run the script shown above using your BPS system:

1. Change the IP address, username, and password displayed in the line, "<bps = BPS(' BPS_IP_ OR_FQDN', 'admin', 'admin')>", to values appropriate for your system.

2. Change the hypervisor settings in hypervisor = {….
3. Make sure the datastore and networks are in accordance with your deployment. You can also edit them if necessary.
4. Save the file.
5. Run the command, `python sample_bpsveadmin_addslot.py`.

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Login

Used to log in into the system before any REST APIs can be implemented.

URL	https://<System Controller IP>/api/v1/auth/session	
METHOD	POST	
Request		
Content Type	Application/json	
Payload Data	username	String
	password	String
Response		
Status	200 OK	
Content Type	application/json	
Sample Request Payload	{"username":"bps", "password":"bps"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Logout

Used to log out of the system.

URL	https://<System Controller IP>/api/v1/auth/session	
METHOD	DELETE	
Request		

Content Type	None	
Request Payload	None	
Response		
Status	204 OK	
Content Type	None	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Viewing Port Status

Used to view the current port status of the system.

Tip: You are strongly advised to view the port status before reserving or unreserving ports.

This GET will fetch a large amount of data including the links of operations supported and the current port reservation state as a string "portReservationState".

URL	https://<System Controller IP>/api/v1/bps/ports	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	portReservationState	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Reserving Ports

Used to reserve ports.

Mandatory input parameters are <slot number> and <port number>. Optional boolean parameter "force" has default value as false.

Note: Passing the "force" as True will allow reserving ports that were previously reserved by other users.

URL	https://<System Controller IP>/api/v1/bps/ports/operations/reserve	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	slot	Integer
	portList	List of Integers
	group	Integer
	force	Boolean
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"slot":"1", "portList":[0,1], "group":"1", "force":"true"}	Forcefully reserve
	{"slot":"1", "portList":[0,1], "group":"1", "force":"false"} [or] {"slot":"1", "portList":[0,1], "group":"1"}	Do not reserve forcefully

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Unreserving Ports

Used to unreserve ports.

The slot and the list of ports to unreserve must be passed.

Note: This command will always unreserve ports regardless as to which user or group has reserved it.

URL	https://<System Controller IP>/api/v1/bps/ports/operations/unreserve	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	slot	Integer
	portList	List of Integers
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"slot":"1", "portList":[0,1]}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Execute a Test

This command will start a test.

- **modelname** – This is the name of the test to run
- **group** – The test to run in the group
- **neighborhood** – The neighborhood to use for running this test instance. This parameter is optional, when not given, the neighborhood already associated with the test will be used.

The test id string (which denotes the ID with which the test is started and running) will be returned for usage in other operations such as getRTS, stop and result.

Note: Passing a different "neighborhood" does not mean that the saved test model will be saved under the new neighborhood name. It will still be saved with the old neighborhood name. If the user wishes to change the neighborhood to be used permanently, then please consider modifying the testmodel using "`/api/v1/bps/workingmodel/`" (details below).

URL	https://<System Controller IP>/api/v1/bps/tests/operations/start	
METHOD	POST	

Request		
Content Type	application/json	
Request Payload	modelname	String
	group	Integer
	neighborhood	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	testid	String
	iteration	Integer
Sample Request Payload	{"modelname":"AppSim", "group":"1", "neighborhood":"BreakingPoint Routing"}	Using a different NN
	{"modelname":"AppSim", "group":"1”}	Using default NN saved in test model

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Stop a Running Test

Used to stop a running test.

The testid returned in the start operation needs to be given.

URL	https://<System Controller IP>/api/v1/bps/tests/operations/stop	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	testid	String
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"testid":"TEST-44"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Get RTS

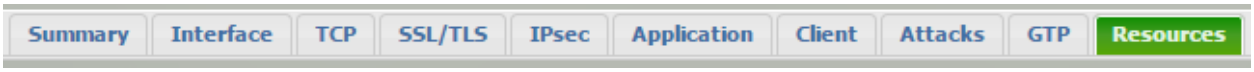
This command will return the real time statistics. The testid returned in the start operation needs to be passed as "runid".

Note: The getRTS method is mostly used for printing the real time statistics. In order to get the response as a JSON string, see the getRealTimeStatistics method.

Note:

- The command will only give RTS at the particular instant the query is fired.
- The "**statsGroup**" parameter is optional. If it is not passed, then by default, only the stats from the "**Summary**" RTS group will be displayed. If passed, then only the stats from that particular group will be displayed.
- In order to add fetch all the stats at once, instead of fetching it statsGroup wise, pass in '**all**' as the value for statsGroup.
- Please note that in this case where the statsGroup is set to '**all**', the statistics belonging to the **Application** group are aggregated and displayed.

The available RTS statsGroup parameters (corresponding to the RTS options in the UI) are shown in the following image and table.



RTS Group Shown in UI	statsGroup Parameter to be Passed
Summary	summary
Interface	iface
TCP	l4stats
SSL/TLS	sslStats
IPsec	ipsecStats

RTS Group Shown in UI	statsGroup Parameter to be Passed
Application	l7Stats
Client	clientStats
Attacks	attackStats
GTP	gtp
Resources	resource

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getrts	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	runid	String
	statsGroup	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	rts	String
Sample Request Payload	{"runid":"TEST-44", "statsGroup":"sslStats"}	

Note: The ability to request a single real time statistic value instead of a group of statistics (shown below) is possible due to a Python API enhancement.

Python API defintion	def getRealTimeStatByName(self, runid, statname, enableRequestPrints = False)	
Parameters	runid	Integer
	statname	String

	enableRequestPrints	Boolean; Optional
Response	result	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Get Real Time Statistics

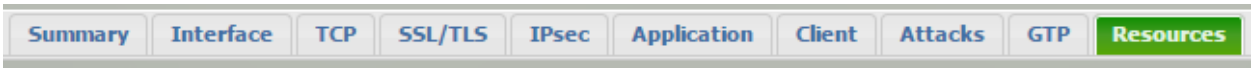
This command will return the real time statistics. The testid returned in the start operation needs to be passed as "runid".

Note: The getRealTimeStatistics method returns a JSON formatted string.

Note:

- The command will only give RTS at the particular instant the query is fired.
- The "**statsGroup**" parameter is optional. If it is not passed, then by default, only the stats from the "**Summary**" RTS group will be displayed. If passed, then only the stats from that particular group will be displayed.
- In order to add fetch all the stats at once, instead of fetching it statsGroup wise, pass in '**all**' as the value for statsGroup.
- Please note that in this case where the statsGroup is set to '**all**', the statistics belonging to the **Application** group are aggregated and displayed.

The available RTS statsGroup parameters (corresponding to the RTS options in the UI) are shown in the following image and table.



RTS Group Shown in UI	statsGroup Parameter to be Passed
Summary	summary
Interface	iface
TCP	l4stats
SSL/TLS	sslStats
IPsec	ipsecStats
Application	l7Stats
Client	clientStats

RTS Group Shown in UI	statsGroup Parameter to be Passed
Attacks	attackStats
GTP	gtp
Resources	resource

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getRealTimeStatistics	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	runid	String
	statsGroup	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	rts	String
Sample Request Payload	{"runid":"TEST-44", “statsGroup”:”sslStats”}	

Note: The ability to request a single real time statistic value instead of a group of statistics (shown below) is possible due to a Python API enhancement.

Python API defintion	def getRealTimeStatByName(self, runid, statname, enableRequestPrints = False)	
Parameters	runid	Integer
	statname	String
	enableRequestPrints	Boolean; Optional

Response	result	String

Sample usage	from bpsRest import *
	import time
	import os.path
	import sys
	import json
	force = True
	slot = 2
	portList = [0,1]
	group = 1
	testName = "testAppSim"
	statname1 = "ethRxFrames"
	statname2 = "ethTxFrames"
	BPSProxy = BPS("<System Controller IP>","admin","admin")
	BPSProxy.login()
	
	BPSProxy.reservePorts(slot,portList,group,force)
	
	runId = BPSProxy.runTest(testName,group)
	if runId == -1:
	raise AssertionError("Failed to run test %s" %testName)
	
	progress = 0
	print "Watching stats: ethRxFrames, ethTxFrames"
	while progress != 100:
	print progress
	if progress != 0:
	stats = {}
	stats[statname1] = BPSProxy.getRealTimeStatByName (runId, statname1)
	stats[statname2] = BPSProxy.getRealTimeStatByName (runId, statname2)
	print stats
	time.sleep(5)
	progress = BPSProxy.getTestProgress(runId)
	
	BPSProxy.getTestResult(runId)
	
	BPSProxy.unreservePorts(slot,portList)
	BPSProxy.logout()
		Forcefully reserve

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Delete All Test Reports On the System

This command will delete all of the test reports that exist on the system.

URL	https://<ip_ address>/api/v1/bps/tests/operations/removeAllTestReports	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload		
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Obtaining the Results of a Test

This command will be used to obtain the final result of a test (passed/failed canceled). The testid returned in the start operation needs to be passed as "runid".

URL	https://<System Controller IP>/api/v1/bps/tests/operations/result	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	runid	String
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"runid":"TEST-44"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Getting Information about Currently Running Tests

This GET command will fetch a lot of data, including the links of operations supported and the current port reservation state as a string "runningTestInfo".

URL	https://<ip_address>/api/v1/bps/tests	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response payload	runningTestInfo	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Obtaining Test Failure Information

Obtain the reason why a test run failed. The testid returned in the start operation needs to be passed as "runid".

Example: Consider that your test with runid = TEST-44 failed, so pass in a runid as "TEST-44".

URL	https://<System Controller IP>/api/v1/bps/tests/operations/failuredescription	
METHOD	POST	

Request		
Content Type	application/json	
Request Payload	runid	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"runid":"TEST-44"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Network Neighborhood Modification Overview

This topic describes the steps used to modify a Network Neighborhood.

In order to modify the network neighborhood, three steps need to be followed.

1. **Retrieve the NN**. This will load the NN and set it for modification. All the supported parameters will be displayed to the user through the GET command.
2. **Modify the desired element.** Each component is assigned an ID and the parameters for that particular ID will be displayed through the GET command. You will need to pass three things through a POST command:
   a. The component ID
   b. The element ID which has to be modified
   c. The new value which has to be set
3. **Save the neighborhood**. Once the new value is set, the user will have to save the neighborhood.

The following methods are described in detail in this section.

Retrieving the Neighborhood

URL	https://<ip_address>/api/v1/bps/network/operations/retrieve	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	name	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"name":"BreakingPoint Switching"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Viewing the Neighborhood

URL	https://<ip_address>/api/v1/bps/network/	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload		

	The neighborhood elements are shown with all the parameters	Object

Setting and viewing the current working Network Neighborhood model

Set one of the NNs (Network Neighborhoods) available on the system as the current NN

HTTP POST to https://<System Controller IP> /api/v1/bps/network/operations/retrieve" the body: {"name": "ASAv10_ESXi_m4_perf_4int_nh"}

>It will return: a 200 OK response with the information JSON body : {"result": "ASAv10_ESXi_m4_perf_4int_nh successfully retrieved for viewing and modification."}

View the current Working NN

You can get the current working network neighborhood JSON model as shown in the following example:

HTTP GET to https://<System Controller IP> /api/v1/bps/network"

>It will return: a 200 OK response with the information json body :of the network neighborhood elements.

For a python example on how to use these API's see: "retrieveNetwork" and "viewNetwork" python methods from the included bpsRest.py library.

Modifying the Neighborhood Parameters

URL	https://<ip_address>/api/v1/bps/network/operations/modify	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	componentId	String
	elementId	String
	value	String
Response		
Status	200 OK	

Content Type	application/json	
Response Payload	result	String

For example, consider a sample neighborhood below:

```
"ip4StaticHostsParams" : {
  "id:Static Hosts i20_default" : "[id:Static Hosts i20_default, tags: c1, ip_address:20.0.0.4,
  "id:Static Hosts i10_default" : "[id:Static Hosts i10_default, tags: s1, ip_address:20.0.0.10,
},
```

The value next to "id" is your componentId, the other parameters are the elements with the corresponding elementId and value displayed.

When you want to modify a parameter, pass in the "componentId", the "elementId" and the new "value".

Assume you want to change the highlighted element. The payload will look like:

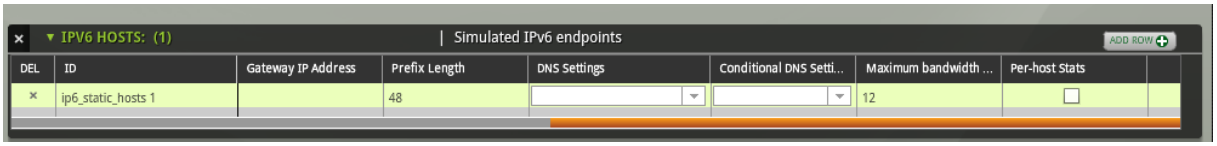
{"componentId":"Static Hosts i20_default", "elementId":"ip_address ", "value":"20.0.0.4"}

Note: Of all the elements displayed to the user, only the following elements are allowed to be modified:

ip_address	count	behind_snapt
mac_address	duplicate_mac_address	mtu
vlan_key	tpid	default_container
inner_vlan	outer_vlan	netmask
gateway_ip_address	maxmbps_per_host	enable_stats
accept_local_offers_only	allocation_rate	lease_address
lease_time	default_lease_time	offer_lifetime
max_lease_time	ia_type	prefix_length
pool_base_address	pool_size	pool_prefix_length
accept_local_requests_only	ignore_pause_frames	

If your NN holds empty values for these parameters specifically, a GET command will return a null value (example below). Other elements, which are not a part of the above table, cannot be shown in the GET response if holding empty values.

For example, in the element below, note that the "Gateway IP Address" holds no value.



Its corresponding GET response is shown below.

```
- ip6Hosts: {
    id:ip6_static_hosts 1: "[host_ipv6_addr_alloc_mode:Static, id:ip6_static_hosts 1, enable_stats:false, default_container:Interface 1, ip_address:FE80:0000:0000:0000:0202:B3FF:FE1E:8329, prefix_length:48, count:2,
    maxmbps_per_host:12, gateway_ip_address:null]"
```

Note that unsupported elements holding empty values (like DNS Settings) are not shown as part of the GET response.

If you wish to set any of the supported element to null, then simply pass "null" as the "value". Again, a user can only do this for the above supported elements.

The following components can be modified:

IP Infrastructure	Endpoint
• Interface • VLAN • IPv4 DHCP server • IPv6 DHCP server • IPv4 Router • IPv6 Router	• IPv4 External Hosts • IPv6 External Hosts • IPv4 Static Hosts • IPv6 Hosts • IPv4 DHCP Hosts

If your network neighborhood contains "Impairments" and "Packet Filtering", then it will not be displayed through the GET command, nor will you be able to modify it. In order to view or modify this type of network neighborhood, you must use the BPS UI.

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Set Multiple Network Neighborhood Elements

URL	https://<System Controller IP>/api/v1/bps/network/operations/modifybatch	
Method	POST	
Request		
Content Type	application/json	
Request Payload	networkArgs	Array of

	componentId	String
	elementId	String
	value	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String

Sample Request Payload	{
	"networkArgs": [{
	"componentId": "Static Hosts i1_default",
	"elementId": "ip_address",
	"value": "100.0.0.1"
	}, {
	"componentId": "Static Hosts i1_default",
	"elementId": "gateway_ip_address",
	"value": "100.0.0.1"
	}
	]
	}
		Modify Batch

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Saving the Neighborhood

There are two ways to save a neighborhood – with the same name using the "save" operation and with a different name using the "saveas" operation.

Using the Save operation

URL	https://<ip_address>/api/v1/bps/network/ operations/save	
METHOD	POST	
Request		
Content Type	None	
Request Payload	None	

Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String

This operation will always save the network neighborhood with the same name that was already present in the system.

Note: You cannot save a canned neighborhood. In order to modify a canned neighborhood, first modify a canned neighborhood, then save it with a different name using the "saveas" operation.

Using the Saveas operation

URL	https://<ip_address>/api/v1/bps/network/ operations/save	
METHOD	POST	
Request Payload	application/json	
	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String
Example payload	{"name":"new network"} [or] {"name":"new network", "force":"false"}	Save the neighborhood with a new name, but do not overwrite if a neighborhood with same name exists
	{"name":"asd", "force":"true"}	Save the neighborhood with a new name, and overwrite if a neighborhood with same name exists

This operation will save the currently retrieved neighborhood with a different name. If you do not pass a new "name", it will save with the same name as that already set for the network.

By default, if a network neighborhood already exists (and is not a canned network neighborhood) with the same name as passed through "name", an error will be displayed informing the user that a network already exists with the same name. In such a scenario the user should either:

- Pass in a different "name"
- Supply the "force" parameter with value "true"

On successful save operation, the "name" of the neighborhood is returned to the user as "result".

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Modification of Saved Tests

(Normal tests with various components.)

There are basically two different types of tests:

- Lab tests
- Normal tests which have different components

Normal Tests

1. Before modifying a saved test, the test has to be set as the current working model.
2. Then parameter values that need to be changed must be modified.
3. Finally, the modified test must be saved before execution.

Note: (Restrictions / Known Issues)

- A canned test cannot be modified. The user must save the canned test with a different name and only then proceed with its component modification.
- Make sure the test model is present on the system before it is set as the working model. If not, then either create a test model with the same name, or upload it.
- These operations only support PATCH and do not support PUT operations.

Setting the current working model

Request payload contains the name of the test model (template) to be set as the working model.

URL	https://<ip_address>/api/v1/bps/workingmodel	
METHOD	PATCH	
Request		
Content Type	application/json	
Request Payload	template	String

Response		
Status	204 OK	
Content Type	None	
Response payload	None	
Sample Request Payload	{"template":"AppSim"}	

Viewing the parameters of the working model

URL	https://<ip_address>/bps/api/v1/bps/workingmodel/settings	
METHOD	GET	
Response	Status – 200 (Ok)	
Response payload	application/json	
	settings	Object

This command will display the different components present in the model. The different components will be indicated through its "componentId". Inside each component, the different elements present will be indicated by its "elementId". There are two different types of elements:

1. Without internal parameters
   Will be visible with just its "elementId" and the "value"
2. With internal parameters
   The element contains different parameters, each parameter having a "paramId" and "value".


```
{
 - appsim1: {
      app: "{Params: [streamsPerSuperflow=8, removedns=false, fidelity=normal, replace_streams=true]}",
      sessions: "{Params: [max=50000, maxPerSecond=125000, target=1, targetPerSecond=1, closeFast=false, maxActive=100000, openFast=false, engine=advanced, emphasis=balanced, allocationOverride=auto, statDetail=maximum]}",
      srcPortDist: "{Params: [type=random, min=1024, max=65535]}",
      ip6: "{Params: [hop_limit=64, traffic_class=0, flowlabel=0]}",
      rampUpProfile: "{Params: [type=calculated]}",
      componentType: "appsim",
      ip: "{Params: [ttl=32, tos=0]}",
      rampDist: "{Params: [up=00:00:01, steady=00:00:28, down=00:00:15, upBehavior=full, steadyBehavior=cycle, downBehavior=full, synRetryMode=obey_retry]}",
      delayStart: "00:00:00",
      active: "true",
      tcp: "{Params: [mss=1460, retry_quantum_ms=250, retries=3, delay_acks=false, initial_receive_window=5792, add_timestamps=true, aging_time=0, reset_at_end=false, handshake_data=false,
      aging_time_data_type=aging_time_data_type_sc, tcp_window_scale=0, shutdown_data=false, initial_congestion_window=4, ecn=support_ecn, raw_flags=-1, tcp_connect_delay_ms=0, tcp_keepalive_timer=0]}",
      rateDist: "{Params: [scope=aggregate, unit=mbps, type=constant, min=1000, unlimited=false, max=10000]}",
      profile: "BreakingPoint Enterprise"
   },
   neighborhood: "BreakingPoint Switching"
}
```

In this example, "appsim1" is the componentId, "sessions" is the "elementId" of an element with internal parameters present and "delayStart" is the "elementId" of an element with no internal parameters with "value" of "00:00:00".

For "sessions", an example of internal parameter is "target" with "paramId" and "value" of "1".

Note that in the example above, a component which is enabled has the attribute “active” set to “true”. If this component is not active, then no parameters are displayed to the user except for the type of component and its active state. In order to view and modify its settings, the user must set active to “true”.


```
{
  - appsim1: {
        active: "false",
        componentType: "appsim"
    },
    neighborhood: "BreakingPoint Switching"
}
```


Modifying the parameters of the working model

In order to modify a parameter of a working model, the user has to pass the "componentId" of the component which has to be modified, the "elementId" of the element which has to be modified, the "paramId" of the parameter (if the element has Params within) and the new "value" which has to be set as a json object recognized with label "newParams".

For example, for the above working model, if you want to modify "delayStart". The passed json object will be:

{“newParams” : {“componentId”:”appsim1”, “elementId”:”delayStart”, “value”:”00:02:00”}}

and if you want to modify the "max" value of "sessions" then:

{“newParams” : {”componentId”:”appsim1”, “elementId”:”sessions”, “paramId”:”max”, “value”:”1500”}}

URL	https://<ip_address>/bps/api/v1/bps/workingmodel/	
METHOD	PATCH	
Request Payload	application/json	
	newParams	Object
Response	Status – 204 (Ok)	
Response payload		

Important parameter: If you wish to change the timeline of a component, change the parameters of the "rampDist" element.

Note: In order to change the neighborhood being used, only pass in "neighborhood" as your "elementId" and the new "value" as part of newParams.

Saving the working model

There are two methods for saving a model (similar to saving a neighborhood):

1. Perform a Save operation – Does not take any input, replaces a non-canned test with the same name with which it was already present.
2. Perform a Saveas operation – Takes as input the new "name" to use to save the working model. By default, a model if already present will not be overwritten. In such an event, a parameter "force" with value "true" will have to be passed.

Once a working model is saved, the name given to the model is sent back as a response to the user.

URL	https://<ip_address>/bps/api/v1/bps/workingmodel/operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

URL	https://<ip_address>/bps/api/v1/bps/workingmodel/operations/saveas	
METHOD	POST	
Request Payload	application/json	
	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Lab Tests

Currently the following lab components are supported:

- Session Lab
- Multicast Lab
- Lawful Intercept Lab
- RFC2544 Lab
- LTE Lab

The usage of REST to modify all the labs is similar with the only difference being in the type of type of parameters taken and the URIs.

The modification follows a similar flow for modification as the modification of test components:

1. Before modifying a saved test, the test has to be set as the current working model.
2. Then parameter values that need to be changed must be modified.
3. Finally, the modified test must be saved before execution.

Known Issues/Limitations: In regard to RFC2544 tests, a canned RFC model must be saved using a different name at least once from the UI before proceeding with modification using REST.

Session Lab Test

Setting a Lab test as working model for modification

URL	https://<ip_address>/api/v1/bps/sessionlabtest	
METHOD	PATCH	
Request		
Content Type	application/json	
Request Payload	template	String
Response		
Status	204 OK	
Content Type	None	
Response Payload	None	

Viewing the parameters of the working model

URL	https://<ip_address>/api/v1/bps/sessionlabtest/	
METHOD	GET	

Response	Status – 200 (Ok)	
Response payload	application/json	
	sessionParams	Object

Modifying the parameters of the working model

Similar to the modification of parameters of a working model with components, but does not require a "componentId" and a "paramId". Only the "elementId" and "value" has to be passed as 'sessionlabParams".

URL	https://<ip_address>/api/v1/bps/sessionlabtest/	
METHOD	PATCH	
Request Payload	application/json	
	sessionlabParams	Object
Response	Status – 204 (Ok)	
Response payload		

Saving the working model

Usage is same as that of how a normal test model is saved. The only difference is the the URL is used.

URL	https://<ip_address>/api/v1/bps/ sessionlabtest /operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

URL	https://<ip_address>/api/v1/bps/ sessionlabtest /operations/saveas	
METHOD	POST	
Request Payload	application/json	
	name	String

	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Multicast Lab Test

Setting a Lab test as working model for modification

URL	https://<ip_address>/api/v1/bps/multicasttest/	
METHOD	PATCH	
Request		
Content Type	application/json	
Request Payload	template	String
Response		
Status	204 OK	
Content Type	None	
Response payload	None	

Viewing the parameters of the working model

URL	https://<ip_address>/api/v1/bps/multicasttest/	
METHOD	GET	
Response	Status – 200 (Ok)	
Response payload	application/json	
	The parameters of the multicast test, distributed into test parameters, source and subscribers	Object

See the following example:

```
    id: 1,
  - multicastParams: {
        duration: "00:02:00",
        network: "medium"
    },
  - sourceParams: {
      - source_0: {
            ipAddress: "10.1.1.2",
            multicastAddress: "239.0.0.1",
            rate: "100"
        },
      - source_1: {
            ipAddress: "10.1.1.3",
            multicastAddress: "239.0.0.1",
            rate: "1000"
        },
      - source_2: {
            ipAddress: "10.2.1.2",
            multicastAddress: "239.0.0.2",
            rate: "10000"
        }
    },
  - subscriberParams: {
      - subscriber_0: {
            maxSubscribers: "5",
            groupAddress: "239.0.0.1",
            sourceSpecific: "true",
            sourceSpecificIPs: "[10.1.1.2],[10.1.1.3],"
        },
      - subscriber_1: {
            maxSubscribers: "5",
            groupAddress: "239.0.0.2",
            sourceSpecific: "false"
        }
    }
`
```

In this example, the multicast test has two elements, “duration” and “network”, 3 “sources” and 2 “subscribers”.

Modifying the parameters of the working model

Similar to modification of parameters of other Lab tests (requires elementId and value) – for “duration” and “network” above.

Requires "elementId", "paramId" when modifying the "sources" and "subscribers" and the "value".

> **Note:** If the user is modifying a "subscriber" and setting the parameter "sourceSpecific" to true, then the complete list of "sourceIp" will also have to be passed as part of the payload multicastNewParams.

URL	https://<ip_address>/api/v1/bps/ multicasttest/	
METHOD	PATCH	
Request Payload	application/json	
	multicastNewParams	Object
Response	Status – 204 (Ok)	
Response payload		

For example, if the user wishes to modify the ipAddress for source_1 and wants to set subscriber_1 to be sourceSpecific, it will work as follows:

- {"multicastNewParams":{"elementId":"subscriber_1", "paramId":"sourceSpecific","value":"true"}}
- {"multicastNewParams":{"elementId":"source_1", "paramId":"ipAddress", "value":"10.2.1.3"}}

The example above will only set sourceSpecific to "true". If the user wants to modify only the sourceSpecificIPs in a subscriber set to true already, then only the new sourceIp list has to be passed, and no "value" parameter.

Please note that if the user wishes to change the sourceIps for subscriber_1 to include 10.1.1.2 and 10.1.1.3 (for instance), then the following payload will have to be passed. It will not append to previous sourceIps.

{"multicastNewParams":{"elementId":"subscriber_1", "paramId":"sourceSpecificIPs", "sourceIp": ["10.1.1.2", "10.1.1.3"]}}

Please note that when passing sourceIp, it has to be a list of IP addresses as ["10.1.1.2", "10.1.1.3"] (note the [ ])

Note: If the user sets sourceSpecific to "true" for a subscriber which was initially false (for example in an event where a new subscriber is added), or did not have any sourceSpecific IPs set in the model itself, the user will be informed of the same through an error. In this case, the user will also have to pass in "sourceIp" parameter as:

{"multicastNewParams":{"elementId":"subscriber_0", "paramId":"sourceSpecific","value":"true", "sourceIp":["10.1.1.2"]}}

Saving the working model

Usage is same as that of how a normal test model is saved. The only difference is the the URL is used.

URL	https://<ip_address>/api/v1/bps/ multicasttest /operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	

Response payload	application/json	
	result	String

URL	https://<ip_address>/api/v1/bps/ multicasttest /operations/saveas	
METHOD	POST	
Request Payload	application/json	
	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Adding elements in multicast test (sources and subscribers)

In order to add a new element, you must to pass in the "type" of component to add, which should either be "source" or "subscriber". If a "source" is being added, then the user will have to pass in "ipAddress", "multicastAddress" and "rate". If a "subscriber" is being added, then the user will have to pass in "maxSubscribers" and "groupAddress" (must be equal to one of the existing "source" multicastAddress).

Note: when a "subscriber" is being added, the value for "sourceSpecific" will be set to false by default. Once the subscriber is added, the user can modify the subscriber and pass in the "sourceIp" after that.

URL	https://<ip_address>/api/v1/bps/ multicasttest /operations/add	
METHOD	POST	
Request Payload	application/json	
	type	String
	maxSubscribers (if adding subscriber)	Long
	groupAddress (if adding subscriber)	String
	ipAddress (if adding source)	String
	multicastAddress (if adding source)	String
	rate (if adding source)	String

Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Deleting elements in multicast test (sources and subscribers)

In order to delete any existing source or subscriber, simply pass in the list of "elementId" to delete.

URL	https://<ip_address>/api/v1/bps/ multicasttest /operations/delete	
METHOD	POST	
Request Payload		
	elementId	List<String>
Response	Status – 200 (Ok)	
Response payload	application/json	

RFC2544 Lab Test

Setting a Lab test as working model for modification.

URL	https://<ip_address>/api/v1/bps/rfc2544test	
METHOD	PATCH	
Request		
Content Type	application/json	
Request Payload	template	String
Response		
Status	204 OK	
Content Type	None	
Response payload	None	

Viewing the parameters of the working model

URL	https://<ip_address>/api/v1/bps/rfc2544test/	
METHOD	GET	

Response	Status – 200 (Ok)	
Response payload	application/json	
	rfcParams	Object

Modifying the parameters of the working model

Similar to modification of parameters of working model with components, but does not require a "componentId" and a "paramId". Only the "elementId" and "value" has to be passed as 'rfc2544Params".

URL	https://<ip_address>/api/v1/bps/rfc2544test/	
METHOD	PATCH	
Request Payload	application/json	
	'rfc2544Params"	Object
Response	Status – 204 (Ok)	
Response payload		

At the end of the document, please find the type of parameters and their type which can be modified.

Saving the working model

Usage is same as that of how a normal test model is saved. Only difference is that the URL used.

URL	https://<ip_address>/api/v1/bps/rfc2544test/operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

URL	https://<ip_address>/api/v1/bps/rfc2544test/operations/saveas	
METHOD	POST	
Request Payload	application/json	

	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Lawful Intercept Lab Test

Known Issues / Limitations:

- Although elementId "pattern" contains the params"numEntries" and "pattern", the user can only change the value for "numEntries". By default, "pattern" will be set to "Ixia"
- For "directentries" elementId, the user cannot modify anything (from UI shown as Triggers). These will only be modifiable through GUI.
- The canned superflow names as displayed in Browse are not the actual names of the superflows. Below is a list of the canned Lawful Intercept Superflows, the user should pass the superflow name listed in the column titled "Actual SuperFlow Name"

Displayed SuperFlow name	Actual SuperFlow Name
Facebook Chrome (Lawful Intercept)	BreakingPoint Facebook Chrome (Lawful Intercept)
Gmail (Lawful Intercept)	BreakingPoint Gmail (Lawful Intercept - Deprecated)
Google Mail (Lawful Intercept)	BreakingPoint Google Mail (Lawful Intercept)
HTTP Request (Lawful Intercept)	BreakingPoint HTTP Request (Lawful Intercept)
IMAPv4-Advanced (Lawful Intercept)	BreakingPoint HTTP Request (Lawful Intercept)
SIP/RTP Call (Lawful Intercept)	BreakingPoint SIP/RTP Call (Lawful Intercept)
SMTP Email (Lawful Intercept)	BreakingPoint SMTP Email (Lawful Intercept)
Windows Live Messenger v15 (Lawful Intercept)	BreakingPoint SMTP Email (Lawful Intercept)

Setting a Lab test as working model for modification

URL	https://<ip_address>/api/v1/bps/lawfulinterceptlabtest	
METHOD	PATCH	
Request		

Content Type	application/json	
Request Payload	template	String
Response		
Status	204 OK	
Content Type	None	
Response payload	None	

Viewing the parameters of the working model

URL	https://<ip_address>/api/v1/bps/ lawfulinterceptlabtest /	
METHOD	GET	
Response	Status – 200 (Ok)	
Response payload	application/json	
	lawfulInterceptParams	Object

Modifying the parameters of the working model

Similar to modification of parameters of working model with components, but does not require a "componentId" and a "paramId". Only the "elementId" and "value" has to be passed as “lawfulInterceptlabParams”.

URL	https://<ip_address>/api/v1/bps/ lawfulinterceptlabtest /	
METHOD	PATCH	
Request Payload	application/json	
	lawfulInterceptlabParams	Object
Response	Status – 204 (Ok)	
Response payload		

Note: Whenever a target is set to active, then by default, that target will be populated with the following parameters. The user should change them accordingly:

elementId	Default Value
superflowName	“BreakingPoint Google Mail (Lawful Intercept)”

elementId	Default Value
fieldType	"phone"
intervalType	"quantity"
quantityInterval	1000
numEntries	10

Saving the working model

Usage is same as how a normal test model is saved. The only difference is that the URL is used.

URL	https://<ip_address>/api/v1/bps/ lawfulinterceptlabtest /operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

URL	https://<ip_address>/api/v1/bps/ lawfulinterceptlabtest /operations/saveas	
METHOD	POST	
Request Payload	application/json	
	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

LTE Lab Test

Setting a Lab test as working model for modification.

URL	https://<ip_address>/api/v1/bps/ltelabtest/

METHOD	PATCH	
Request		
Content Type	application/json	
Request Payload	template	String
Response		
Status	204 OK	
Content Type	None	
Response payload	None	

Viewing the parameters of the working model

URL	https://<ip_address>/api/v1/bps/ltelabtest/	
METHOD	GET	
Response	Status – 200 (Ok)	
Response payload	application/json	
	lteParams	Object

Modifying the parameters of the working model

Similar to modification of parameters of working model with components, but does not require a "componentId" and a "paramId". Only the "elementId" and "value" has to be passed as "lteLabParams".

URL	https://<ip_address>/api/v1/bps/ltelabtest/	
METHOD	PATCH	
Request Payload	application/json	
	lteLabParams	Object
Response	Status – 204 (Ok)	
Response payload		

The user cannot modify the MME using the above PATCH. For modification of MME, one of the following operations has to be used.

Adding an MME

URL	https://<ip_address>/api/v1/bps/ltelabtest/operations/addmme	
METHOD	POST	
Request Payload	application/json	
	mme_ip	IP Address of the MME to be added
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Removing an MME

URL	https://<ip_address>/api/v1/bps/ltelabtest/operations/removemme	
METHOD	POST	
Request Payload	application/json	
	mme_ip	IP Address of the MME to be removed
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Modifying the MME

URL	https://<ip_address>/api/v1/bps/ltelabtest/operations/modifymme	
METHOD	POST	
Request Payload	application/json	
	oldMme	Old IP Address of the MME to be modified
	newMme	New IP Address of the MME
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

Saving the working model

Usage is same as that of how a normal test model is saved. Only difference is the the URL is used.

URL	https://<ip_address>/api/v1/bps/ltelabtest/operations/save	
METHOD	POST	
Request Payload		
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

URL	https://<ip_address>/api/v1/bps/ltelabtest/operations/saveas	
METHOD	POST	
Request Payload	application/json	
	name	String
	force	Boolean
Response	Status – 200 (Ok)	
Response payload	application/json	
	result	String

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Importing a BPT

When a test model or network neighborhood is exported, the components and the component parameters are saved in a .bpt file

This command will upload a BPT (can be a test model BPT or a Network Neighborhood BPT). If the BPT is successfully uploaded, it sends back the name with which the BPT was uploaded and saved on to the system.

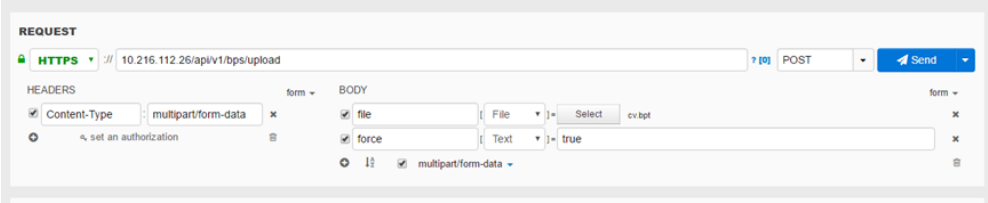
Request payload contains the file to be uploaded.

URL	https://<ip_address>/api/v1/bps/upload	
METHOD	POST	
Request		

Content Type	multipart/form-data	
Content Disposition		
Type	form-data	
name	force	Boolean
Content Disposition		
Type	form-data	
name	file	File
Content Type	application/xml	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String

Note: There may be an instance where you attempt to import a BPT and the same name already exists. In such an event, the user will have to provide a "force" parameter, which is not a json object, but normal form data which is passes as part of the request in addition to the file being uploaded.

Example #1: An example of this upload process as seen on the DHC REST extension on Chrome:



Example #2: An example of this upload process in Python using the 'Requests' library:

```
service = 'https://' + self.ipstr + '/api/v1/bps/upload'
fileName = basename(filePath)
files = {'file': (fileName, open(filePath, 'rb'), 'application/xml')}
jdata = {'force':force}
print jdata
r = self.session.post(service, files=files, data=jdata, verify=False)
```

Note: It is not required to pass the **Expires** parameter.

The corresponding request's header sent for both of the examples above is:

Content-Type: multipart/form-data; boundary=10d8504f9a5d41d9b653c585d2ec63a4

--10d8504f9a5d41d9b653c585d2ec63a4

Content-Disposition: form-data; name="force"

True --10d8504f9a5d41d9b653c585d2ec63a4

Content-Disposition: form-data; name="file"; filename="cv.bpt" Content-Type: application/xml

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Exporting a BPT

Exporting a BPT by Name

This will export the BPT of the given testname.

URL	https://<ip_address>/api/v1/bps/export/bpt/testname/<testname>	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/xml	
Response Payload	The bpt of the test	
Example request	https://10.10.10.10/bps/api/v1/bps/export/bpt/testname/AppSim	

Exporting a BPT by TestId

This will export the BPT of the given testid.

URL	https://<ip_address>/api/v1/bps/export/bpt/testid/<testid>	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	

Response		
Status	200 OK	
Content Type	application/xml	
Response Payload	The bpt of the test	
Example request	https://10.10.10.10/api/v1/export/bpt/testid/TEST-29	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Exporting a Summary of All Tests

This command will produce a csv file containing a short summary of all the tests that have been run on the chassis.

URL	https://<ip_address>/api/v1/bps/export/tests	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/csv	
Response Payload	A CSV containing summary of all the tests	
Example request	https://10.10.10.10/api/v1/bps/export/tests	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Exporting a Report

This command will produce the detailed report for a given testid in a specified format.

Supported formats: pdf ,csv ,rtf, html, xml, zip

URL	https://<ip_address>/api/v1/bps/export/report/<test_id>/<format>

METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/csv, application/pdf, application/zip, application/rtf, application/html	
Response Payload	File	
Example request	https://10.10.10.10/api/v1/bps/export/report/TEST-29/pdf	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Importing a Packet Capture

(For Recreate component)

This command will upload a packet capture (extension "cap" or "pcap"). If the capture is successfully uploaded, it returns the name of the capture of file that was uploaded and saved on the system.

Request payload contains the file to be uploaded.

URL	https://<ip_address>/api/v1/bps/upload/capture	
METHOD	POST	
Request		
Content Type	multipart/form-data	
Content Disposition		
Type	form-data	
name	file	File
Content Type	multipart/form-data	
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	result	String

Example using Python's Requests library.

```
service = 'https://' + self.ipstr + '/api/v1/bps/upload/capture'
fileName = basename(filePath)
files = {'file': (fileName, open(filePath, 'rb'), 'multipart/form-data')}
r = self.session.post(service, files=files, verify=False)
```

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Modifying the Evasion Profile

An evasion profile is used in case of a security component test. For any working model, if that test contains a security component, then it is shown to user as "attackProfile". Here the user can only change the profile they wish to use and not settings within. In order to change the evasion settings, the following similar steps have to be followed:

- Retrieve the template of a pre-existing saved evasion profile using its name
- Modify the settings.
- Save the evasion profile.

Once the profile has been saved, the user should modify the "attackProfile" element in their working model (if the profile was saved with a different name).

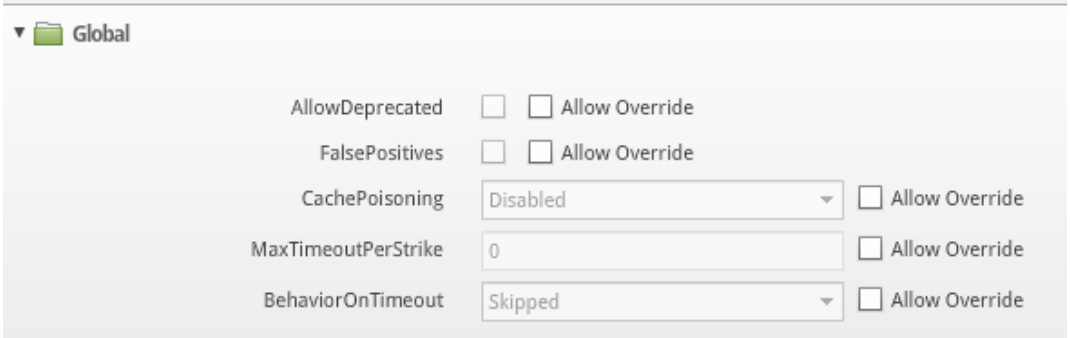
Retrieve an evasion profile for modification

URL	https://<ip_address>/api/v1/bps/evasionprofile/	
METHOD	PATCH	METHOD
Request	Request	Request
Content Type	application/json	Content Type
Request Payload	template	Request Payload
Response	Response	Response

Viewing the evasion profile

All the settings configured will be displayed to the user. The value “Override Not Enabled” means that the user has not set this value and if the user wishes to change a particular value, this state will change. It is same as:

URL	https://<ip_address>/api/v1/bps/evasionprofile	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	A json object showing the evasion profile settings	Object



A sample of a GET response:

```
- evasionSettings: {
    - Global: {
          AllowDeprecated: "Override Not Enabled",
          FalsePositives: "Override Not Enabled",
          CachePoisoning: "Override Not Enabled",
          MaxTimeoutPerStrike: "Override Not Enabled",
          BehaviorOnTimeout: "Override Not Enabled"
      },
    - Ethernet: {
          MTU: "Override Not Enabled"
      },
    - IP: {
          IPEvasionsOnBothSides: "Override Not Enabled",
          ReadWriteWindowSize: "Override Not Enabled",
          TTL: "Override Not Enabled",
          TOS: "Override Not Enabled",
          MaxFragSize: "Override Not Enabled",
          MaxWriteSize: "Override Not Enabled",
          MaxReadSize: "Override Not Enabled",
          FragEvasion: "Override Not Enabled",
          FragPolicy: "Override Not Enabled",
          FragOrder: "Override Not Enabled",
          RFC3514: "Override Not Enabled"
      },
```


Note: In this example, there are three parameters: "componentId" ("Global"), "Ethernet" and "IP"; "elementId" (AllowDeprecated) and the "value" they hold.

Modifying the evasion profile

In order to modify a parameter, the user must pass the "componentId", the "elementId" and the "value".

If the user wishes to disable some element so that its value becomes "Override Not Enabled", then for that elementId, the value "disable" must be passed.

URL	https://<ip_address>/api/v1/bps/evasionprofile/operations/modify	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	componentId	String
	elementId	String
	value	String

Response		
Status	200 OK	
Content Type	application/json	
Response payload	result	String

Saving the evasion profile

Usage is similar to the other evasion commands but in the case of an evasion profile, force parameter is not present.

You cannot save an evasion profile with a name that already exists. No provision for overwriting an existing profile is available.

Table 1

URL	https://<ip_address>/api/v1/bps/ evasionprofile/operations/save	
METHOD	POST	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String

Table 2

URL	https://<ip_address>/api/v1/bps/ evasionprofile/operations/saveas	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	name	String

	force	Boolean
Response		
Status	Status – 200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"name":"custom evasion"} [or] {"name":"custom evasion", “force”:”false”}	Save the evasion profile with a new name, but do not overwrite if an evasion profile with same name exists
	{"name":"custom evasion", “force”:”true”}	Save the evasion profile with a new name, and overwrite if an evasion profile with same name exists

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Aggregate, Component and Protocol Level Statistics

Getting the Aggregate Statistics

In order to get the aggregate stats (which are only available after a test ends) the user must POST the ‘testid’ of the test, which is the same as the ‘runid’ of the running test.

Note: The user must run this only once the test ends for complete data

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getAggregateStats	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	testid	String
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	A json object depicting the aggregate statistics	Object
Sample Request Payload	{"testid":"TEST-44”}	

Getting Component Level Statistics

In order to the get the component level stats (only available after a test ends), the user must POST the ‘testid’ of the test, which is the same as the ‘runid’ of a running test.

Note: If the user wishes to view the components in a test, use the operation “getComponents” as explained below

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getComponentStats	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	testid	String
	componentId	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	A json object depicting the component level statistics for the given componentId	Object
Sample Request Payload	{"testid":"TEST-44", “componentId”:”appsim_1”}	

Getting Protocol Level Statistics

Important: The protocol stats are also defined on a per component basis. The user must pass the “componentId” in addition to “testid” to get the protocol stats for that particular component.

In order to the get the component level stats (only available after a test ends), the user must POST the ‘testid’ of the test, which is the same as the ‘runid’ of a running test.

Note: If the user wishes to view the components in a test, use the operation “getComponents” as explained below.

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getProtocolStats	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	testid	String
	componentId	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	A json object depicting the protocol level statistics for the given componentId	Object
Sample Request Payload	{"testid":"TEST-44", “componentId”:”appsim_1”}	

Getting the Components of a Test for Component and Protocol Level Statistics

Note: In order to retrieve the protocol level and component level stats above, you must pass the correct componentId as a parameter. The below method can be used to retrieve a list of all the components in the test. The user must know the name of the test. (This is the same name which is passed when starting a test run)

URL	https://<System Controller IP>/api/v1/bps/tests/operations/getComponents	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	modelName	String
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"modelName":"AppSim"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Changing the Card Configuration of Slots

The cards support various configurations:

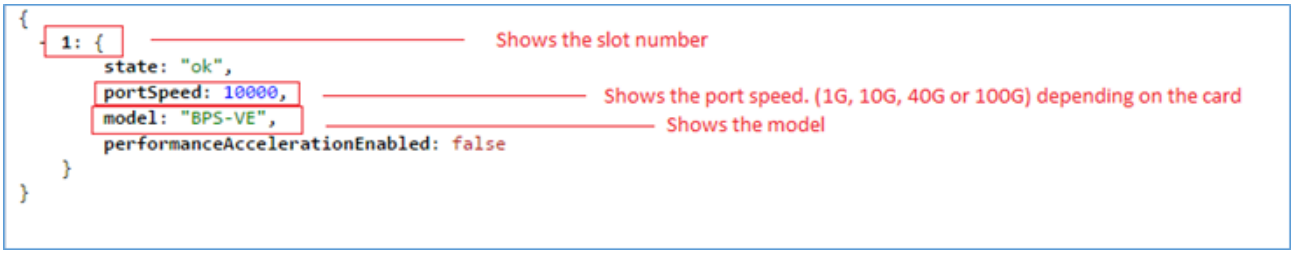
- Mode Change for PS cards between IxLoad, BPS and BPS-L2/3 mode (BPS L2/3 mode not applicable for CloudStorm)
- FanOut: for PS cards (only for 100G and 40G)
- Speed: for PS 10G, between 1G and 10G
- Performance Acceleration: for BPS-VE

The below API provides support for all the above operations.

Known Issue/Limitation: Since it is a REST API call, any changes triggered through REST API will not be shown on UI immediately in the slot window. You must refresh the Port UI to view the correct set of changes which have happened. (Possibly happens through Tcl path as well).

URL	https://<ip_address>/api/v1/bps/ports/chassisconfig
METHOD	GET
Request	
Content Type	application/json
Request Payload	
Response	
Status	200 OK
Content Type	application/json
Response Payload	
Sample Request Payload	

The example below shows a response obtained from a BreakingPoint Virtual Edition system.



There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Changing the Card Configuration

The different options are described below the table with sample request payloads. Please review for a better understanding of the command.

URL	https://<ip_ address>/api/v1/bps/ports/operations/changeCardConfig	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	slot	Integer
	action	String [must be one of 'mode', 'fanout', 'speed', 'perfacc'
	mode	Integer (only applicable when 'action' == 'mode')
	fanid	String (different options explained below)
	speed	String (only applicable for PS10G cards.)
	perfacc	String (only applicable for BPS-VE)
Response		

Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload		

1. Mode Change (action = 'mode')
    a. Will only work for PerfectStorm (and CloudStorm) cards
    b. The acceptable values for 'mode' are:
        i. '0' to change to IxLoad mode
        ii. '1' to change to 'BreakingPoint' mode
        iii. '2' to change to 'BreakingPoint L2/3' mode (this will not work for CloudStorm cards)
    c. Sample request payload: {"slot":3, "action":"mode", "mode":2}. This will change the **mode** for slot 3 to BreakingPoint L2/3 mode
2. Fanout (action = 'fanout')
    a. Will only work for PS100G and PS40G cards
    b. The acceptable values for 'fanid' are:
        i. If PS40G, then 'fanid' = 0 will change to 2x40 and 'fanid' = 1 will change to 8x10 mode
        ii. If PS100G, then 'fanid' = 0 will change to 1x100, 'fanid' = 1 will change to 2x40 and 'fanid' = 2 will change to 8x10 mode
    c. Sample request payload: {"slot":3, "action":"fanout", "fanid":1}. This will **fanout** slot 3 to 2x40 in case of 100G cards or 8x10 in case of 40G cards
3. Speed (action = 'speed')
    a. Will only work for PS10G cards
    b. The acceptable values for 'speed' are:
        i. '1G' if switching to 1G speed
        ii. '10G' if switching to 10G speed
    c. Sample request payload: {"slot":1, "action":"speed", "speed":"1G"}. This will switch the PS10G slot 1 to a speed of 1G.
4. Performance Acceleration (action = 'perfAcc')
    a. Will only work for BPS-VE cards
    b. The only acceptable values are 'true' if switching on the perf acc. mode, and 'false' is switching it off
    c. Sample request payload: {"slot":1, "action":"perfacc", "perfacc":"true"}. This will enable performance acceleration on slot 1 of the BPS-VE.

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

RebootCard

This command is used for rebooting the card from the indicated slot of the chassis.

URL	https://<ip_ address>/api/v1/bps/ports/operations/rebootcard	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	cardNumber	Integer
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	String
Sample Request Payload	{"cardNumber":1}	Reboots the card from the given slot of the chassis
Sample Response Payload	{"result" : "Card 1 is rebooting"}	

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

GetSharedComponentSettings

This command is used for retrieving the list of shared component settings associated with the given test model.

URL	https://<ip_address>/api/v1/bps/ tests/operations/getSharedComponentSettings	
METHOD	POST	
Request		

Content Type	application/json	
Request Payload	modelName	String
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	result	Object
Sample Request Payload 1	{"modelName":"testAppSim"}	Returns the shared component settings for the given test model

Sample Response Payload 1	{"result":{'sharedComponentSettings': [{
	'originalValue': '100000',
	'currentValue': '100000',
	'percentage': '100',
	'enabled': True,
	'name': 'maxFlowCreationRate'
	},{
	'originalValue': '20000.0',
	'currentValue': '20000.0',
	'percentage': '100',
	'enabled': True,
	'name': 'totalBandwidth'
	},{
	'originalValue': '100000',
	'currentValue': '100000',
	'percentage': '100',
	'enabled': True,
	'name': 'maximumConcurrentFlows'
	},{
	'originalValue': '0',
	'currentValue': '0',
	'percentage': '100',
	'enabled': False,
	'name': 'totalAttacks'
	},{
	'originalValue': '131068',
	'currentValue': '131068',
	'percentage': '100',
	'enabled': True,
	'name': 'totalAddresses'
	},{
	'originalValue': '1.0',
	'currentValue': '1.0',
	'percentage': '100',
	'enabled': True,
	'name': 'samplePeriod'
	}],
	'testModelName': 'testAppSim'}
	}
		If the given test model exists and is not a test template, the list of shared component settings is returned.

Sample Request Payload 2	{"modelName":"AppSim"}	
Sample Response Payload 2	{"error": "An unexpected condition occurred: Test model AppSim is a template and cannot be modified. Make a copy of it and use it instead. Please consult logs for more detail. "}	The given test model represent a test template. (error code 400)
Sample Request Payload 3	{"modelName":"testAppSim123"}	
Sample Response Payload 3	{"error": "An unexpected condition occurred: Test model testAppSim123 does not exist. Please consult logs for more detail. "}	The given test model does not exist on the chassis. (error code 400)

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

SetSharedComponentSettings

This command is used for modifying the list of shared component settings associated with the given test model.

URL	https://<ip_address>/api/v1/bps/tests/operations/setSharedComponentSettings	
METHOD	POST	
Request		
Content Type	application/json	
Request Payload	modelName	String
	sharedComponentSettings	Array of objects
Response		
Status	200 OK	
Content Type	application/json	

Response Payload	result	Object

Sample Request Payload 1	{
	"modelName":"testAppSim",
	"sharedComponentSettings":[{
	"paramName":"totalBandwidth",
	"paramValue":"200"
	} , {
	"paramName":"maxFlowCreationRate",
	"paramValue":"150"
	}]
	}
		Sets the modification percentage for the given shared component settings

Sample Response Payload 1	{"result":{'sharedComponentSettings': [{
	'originalValue': '100000',
	'currentValue': '150000',
	'percentage': '150',
	'enabled': True,
	'name': 'maxFlowCreationRate'
	},{
	'originalValue': '20000.0',
	'currentValue': '40000.0',
	'percentage': '200',
	'enabled': True,
	'name': 'totalBandwidth'
	},{
	'originalValue': '100000',
	'currentValue': '100000',
	'percentage': '100',
	'enabled': True,
	'name': 'maximumConcurrentFlows'
	},{
	'originalValue': '0',
	'currentValue': '0',
	'percentage': '100',
	'enabled': False,
	'name': 'totalAttacks'
	},{
	'originalValue': '131068',
	'currentValue': '131068',
	'percentage': '100',
	'enabled': True,
	'name': 'totalAddresses'
	},{
	'originalValue': '1.0',
	'currentValue': '1.0',
	'percentage': '100',
	'enabled': True,
	'name': 'samplePeriod'
	}],
	'testModelName': 'testAppSim'}
	}
		If the given test model exists and is not a test template, the updated list of shared component settings is returned

Sample Request Payload 2	{
	"modelName":"testAppSim",
	"sharedComponentSettings":[{
	"paramName":"totalBandwidth",
	"paramValue":"-200"
	}]
	}	

Sample Response Payload 2	{"error": "Value for “totalBandwidth” must be greater than or equal to “0”."}	The percentage cannot be set as negative number (error code 400)

Sample Request Payload 3	{
	"modelName":"testAppSim",
	"sharedComponentSettings":[{
	"paramName":"totalBandwidth",
	"paramValue":"10000000"
	}]
	}	

Sample Response Payload 3	{"error": "Value for “totalBandwidth” must be less than or equal to “9999999”."}	The percentage cannot be set greater than 9999999 (error code 400)

1. If the test model name represents a test template the following error message is received:
   {"error”:"An unexpected condition occurred: Test model <testModelName> is a template and cannot be modified. Make a copy of it and use it instead. Please consult logs for more detail. "}
2. If the test model name does not exist on the chassis the following error message is received:
   {"error”:"An unexpected condition occurred: Test model <testModelName> does not exist. Please consult logs for more detail. "}
3. If the user sets multiple shared components in a single request and one of the values is invalid the entire operation is reverted and the shared component settings remain unchanged.
4. If the user sets the same shared component setting multiple times in a single request, the last value of the shared component setting will be applied.
   Example:

```
For the request payload:
{
  "modelName":"testAppSim",
  "sharedComponentSettings":[ {
        "paramName":"totalBandwidth",
        "paramValue":"200"
    } , {
        "paramName":" totalBandwidth ",
        "paramValue":"250"
    } ]
```

```
}
the value of the "totalBandwidth" shared component will be set to 250.
```

There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

Get Version of Installed ATI Strikepacks, Malware, Evergreen and Daily Malware

This GET command will retrieve a json list of the ATI Strikepacks, Malware, Evergreen & Daily Malware updates installed on the system.

URL	https://<ip_address> /api/v1/admin/cloudupdates/installedversions	
METHOD	GET	
Request		
Content Type	None	
Request Payload	None	
Response		
Status	200 OK	
Content Type	application/json	
Response Payload	Json list of the modules with the corresponding installed versions	String

Python method sipped of usage from the provided BPS_Updates example class which configures and authenticates the "self.session" used below.

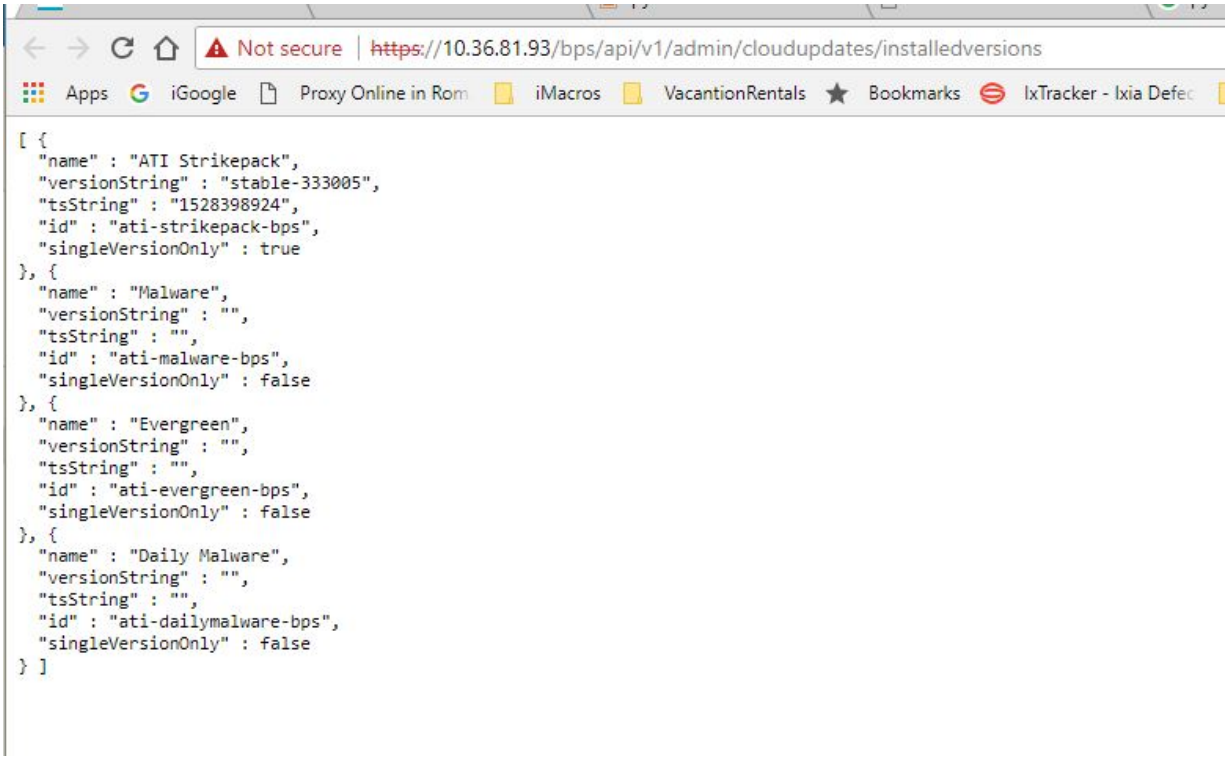
Note: For information on how to download and try the example samples, see Python REST Overview and Examples on page 1303.

```
#installedVersions

def installedVersions(self, enableRequestPrints = True):

#https://10.10.x.x/bps/api/v1/admin/cloudupdates/installedversions

jheaders = {'x-api-key': self.api_key}

service = 'https://' + self.ipstr + '/bps/api/v1/admin/cloudupdates/installedversions'

r = self.session.get(service, verify=False, headers=jheaders)

if(r.status_code == 200):
```

print r.json().get('installedversions')

Browser Example

You can access the REST URL after you've logged in to get the response json.



There are Python script usage examples for most of the URIs. To access the examples, see Getting Started with Python REST Calls on page 1305.

CHAPTER 24 Enhanced Shell

This section includes:

Enhanced Shell Overview

The Enhanced Shell (or `esh`) is a collection of Bash and Tcl scripts that leverages the power of the BreakingPoint Tcl API to simplify common tasks related to using and maintaining a BreakingPoint device. The Enhanced Shell is the default (non-root) user shell beginning with Firmware Release 3.0. However, `esh` may also be installed remotely on a Mac OS X or Linux client system.

Note: The locally-installed Enhanced Shell works with BreakingPoint devices running firmware releases 2.1, 2.2, or 3.0. Any and all Tcl scripts may be used with the Enhanced Shell prompt.

Installation

Local installation of the Enhanced Shell requires that you install the files via Strike Center (see platform-specific commands below). When the files are installed, they will end up in the `$HOME/.bpsh` directory. No files unique to the Enhanced Shell installation will be kept outside that directory, although `esh` will (by default) save tests to `$HOME/tests`, reports to `$HOME/reports`, and packet captures to `$HOME/pcaps`.

For the Enhanced Shell to work properly, you must have `$HOME/.bpsh/bin` in your PATH; usually set in your `.profile`, `.bash_profile`, or `.bashrc`.

Downloading the Enhanced Shell

You can download the Enhanced Shell from the BreakingPoint Systems Start Page.

Note: By default, the BPS IP address is http://10.10.10.10; however, this address may have been changed during the initial configuration. Please see your system administrator for the IP address.

To download the Enhanced Shell:

1. Open a Web browser.
2. Enter the IP address for BreakingPoint in the Address bar and hit Enter. The BreakingPoint Systems Start Page will be displayed.
3. Click the Enhanced Shell link. A new browser window will be displayed with links to the executable files.
4. Click one of the following links:
   a. Enhanced Shell – Mac OS X Version
   b. Enhanced Shell – Linux Version
5. Click the Save button.
6. Select the location to store the .exe file.
7. Click the Save button.

Note: Double-click the executable file to open the Enhanced Shell.

Accessing the Enhanced Shell

Mac OS X

To extract the installation files from the Strike Center, run the following syntax for systems using Mac OS X:

```
curl -s https://strikecenter.ixiacom.com/esh/install.sh | bash
```

Linux

To extract the installation files from the Strike Center, run the following syntax for systems using Linux:

```
wget -qO- https://strikecenter.ixiacom.com/esh/install.sh | bash
```

Windows

Due to the differences in the way `bpsh` works on Windows compared to Mac OS X and Linux, there are no plans to support a Windows version of the Enhanced Shell.

Uninstalling the Enhanced Shell

To uninstall the Enhanced Shell, use either the appropriate platform-specific uninstaller or the following command:

```
rm -rf
~/.bpsh
```

Be sure to remove any modifications you may have made to your `PATH` variable (usually in `.profile`, `.bash_profile`, or `.bashrc`).

Mac OS X

To uninstall the Enhanced Shell from a Mac OS X system, use the following command:

```
curl -s https://strikecenter.ixiacom.com/esh/uninstall.sh |
bash
```

Linux

To uninstall the Enhanced Shell from a Linux system, use the following command:

```
wget -qO- https://strikecenter.ixiacom.com/esh/uninstall.sh |
bash
```

Getting Started

To connect to the Enhanced Shell, type `bps` at the prompt and press Enter. The following help message will be displayed:

```
usage: /Users/username/.bpsh/bin/bps <bps_address> [user]
[password]
* If the username and password are left unspecified,
"admin" and "admin" are assumed.

* If the password is left unspecified,
it is assumed to be the same as the username.

* If the user does not exist,
the script will attempt to create it as the admin user.

* The following shell environment variables have special
significance if they are set. (Set them in your .bashrc
or .bash_profile if desired.)

BPS_HOST : bps_address
BPS_USER : username
BPS_PASSWORD : password
BPS_ADMIN_USER : admin username
BPS_ADMIN_PASSWORD : admin password
```

To show the help message again, regardless of environment variables, type `bps help` at the prompt.

After connecting to a BreakingPoint device, you will be presented with a few initial status messages and a Tcl shell prompt.

Example

```
[+] BPS Enhanced Shell version 96902
[+] Storm CTM - version 2.2.7, build 97960, strikepack 97799, bpsh
97799

[?] type "help" or "?" for help

%
```

If you hit enter at this point, you will then get a named prompt showing your username, host, and group.

Example

```
username@storm
(group:1)%
```

The Help Command

To access help for the Enhanced Shell environment, type `?` or `help` at the prompt.

Example

```
username@storm (group:1)% help

[+] Global variables:
$bps - bps connection
$ch - chassis

COMMAND ARGUMENTS DESCRIPTION
---------- ------------ ----------------------------------------------------
----
alias [alias] [val] - set, unset, or list aliases
cleanup [type] - remove temporary files
config [var] [val] - show, set, unset, or reset configuration settings
ep [cmd] [args] - create, delete, reload, or list evasion profiles
group [group] - choose your current test group explicitly or automatically
help [command] - show this help or detailed help for [command]
host [cmd] [args] - view or modify host system settings
nn [cmd] - list network neighborhoods
pcap [cmd] [args] - export and open pcaps
port [cmd] [args] - list, reserve, or unreserve ports
repeat [opts] [cmd] - repeat a command
report [cmd] [args] - list, export, and open test reports
shell [command] - invoke a local subshell (or execute [command])
strike [cmd] [args] - manage or execute strikes and strike lists
test [cmd] [args] - list or run tests
testseries [cmd] [args] - manage test series
user [cmd] [args] - add, delete, or list users

[+] TIP: use "<command> ?" to get more help for any command
```

Command-Specific Help

Type a question mark after a command to get command-specific help for that particular command.

Syntax

```
username@storm (group:1)%
help
```

Example

The following example displays command-specific help for the `test` command.

```
username@storm (group:1)% test ?

NAME

test - list or run tests

SHORTCUT: t

CONFIGURATION

test_dir: /home/username/tests
test_mode: async

SYNOPSIS

test [cmd] [args]

test - synonomous with "test list all"
test cancel - cancel all tests running under your username
test cancel <tid> [...] - cancel the specified test(s) with matching <tid>
test copy <name> to <newname> - copy test <name> to <newname>
test delete <names> - delete tests matching <names>
test import <URL> [as <name>] [force] - import test [as <name>] from <URL>
test import <pattern> [force] - import local tests matching <pattern>
test export <name> - export test named <name>
test export <pattern> - export tests matching <pattern>
test list - list all tests created by any user
test list <user|mine|all> - list all tests created by <user|mine|all>
test list <user|mine|all> <query> - list all tests matching <query> created by
<user|mine|all>
test list <query> - list all tests with title matching <query>
test list local <pattern> - list local tests matching <pattern>
test rename <name> to <newname> - rename test <name> to <newname>
test run <tests> [options] - run test(s)
test run <tests> series - run test(s) as a test series

DESCRIPTION

You can specify tests to run via the numbers shown in the test list. Test
options: mode (async or sync) and nn (network neighborhood). Test "patterns"
should contain an asterisk to indicate multiple files.
```

Environment Variables

Certain environment variables map to certain arguments that are required to start the Enhanced Shell. As a result, when the value for these variables are set, the arguments that they map to share their set values.

For example, if the variable BPS_HOST is set to `storm` and BPS_USER set to `alice`, the BreakingPoint Shell (without arguments) would attempt to connect to the BreakingPoint system named storm as user alice and password alice.

lists the available environment variables and the arguments they map to. If any of the following environment variables are set or reset, the arguments required to start the Enhanced Shell will also be changed.

Enhanced Shell Environment Variables

Variable	Corresponding Argument
BPS_HOST	bps_address
BPS_USER	username
BPS_PASSWORD	password
	admin username
BPS_PASSWORD	admin password

Enhanced Shell Commands Overview

Whenever possible, the default subcommand (if none is provided) is `list`. So just typing `t` is the same as typing `t list` or `test list`.

Note: Previous experience and knowledge of Tcl scripting is required for use of the Enhanced Shell.

Enhanced Shell Commands

Command	Shortcuts	Description
`alias`	`a`	A macro that represents one or more commands. The alias command allows you to set, unset, or list aliases.
`cleanup`	`cl`, `clean`	Allows you to manually remove temporary files created by the Enhanced Shell. These temporary files may include items such as tests, test series, and Strike Lists.
`config`	`c`, `conf`	Allows you to display, set, unset, or reset configuration settings.
`ep`	`N/A`	Allows you to create, delete, reload, or list evasion profiles.
`group`	`g`	Allows you to select your current test group explicitly or automatically.
`help`	`?`	Allows you to display detailed help for the specified command.
`host`	`h`	Allows you to view or modify the system settings for the host.

Command	Shortcuts	Description
module	m	Allows you to list, install, or uninstall modules.
pcap	pc	Allows you to export and open pcap files.
port	p	Allows you to list, reserve, or unreserve ports.
repeat	x	Allows you to repeat a command.
report	rep	Allows you to list, export, and open test reports.
shell	sh	Allows you to invoke a local subshell (or execute a specified command).
strike	s	Allows you to manage or execute Strikes and Strike Lists.
test	t	Allows you to list or run tests.
testseries	ts	Allows you to manage a test series.
user	u	Allows you to add, delete, or list users.

The Alias Command

An alias is a macro representing one or more commands. A shortcut is a shorter version of a subcommand. The `alias` command allows you to set, unset, or list aliases. An alias can be presented as a single command or as a complex string of commands with argument substitution.

Listing all Aliases

Use the following syntax to display a list of all available aliases.

```
username@storm (group:1)%
alias
```

Example

```
username@storm (group:1)% alias; #yields the
following:

CUSTOM ALIAS COMMAND
------ --------- ----------
no a - alias
no cl, clean - cleanup
no c, conf - config
no g - group
no ? - help
no h - host
no m - module
no pc - pcap
no p - port
no x - repeat
no rep - report
no sh - shell
no s - strike
no t - test
no ts - testseries
no u - user
```

Note: A yes value is displayed in the Custom column indicates that the alias is a user-created alias.

Searching for a Specific Alias

Use the following syntax to determine the aliases for the `group` command.

```
username@storm (group:1)% alias
group
```

Example

```
username@storm (group:1)% alias group; #yields the
following:

CUSTOM ALIAS COMMAND
------ ----- -------
no g - group
```

Use the following syntax to find out what the `g` alias maps to.

```
username@storm (group:1)% alias
g
```

Example

```
username@storm (group:1)% alias g; #yields the
following:

CUSTOM ALIAS COMMAND
------ ----- -------
no g - group
```

Creating an Alias

Use the following syntax to create an alias using tandard Tcl commands.

```
username@storm (group:1)% a hello puts "hello, world!"
[+] creating alias 'hello'
[+] writing aliases to
/Users/username/.bpsh/etc/aliases.conf

username@storm (group:1)% a hello
hello

CUSTOM ALIAS COMMAND
------ ----- --------------------
yes hello - puts {hello, world!}

username@storm (group:1)% hello
hello, world!
```

Use the following syntax to create an alias using Enhanced Shell commands.

```
username@storm (group:1)% a myports port reserve 1/2-3
[+] creating alias 'myports'
[+] writing aliases to
/Users/username/.bpsh/etc/aliases.conf
username@storm (group:1)% a myports

CUSTOM ALIAS COMMAND
------ ------- ------------------
yes myports - port reserve 1/2-3
```

Use the following syntax to create an alias using arguments.

```
username@storm (group:1)% alias ls shell ls
[+] creating alias 'ls'
[+] writing aliases to /Users/username/.bpsh/etc/aliases.conf
username@storm (group:1)% ls *out
perf_2.1_10g.out perf_2.2.6_10g.out perf_2.2.7_10g.out
verbose.out
perf_2.1_1g.out perf_2.2.6_1g.out perf_2.2.7_1g.out
```

Use the following syntax to create an alias using embedded arguments.

```
a sr {p r 1/0-1; s r $args mode sync; pc o 1/0; p u 1/0-1}
```

The above command will create an alias (using the "a" shortcut") called sr that executes the code in brackets as follows: 1. Reserve ports 0 and 1 on slot 1 2. Run the strike matching $args (more in that in a second) in synchronous mode 3. Open the contents of the packet capture buffer from slot 1 port 0 using Wireshark 4. Unreserve ports 0 and 1 on slot 1 The $args variable means to take any arguments to the alias and insert it into the alias at that point. For example:

```
sr tippingpoint dns
```

… will perform the above tasks, running a single security test using strikes matching "tippingpoint" and "dns" anywhere in the strike data fields (a total of 3 strikes currently).

Removing an Alias

Use the following syntax to remove an alias.

```
username@storm (group:1)% a hello none
[+] removing alias 'hello'
[+] writing aliases to
/Users/username/.bpsh/etc/aliases.conf
```

The Cleanup Command

The Enhanced Shell creates temporary test series, tests, and strike lists so the original copies are not altered accidentally when used. The `cleanup` command allows you to manually remove those temporary items.

```
NAME

cleanup - remove temporary files

SHORTCUTS: cl, clean

CONFIGURATION

cleanup_immediately: yes
cleanup_on_load: yes

SYNOPSIS

cleanup [type]

cleanup - synonymous with "cleanup
list"
cleanup list - lists temp files
cleanup types - lists temp file types
cleanup all - remove all temp files
cleanup [type] - remove [type] temp
files
```

Cleanup Command Configuration Options

When the configuration option **cleanup_on_load** is set to yes (the default), temporary items will automatically be removed each time you connect to the BreakingPoint system where those items were created.

When the configuration option **cleanup_immediately** is set to yes (the default), temporary items will automatically be removed as soon as possible after they have been created.

Listing Temporary Items

Use the following syntax to list temporary items.

```
admin@storm (group:1)% cleanup
list
```

Example

```
admin@storm (group:1)% cleanup list
Tests: admin - Security aAjD1UJcFY
Tests: admin - Security sj8pXmwJWc
Tests: admin - Security hrlqdZfW5T
Tests: admin - Security a5kexwb3sh
StrikeLists: admin - Strikelist
aAjD1UJcFY
StrikeLists: admin - Strikelist
sj8pXmwJWc
StrikeLists: admin - Strikelist
hrlqdZfW5T
StrikeLists: admin - Strikelist
a5kexwb3sh
TestSeries: admin - Test Series
WVbOw3RzSw
TestSeries: admin - Test Series
Sezpui4FDU
TestSeries: admin - Test Series
7jrqAqyGGi
TestSeries: admin - Test Series
hhBxu0lwVs
TestSeries: admin - Test Series
E17FLv5DNz
TestSeries: admin - Test Series
BTB4bWeU3h
TestSeries: admin - Test Series
o5C6fQYjNa
TestSeries: admin - Test Series
I0Nt7G6Xaz
```

Listing Temporary Item Types

Use the following syntax to list temporary item types.

```
admin@storm (group:1)% cleanup types
```

Example

```
admin@storm (group:1)% cleanup
types
Tests
StrikeLists
TestSeries
```

Removing all Temporary Items

Use the following syntax to remove all temporary items.

```
admin@storm (group:1)% cleanup
all
```

Removing Specific Types of Temporary Items

Use the following syntax to remove temporary test series.

```
admin@storm (group:1)% cleanup TestSeries
[+] deleting test series 'admin - Test Series
WVbOw3RzSw'
[+] deleting test series 'admin - Test Series
Sezpui4FDU'
[+] deleting test series 'admin - Test Series
7jrqAqyGGi'
[+] deleting test series 'admin - Test Series
hhBxu0lwVs'
[+] deleting test series 'admin - Test Series
E17FLv5DNz'
[+] deleting test series 'admin - Test Series
BTB4bWeU3h'
[+] deleting test series 'admin - Test Series
o5C6fQYjNa'
[+] deleting test series 'admin - Test Series
I0Nt7G6Xaz'
[+] removed 8 temporary files
```

If items are in use by other objects (for example, tests may be part of a test series), then you must delete the containing object before you can delete the objects within that container.

Configuration Command Options

The table below lists the available `config` options along with their descriptions and their default values.

Configuration Command Options

Variable	Description	Default Value
`cleanup_on_load`	Automatically removes temporary items each time a connection to BreakingPoint is established.	Yes
`group_autochange`	Automatically changes the current test group number to the lowest available group number.	Yes

Variable	Description	Default Value
pcap_dir	Identifies the directory that packet capture files are saved to after an export.	$HOME/pcaps
port_force_reserve	Unreserves selected ports if they are currently being used by another user.	Yes
port_mtu	Specifies the default MTU that will be used when reserving ports.	9198
report_dir	Identifies the directory where reports will be saved after an export.	$HOME/reports
report_format	Identifies the default report format used for export.	.pdf
test_dir	Identifies the directory where tests will be saved after an export.	$HOME/tests
test_mode	Indicates the default mode that tests will run in.	async

Listing all Configuration Command Options

Use the following syntax to list all options available for the `config` command.

Syntax

```
username@storm (group:1)%
config
```

Example

```
username@storm (group:1)% config

VARIABLE DEF VALUE
------------------ --- -----------------
---
cleanup_on_load yes yes
group_autochange yes yes
pcap_dir yes /home/username/pcaps
port_force_reserve yes yes
port_mtu yes 9198
prompt_color yes bold yellow
report_dir yes /home/username/reports
report_format yes pdf
test_dir yes /home/username/tests
test_mode yes async
```

Listing Specific Configuration Options

To show the options relating to a particular `esh` command, pass a pattern as the first argument to `config`:

```
username@storm (group:1)% config ^port

VARIABLE DEF VALUE
------------------ --- -----------------
---
port_force_reserve yes yes
port_mtu yes 9198
```

To look for a single option:

```
username@storm (group:1)% config port_
mtu

VARIABLE DEF VALUE
------------------ --- -----------------
---
port_mtu yes 9198
```

Setting Configuration Options

To change the value of a `config` option, add the value to the end of the command string:

```
username@storm (group:1)% config port_mtu
1500
[+] setting 'port_mtu' to '1500'
username@storm (group:1)% config port_mtu

VARIABLE DEF VALUE
------------------ --- --------------------
port_mtu no 1500
```

Resetting Configuration Options

To reset the value of a `config` option to the default value:

```
username@storm (group:1)% config port_mtu
default
[+] resetting 'port_mtu' to '9198'
```

To reset all `config` options to their default values:

```
username@storm (group:1)% config all
default
[+] resetting configuration to defaults
[+] using the default configuration
settings
```

The Evasion Profile Command

The evasion profile (`ep`) command allows you to manipulate the evasion profiles used in security tests. It allows you to create, delete, reload, or list evasion profiles.

Listing all Evasion Profiles

The `ep` command with no arguments or executed via `ep list` will return a list of all evasion profiles on the system.

Listing Specific Evasion Profiles

Use the following example to see how to conduct a non-case-sensitive search for the substring called ordered in an evasion profile name.

```
username@storm (group:1)% ep list ordered
[+] showing evasion profiles created by all users matching 'ordered'
1: IP: Ordered 16 byte, overlapping (new)
2: IP: Ordered 16 byte, overlapping (old)
3: IP: Ordered 24 byte fragments
4: IP: Ordered 8 byte fragments
5: Ordered 2byte Overlap
6: Ordered 8byte overlap
7: TCP: Ordered 1 byte segments
8: TCP: Ordered 1 byte segments, duplicate last packet
9: TCP: Ordered 1 byte segments, interleaved duplicate segments with invalid
TCP checksums
10: TCP: Ordered 1 byte segments, interleaved duplicate segments with null TCP
control flags
11: TCP: Ordered 1 byte segments, interleaved duplicate segments with out-of-
window sequence numbers
12: TCP: Ordered 1 byte segments, interleaved duplicate segments with requests
to resync sequence numbers mid-stream
```

If the first argument after the list subcommand is the word mine, only profiles created by you will be returned.

If the first argument after the list subcommand is the word all, profiles created by all users will be returned.

If the first argument after the list subcommand is a valid system username, then only profiles created by that specific user will be returned.

Copying Evasion Profiles

Review the following example to examine the syntax for copying an evasion profile by name.

```
username@storm (group:1)% ep copy TCP: Ordered 1 byte segments to My New
Evasion Profile
[+] copying evasion profile 'TCP: Ordered 1 byte segments' to 'My New Evasion
Profile'
```

You can also select an evasion profile based on the number preceding the name in the list of the most recent ep list results. Review the following example to examine the syntax for copying an evasion profile by number.

```
username@storm (group:1)% ep list ordered
[+] showing evasion profiles created by all users matching 'ordered'
1: IP: Ordered 16 byte, overlapping (new)
2: IP: Ordered 16 byte, overlapping (old)
3: IP: Ordered 24 byte fragments
4: IP: Ordered 8 byte fragments
5: Ordered 2byte Overlap
6: Ordered 8byte overlap
7: TCP: Ordered 1 byte segments
8: TCP: Ordered 1 byte segments, duplicate last packet
9: TCP: Ordered 1 byte segments, interleaved duplicate segments with invalid
TCP checksums
10: TCP: Ordered 1 byte segments, interleaved duplicate segments with null TCP
control flags
11: TCP: Ordered 1 byte segments, interleaved duplicate segments with out-of-
window sequence numbers
12: TCP: Ordered 1 byte segments, interleaved duplicate segments with requests
to resync sequence numbers mid-stream
username@storm (group:1)% ep copy 7 to My New Evasion Profile
[+] copying evasion profile 'TCP: Ordered 1 byte segments' to 'My New Evasion
Profile'
```

Creating Evasion Profiles

Review the following example to see how to create an evasion profile name My New Evasion Profile that uses ordered 1-byte TCP segments and duplicates the last TCP segment.

```
username@storm (group:1)% ep create My New Evasion Profile using -
TCP.DuplicateLastSegment true -TCP.MaxSegmentSize 1
[+] creating evasion profile 'My New Evasion Profile'
username@storm (group:1)% ep info My New Evasion Profile
[+] showing parameters for evasion profile 'My New Evasion Profile'

OPTION VALUE
------------------------ ----
-TCP.DuplicateLastSegment true
-TCP.MaxSegmentSize 1
```

Deleting Evasion Profiles

Review the following example to examine the syntax for deleting an evasion profile by name.

```
username@storm (group:1)% ep delete My New Evasion
Profile
[+] deleting evasion profile 'My New Evasion Profile'
```

Review the following example to examine the syntax for deleting an evasion profile by number.

```
username@storm (group:1)% ep list evasion
[+] showing evasion profiles created by all users matching
'evasion'
1: Browser: High Evasion
2: Browser: Low Evasion
3: Browser: Medium Evasion
4: Confirmed Kill Firewall 2010-07-23 AppSimSmartFlowEvasion
5: DCERPC: High Evasion
6: DCERPC: Low Evasion
7: DCERPC: Medium Evasion
8: Default evasion settings
9: HTTP: Apache High Evasion
10: HTTP: Apache Low Evasion
11: HTTP: Apache Medium Evasion
12: HTTP: Apache No Evasion
13: HTTP: IIS High Evasion
14: HTTP: IIS Low Evasion
15: HTTP: IIS Medium Evasion 1
16: HTTP: IIS Medium Evasion 2
17: HTTP: IIS No Evasion
18: My New Evasion Profile
username@storm (group:1)% ep delete 18
[+] deleting evasion profile 'My New Evasion Profile'
```

Review the following example to examine the syntax for deleting more than one evasion profile by number.

```
username@storm (group:1)% ep list new evasion profile
[+] showing evasion profiles created by all users matching 'new evasion
profile'
1: My New Evasion Profile
2: My New Evasion Profile 2
3: My New Evasion Profile 3
username@storm (group:1)% ep delete 1-3
[+] deleting evasion profile 'My New Evasion Profile'
[+] deleting evasion profile 'My New Evasion Profile 2'
[+] deleting evasion profile 'My New Evasion Profile 3'
```

To delete everything in the resulting list, use an asterisk instead of a number. Review the following example to examine the syntax for deleting everything in the resulting list.

```
username@storm (group:1)% ep list new evasion profile
[+] showing evasion profiles created by all users matching 'new evasion
profile'
1: My New Evasion Profile
2: My New Evasion Profile 2
3: My New Evasion Profile 3
username@storm (group:1)% ep delete *
[+] deleting evasion profile 'My New Evasion Profile'
[+] deleting evasion profile 'My New Evasion Profile 2'
[+] deleting evasion profile 'My New Evasion Profile 3'
```

Obtaining the Details of an Evasion Profile

Review the following example to examine the syntax for obtaining the details of an evasion parameter by name.

```
username@storm (group:1)% ep list Browser: High Evasion
[+] showing parameters for evasion profile 'Browser: High
Evasion'

OPTION VALUE
-------------------------------- --------
-HTML.HTMLUnicodeEncoding UTF_7
-HTML.HTMLUnicodeUTF7EncodingMode standard
-HTTP.ServerChunkedTransfer true
-HTTP.ServerChunkedTransferSize 3
-HTTP.ServerCompression gzip
```

Review the following example to examine the syntax for obtaining the details of an evasion parameter by number.

```
username@storm (group:1)% ep list browser
[+] showing evasion profiles created by all users matching
'browser'
1: Browser: High Evasion
2: Browser: Low Evasion
3: Browser: Medium Evasion
username@storm (group:1)% ep list 1
[+] showing parameters for evasion profile 'Browser: High
Evasion'

OPTION VALUE
-------------------------------- --------
-HTML.HTMLUnicodeEncoding UTF_7
-HTML.HTMLUnicodeUTF7EncodingMode standard
-HTTP.ServerChunkedTransfer true
-HTTP.ServerChunkedTransferSize 3
-HTTP.ServerCompression gzip
```

Browsing Evasion Profile Parameters

To browse evasion profile parameters, use the `params` subcommand. With no arguments, `ep params` returns the list of all available evasion profile parameters. If you provide an argument to the `params` subcommand, your search will be narrowed considerably. The search syntax used can be either of the following examples:

```
ep params
<group>
```

```
ep params
<group>.<option>
```

```
ep params
.<option>
```

Note: The group or option can be a substring.

For example, to show all options within the UDP group:

```
username@storm (group:1)% ep params
udp

UDP.DestinationPort (int)
default: 0
values: min: 0, max: 65535

UDP.DestinationPortType (enum)
default: default
values: default, random, static

UDP.SourcePort (int)
default: 0
values: min: 0, max: 65535

UDP.SourcePortType (enum)
default: default
values: default, random, static
```

To show just the UDP options containing the word type:

```
username@storm (group:1)% ep params
udp.type

UDP.DestinationPortType (enum)
default: default
values: default, random, static

UDP.SourcePortType (enum)
default: default
values: default, random, static
```

To find all options (regardless of group) containing the word handshake:

```
username@storm (group:1)% ep params
.handshake

TCP.SkipHandshake (boolean)
default: false
values: true or false

TCP.SneakAckHandshake (boolean)
default: false
values: true or false
```

Renaming Evasion Profiles

Review the following example to examine the syntax for renaming an evasion profile by name.

```
username@storm (group:1)% ep rename My Old Evasion Profile to My New Evasion
Profile
[+] renaming evasion profile 'My Old Evasion Profile' to 'My New Evasion
Profile'
```

Review the following example to examine the syntax for renaming an evasion profile by number.

```
username@storm (group:1)% ep list old evasion profile
[+] showing evasion profiles created by all users matching 'old evasion
profile'
1: My Old Evasion Profile
2: My Old Evasion Profile 2
3: My Old Evasion Profile 3

username@storm (group:1)% ep rename 1 to My New Evasion Profile
[+] renaming evasion profile 'My Old Evasion Profile' to 'My New Evasion
Profile'
```

The Group Command

The group command allows you to choose your current test group explicitly or automatically.

```
NAME

group - choose your current test group explicitly or
automatically

SHORTCUT: g

CONFIGURATION

group_autochange: yes

SYNOPSIS

group [group]

group - select the next available group
group <1-99> - select a group between 1 and 99
```

Group Command Configuration Options

When the configuration option `group_autochange` is set to yes (the default), each time you run a test, your group number will be automatically changed to the lowest available group number. When set to no, you will need to change your group manually.

Automatically Selecting a Group

When you execute the group command with no arguments, it will select the lowest-numbered test group that is not currently running a test for your user.

Example

```
username@storm (group:2)%
group
[+] now using group 1
```

Explicitly Selecting a Group

When you execute the group command with a specific group number as an argument, it will change to that test group number whether or not it is busy running a test.

Example

```
username@storm (group:1)%
group 2
[+] now using group 2
```

The Help Command

The `help` command, without arguments, shows only a summarized list of available commands. If you add a specific command name or shortcut as an argument, it will give you help for that specific command.

Using a question mark as the first argument to any command performs the same function as using that command as the argument to the `help` command. For example, the following commands will give the same help output:

```
help
test
test ?
```

The Host Command

The `host` command provides slightly different functionality depending on whether you are using a locally-installed `esh` client or you are using the `esh` on a BreakingPoint device.

Example

On a BreakingPoint device:

```
NAME

host - view or modify host system settings

SHORTCUT: h

SYNOPSIS

host [cmd] [args]

host (no arguments) - synonomous with "host show all"
host get <parameter> - show current value for <parameter>
host install <URL> [slot <num>] - install the update at <URL> [on slot <num>]
host reboot - reboot the system
host revert [to] <previous|factory> - revert system to previous or factory
state
host set <param> <value> [<param> <value> [...]] - set host parameters
host show [all|config|running] - show current host settings
host version - show software versions
```

On a locally-installed `esh` client:

```
NAME

host - view or modify host system settings

SHORTCUT: h

SYNOPSIS

host [cmd] [args]

host install <URL> [slot <num>] - install the update at <URL> [on slot <num>]
host reboot - reboot the system
host revert [to] <previous|factory> - revert system to previous or factory
state
host version - show software versions
```

Installing Updates Using the Enhanced Shell

Note: You must be an administrator to install updates.

An update is either an OS update or an ATI update. The update can be specified as a file name (with or without a file path) or a URL.

Note: Only HTTP URLs are supported at this time.

Review the following example to examine the syntax for uploading and installing an update.

```
admin@storm (group:1)% host install Downloads/update-66329-
98194.bps
upload |=======================================| 100 %
update |======================================= | 98 %
```

After an OS update, the device will automatically restart. After an ATI update, the device will not restart automatically but a reboot is highly recommended.

Example

Review the following example to examine the syntax for installing an update on a slot other than slot 0.

```
admin@storm (group:1)% host install Downloads/update-66329-98194.bps
slot 1
upload |=======================================| 100 %
update |======================================= | 98 %
```

Rebooting

Note: You must be an administrator to reboot the system.

Review the following example to examine the syntax for rebooting the device.

```
admin@storm (group:1)% host
reboot
[+] attempting to reboot
system...
```

Reverting Software With the Enhanced Shell

Note: You must be an administrator to revert the system to a previous state.

Example

To revert the system to the previously-installed state:

```
admin@storm (group:1)% host revert to
previous
```

To revert the system to the factory-installed state:

```
admin@storm (group:1)% host revert to
factory
```

Note: Reverting a system will cause it to reboot and may require you to re-configure the system Out of Box Experience (OBE) parameters via telnet to 10.10.10.10 or serial connection.

Version Numbers

Review the following example to examine the syntax for determining the OS, build, ATI, and `bpsh` version.

```
admin@ati22 (group:1)% host version
[+] Storm CTM - version 2.2.6, build 94526, strikepack 94519, bpsh
94441
```

Showing Host Settings

The `host` command is only available when running `esh` directly on the BreakingPoint system.

Note: The commands `host`, `host show`, and `host show all` are synonymous.

Example

Review the following example to examine the syntax for showing all the current host settings.

```
admin@storm (group:1)% host show all

CONFIGURABLE CONFIGURED
PARAMETER VALUE
------------ ----------------------------
---
dhcp false
dns1 10.0.0.100
dns2 10.0.1.100
gw 10.0.0.1
hostname storm.example.com
ip 10.0.0.25
netmask 23

NETWORK CURRENT
SETTING VALUE
------------ ----------------------------
---
address1 10.0.0.25/23:Global
address2 fe80::21a:c5ff:fe00:1c6/64:Link
curr4mask 23
curr6masklink 64
currip 10.0.0.25
currip4 10.0.0.25
currip6link fe80::21a:c5ff:fe00:1c6
```

Review the following example to examine the syntax for showing only the configurable parameter settings.

```
admin@storm (group:1)% host show config

CONFIGURABLE CONFIGURED
PARAMETER VALUE
------------ ----------------------------
---
dhcp false
dns1 10.0.0.100
dns2 10.0.1.100
gw 10.0.0.1
hostname storm.example.com
ip 10.0.0.25
netmask 23
```

Review the following example to examine the syntax for showing only the running/configured parameters representing the current state of the device.

```
admin@storm (group:1)% host show running

NETWORK CURRENT
SETTING VALUE
------------ ----------------------------
---
address1 10.0.0.25/23:Global
address2 fe80::21a:c5ff:fe00:1c6/64:Link
curr4mask 23
curr6masklink 64
currip 10.0.0.25
currip4 10.0.0.25
currip6link fe80::21a:c5ff:fe00:1c6
```

Getting Specific Host Settings

The `host get` command is only available when running `esh` directly on the BreakingPoint system. Available parameters include dhcp, dns1, dns2, gw, hostname, ip, and netmask.

Example

Review the following example to examine the syntax for getting a specific parameter (dns2 in this case).

```
admin@storm (group:1)% host get
dns2
10.0.1.100
```

Changing Host Settings

The `host set` command is only available when running `esh` directly on the BreakingPoint system. You must be an administrator to use this command. Available parameters include dhcp, dns1, dns2, gw, hostname, ip, and netmask.

To set a specific parameter (dns2, in this example):

```
admin@storm (group:1)% host set dns2 10.0.0.101
[+] setting new host dhcp, dns1, dns2, gw, hostname, ip, netmask
values
```

Note: All values will be set again (even if no changes are being made to them) due to the way parameters are handled internally.

The Module Command

The `module` command allows you to list, install, or uninstall modules.

Listing all Modules

Review the following example to examine the syntax for listing all modules.

```
username@storm (group:1)% module list all

REPO MODULE LVER RVER DESCRIPTION
----- ---------- ------- ------- ---------------------
---
IR local apcon unknown unknown - Apcon Port Management
IR local dev unknown unknown - Developer Tools
IR local rshell unknown unknown - BPS Remote Shell
```

Column Descriptions:

An I in the first column means that the module is currently installed.

An R in the first column means that the module is available in the module repository.

REPO indicates which repository the module is from.

MODULE indicates the module name.

LVER indicates the version number of the local module.

RVER indicates the version number of the module in the repository.

DESCRIPTION indicates the purpose of the module.

Listing installed modules

Use the following syntax to list installed modules.

```
username@storm (group:1)% module list
installed
```

Listing Available Modules

Use the following syntax to list modules available for installation.

```
username@storm (group:1)% module list
available
```

The available modules will depend upon where you installed your Enhanced Shell distribution from, and whether or not any custom module repositories have been added.

Installing Modules

Use the following syntax to install specific modules.

```
username@storm (group:1)% module install module1 module2
```

Use the following syntax to install all available modules.

```
username@storm (group:1)% module install all
```

Uninstalling Modules

Use the following syntax to uninstall specific modules.

```
username@storm (group:1)% module uninstall module1
module2
```

Use the following syntax to uninstall all modules.

```
username@storm (group:1)% module uninstall
all
```

The Network Neighborhood Command

The `nn` command allows you to list the available Network Neighborhoods.

Listing Network Neighborhoods

If the first argument after the `list` subcommand is the word mine, only Network Neighborhoods created by you will be returned.

If the first argument after the `list` subcommand is a valid system username, only Network Neighborhoods created by that user will be returned.

If the first argument after the list subcommand is the word all or does not match anything else above, then Network Neighborhoods created by all users will be returned.

Use the following syntax to list all Network Neighborhoods.

```
username@storm (group:1)% nn
list
```

Use the following example to examine the syntax for listing Network Neighborhoods with a specific word in the name (switching in this example).

```
username@storm (group:1)% nn list switching
[+] showing network neighborhoods created by all users matching
'switching'
1: BreakingPoint IPv6 Switching
2: BreakingPoint Resiliency Switching
3: BreakingPoint Switching
```

The Pcap Command

The `pcap` command is used to export and open pcap files.

Configuration Options for the Pcap Command

The configuration option `pcap_dir` is the directory that packet capture files (pcap files) will be saved to after an export. The default location is $HOME/pcaps.

The configuration option `pcap_command` is the command that specifies which application to use to view exported pcap files. The default value is open.

Checking Packet Buffer Status

Use the following syntax to view the number of bytes/frames available for export on all slots and ports.

```
username@storm (group:1)% pcap
status
```

To view the packet capture buffers on specific slots and ports, specify combinations of slots and ports as described in the port command.

Example

```
username@storm (group:1)% pcap status
0/0-1

SLOT 0

PORT 0
txbytes: 1703362507
rxbytes: 2167218377
txframes: 11206185
rxframes: 8948002

PORT 1
txbytes: 1129451072
rxbytes: 1703363303
txframes: 4861552
rxframes: 11206180
```

Exporting Captured Packets to a File

Use the following syntax to export the packet buffer from slot 0, port 1.

```
username@storm (group:1)% pcap export
0/1
```

You can also specify a range of frames/data to export, using an `f` (to specify frames) or an `m` (to specify megabytes).

Example

To export 5Mb of data starting at the 5th Mb on slot 2, ports 0 and 1:

```
username@storm (group:1)% pcap export 2/0-1 range 5-
10m
```

To export the first 10 frames from the packet capture buffer on slot 1, port 0:

```
username@storm (group:1)% pcap export 1/0 range
10f
```

Opening a Pcap After Export

Review the following example to examine the syntax for opening a packet capture file automatically (with your default application for viewing pcap files) after export.

```
username@storm (group:1)% pcap open
2/0-1
```

Note: You cannot automatically open an exported pcap file while running `esh` directly on a BreakingPoint device.

The Port Command

The port command allows you to list, reserve, or unreserve ports.

```
NAME

port - list, reserve, or unreserve ports

SHORTCUT: p

CONFIGURATION

port_force_reserve: yes
port_mtu: 9198

SYNOPSIS

port [cmd] [args]

port - synonomous with "port list"
port configure <slot/port combo> [opts] - configure specified slots/ports with
options
port details [slot/port combo] - show details of specified ports
port list [slot/port combo] - show status of slots and ports
port note <slot/port combo> - list notes for specified slots/ports
port note <slot/port combo> <msg> - set note for specified slots/ports to <msg>
port note <slot/port combo> remove - remove note from specified slots/ports
port reserve [slot/port combo] [opts] - reserve the specified slots/ports with
options
port unreserve [slot/port combo] - unreserve the specified slots/ports

DESCRIPTION

Ports can be specified in many ways. For example, if you wanted to specify
slot 2, ports 4 through 7 (inclusive), you could use any of these
interchangeably: 2/4-7 or 2/4,5,6,7 or 2/4 2/5 2/6 2/7. If you don't specify
any ports, the entire slot is assumed. If you only have a blade in slot 2,
you could leave that off, e.g. 4-7 or 4,5,6,7 or 4 5 6 7. You can also
specify the following options: auto, fullduplex, speed, mtu, and
ignorepause. Example: "port reserve 1/0-3 mtu 1500 fullduplex false speed
100"
```

Configuration Options for the Port Command

When the configuration option `port_force_reserve` is set to yes, ports will be unreserved first if they are already reserved by someone else when you try to reserve them.

The configuration option `port_mtu` specifies the default MTU that will be used when reserving ports.

Specifying Slots and Ports

An asterisk (*) means "all slots and ports".

On a multi-slot system, a single number indicates a slot and all ports on that slot.

On a single-slot system, a single number indicates the port number.

Ports are specified by a slot number, a forward-slash, and a port number. For instance, 2/0 means "slot 2, port 0".

Multiple ports can be specified by space-separated slot/port combinations (2/0 2/1 2/2 2/3), by a hyphen-separated range (2/0-3), or as a comma-separated list (2/0,1,2,3).

Listing Port Status

The commands `port`, `port list`, and `port list *` are all synonymous.

Example

Review the following example to examine the syntax for displaying the status of all ports.

```
username@storm (group:1)% port

SLOT 1 - model: np_ctm_10g, status: ok

0:1 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
1:2 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
2:3 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)
3:4 <|1|> (admin) [1] admin - Test Series g6xcZygPbA
(46.43%)

SLOT 2 - model: np_ctm_1g, status: ok

0:1 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
1:2 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
2:3 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
3:4 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
4:5 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
5:6 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
6:7 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
7:8 <|2|> (admin) [2] admin - Test Series wyT0mSj4wt
(46.30%)
```

The ports for each slot are listed underneath each slot model/status line.

The first number on the line is the port number, the number after the colon (if present) is the logical interface number.

The port's test group is displayed like this: <|1|>. If the link is down, it would look like this: |1| . If the port was not reserved, it would not have a group number or vertical bars, like this: <--->.

The username of the user who reserved the port (if any) is next, in parentheses.

If a test is running on the port, the test ID is next in square brackets, followed by the test name and progress percentage. If no test is running, then the port note (if any) would be shown.

To just show the status for ports 4 through 7 on slot 1:

```
username@storm (group:1)% port list
1/4-7

SLOT 1 - model: np_ctm_1g, status: ok

4:1 <|3|> (admin) [1439] 0-as-bw
(0.38%)
5:2 <|3|> (admin) [1439] 0-as-bw
(0.38%)
6:3 <|3|> (admin) [1439] 0-as-bw
(0.38%)
7:4 <|3|> (admin) [1439] 0-as-bw
(0.38%)
```

Reserving Ports

To reserve all ports on slot 2 (on a multiple slot system):

```
username@storm (group:1)% port
reserve 2
```

To reserve ports 4-7 on slot 2:

```
username@storm (group:1)% port reserve
2/4-7
```

Displaying Port Details

To display the details for slot 1, port 3:

```
username@storm (group:1)% port details
1/3

SLOT 1, PORT 3 (ok, link up)
Auto: false
Auto-Capable: false
Duplex: full
Ignore Pause Frames: false
Media: 10Gbase-SR
MTU: 9198
NP ID: 0
Speed: 10000 Mbit/s
```

Configuring Ports

Port configuration options include: auto, fullduplex, speed, mtu, and ignorepause.

Review the following example to examine the syntax for changing the MTU for slot 1 port 3.

```
username@storm (group:1)% port configure 1/3 mtu
1500
[+] setting mtu for slot 1 port 3 to 1500
username@storm (group:1)% port details 1/3

SLOT 1, PORT 3 (ok, link up)
Auto: false
Auto-Capable: false
Duplex: full
Ignore Pause Frames: false
Media: 10Gbase-SR
MTU: 1500
NP ID: 0
Speed: 10000 Mbit/s
```

Unreserving Ports

Use the following syntax to unreserve all ports on slot 2 (on a multi-slot system).

```
username@storm (group:1)% port
unreserve 2
```

Use the following syntax to unreserve ports 4-7 on slot 2.

```
username@storm (group:1)% port unreserve
2/4-7
```

Setting a Port Note

Review the following example to examine the syntax for setting a port note for slot 2, port 0.

```
username@storm (group:1)% port note 2/0 My port note
[+] setting port note for slot 2 port 0 to 'My port
note'
username@storm (group:1)% port list 2/0

SLOT 2 - model: np_ctm_1g, status: ok

0 <---> NOTE: My port note
```

Displaying a Port Note

Review the following example to examine the syntax for showing a port note (even when a test is running).

```
username@storm (group:1)% port note
2/0
SLOT 2, PORT 0 - NOTE: My port note
```

Removing a Port Note

Review the following example to examine the syntax for removing a port note for slot 2, port 0.

```
username@storm (group:1)% port note 2/0
remove
[+] removing port note for slot 2 port 0
```

The Repeat Command

The `repeat` command allows you to repeat a command.

```
NAME

repeat - repeat a command

SHORTCUT: x

SYNOPSIS

repeat [opts] [cmd]

repeat [<x>] [delay <seconds>] <cmd> - repeat <cmd> <x> times with <seconds>
delay

DESCRIPTION

The delay may also be specified as a range, e.g. 5-15, in which case a
random value in that range will be selected for each repetition.
```

To repeat a command:

```
username@storm (group:1)% repeat puts
hello
[+] command 'puts hello' - iteration 1
of 2
hello
[+] command 'puts hello' - iteration 2
of 2
hello
```

To repeat a command 3 times:

```
username@storm (group:1)% repeat 3 puts
hello
[+] command 'puts hello' - iteration 1 of
3
hello
[+] command 'puts hello' - iteration 2 of
3
hello
[+] command 'puts hello' - iteration 3 of
3
hello
```

To repeat a command 3 times with a two-second delay between iterations:

```
username@storm (group:1)% repeat 3 delay 2 puts
hello
[+] command 'puts hello' - iteration 1 of 3
hello
[+] waiting 2 seconds..
[+] command 'puts hello' - iteration 2 of 3
hello
[+] waiting 2 seconds..
[+] command 'puts hello' - iteration 3 of 3
hello
```

To repeat a command 3 times with 2- to 5- second variable delay between iterations:

```
username@storm (group:1)% repeat 3 delay 2-5 puts
hello
[+] command 'puts hello' - iteration 1 of 3
hello
[+] waiting 2 seconds..
[+] command 'puts hello' - iteration 2 of 3
hello
[+] waiting 4 seconds....
[+] command 'puts hello' - iteration 3 of 3
hello
```

The Report Command

The `report` command allows you to list, export, and open test reports.

```
NAME

report - list, export, and open test reports

SHORTCUT: rep

CONFIGURATION

report_command: open
report_dir: /Users/username/reports
report_format: pdf

SYNOPSIS

report [cmd] [args]

report - synonomous with "report list all"
report export <tid> [format [format] [...]] - export the report with test ID
<tid>
report open <tid> [format [format] [...]] - export and open the report with
test ID <tid>
report list - list up to 25 reports created by any user
report list <user|mine|all> - list up to 25 reports created by <user|mine|all>
report list <user|mine|all> <query> - list all reports matching <query> created
by <user|mine|all>
report list <query> - list all reports with title matching <query>

DESCRIPTION

Valid report formats: bpt csv html pdf rtf xls zip

The commands report, report list, and report list all are synonymous.
```

Configuration Options for the Report Command

The configuration option `report_dir` is the directory where reports will be saved after an export. The default location is $HOME/reports.

The configuration option `report_command` is the command that specifies which application to use to view exported reports. The default value is open.

The `report_format` configuration option indicates the default report format used for export. This can be a space-separated string of formats. The default format is .pdf.

Listing Reports

If the first argument after the `list` subcommand is the word mine, then only reports created by you will be returned.

If the first argument after the `list` subcommand is a valid system username, then only reports created by that user will be returned.

If the first argument after the `list` subcommand is the word all or does not match anything else above, then reports created by all users will be returned.

To list all reports:

```
username@storm (group:1)% report
list
```

To list reports with http in the title:

```
username@storm (group:1)% report list http
[+] showing reports belonging to all users with 'http' in the title
[1077] (admin) "Resiliency_HTTP_Client_2_2" May 6, 2012 2:26:44 AM CDT
(0:01:27.532)
[1076] (admin) "Resiliency_HTTP_Client_1_1" May 6, 2012 2:25:39 AM CDT
(0:01:27.545)
...
```

Exporting Reports

To export the report for test ID 1176:

```
username@storm (group:1)% report export
1176
```

To export the report for test ID 1176 in CSV, PDF, and XLS formats:

```
username@storm (group:1)% report export 1176 csv pdf
xls
```

Opening Reports

To export and open the report for test ID 1176:

```
username@storm (group:1)% report open
1176
```

To export and open the report for test ID 1176 in CSV, PDF, and XLS formats:

```
username@storm (group:1)% report open 1176 csv pdf
xls
```

Note: You cannot automatically open an exported report while running `esh` directly on a BreakingPoint device.

The Shell Command

The `shell` command allows you to execute a local Bash shell or a specific command outside the Enhanced Shell.

```
NAME

shell - invoke a local subshell (or execute
[command])

SHORTCUT: sh

SYNOPSIS

shell [command]
```

Getting a Bash subshell:

```
username@storm (group:1)% shell
[+] invoking subshell; ctrl-D or 'exit' to resume enhanced shell
session
12:12:23 [username@username:~]
$
```

Executing a command in a subshell:

```
username@storm (group:1)% shell ls Downloads/*bps
Downloads/strike-98219-OS2.2.bps Downloads/strike-98221-OS2.1.bps
Downloads/update-66329-98194.bps
username@storm (group:1)%
```

The Strike Command

The `strike` and `strike list` commands are synonymous. They allow you to manage and execute Strikes and Strike Lists.

```
NAME

strike - manage or execute strikes and strike lists

SHORTCUT: s

SYNOPSIS

strike [cmd] [args]

strike - synonomous with "strike info"
strike append <query> to <name> - append strikes matching <query> to strike
list <name>
strike copy <name> to <newname> - copy strike list <name> to <newname>
strike count - count strikes in all strike lists
strike count [query] - count strikes in strike lists matching [query]
strike create <name> using <query> - create/alter strike list <name> using
<params>
strike delete <name> - delete strike list <name>
strike details <name> - lists details for strikes in strike list <name>
strike info [query] - get details for strikes matching [query]
strike list - lists all strike lists
strike list <name> - lists all strike lists matching <name>
strike members <name> - lists strikes in strike list <name>
strike reload - reload strike data from host
strike remove <query> from <name> - remove strikes matching <query> from strike
list <name>
strike rename <name> to <newname> - rename strike list <name> to <newname>
strike run [name] [using <query>] [ep <ep>] [nn <nn>] - execute strikes/strike
list with optional ep/nn

DESCRIPTION

Parameters: c:category, desc:description, dir:direction, i:id, k:keyword,
n:name, p:protocol, r:reference, s:strikepath. All fields are searched by
default.
```

Listing Strike Lists

Use the following syntax to list all Strike Lists.

```
admin@storm (group:1)% strike
list
```

Review the following example to examine the syntax for listing Strike Lists containing a specific word (level, in this example).

```
admin@storm (group:1)% strike list
level
1: Strike Level 1
2: Strike Level 2
3: Strike Level 3
4: Strike Level 4
5: Strike Level 5
```

Counting Strikes in all Strike Lists

Use the following syntax to count the number of Strikes within a Strike List.

```
admin@storm (group:1)% strike
count
```

Counting Strikes in Strike Lists Matching a Pattern

To count Strikes in Strike Lists containing the word level:

```
admin@storm (group:1)% strike count level

STRIKES STRIKE LIST
------- ----------------------------------------
---
183 Strike Level 1
273 Strike Level 2
480 Strike Level 3
1757 Strike Level 4
2402 Strike Level 5
```

Counting Strikes in a Specific Strike List

To count Strikes in a specific Strike List, use the full Strike List name:

```
admin@storm (group:1)% strike count strike level
1

STRIKES STRIKE LIST
------- ----------------------------------------
---
183 Strike Level 1
```

Showing Strikes in a Strike List

To show individual Strikes in a specific Strike List, use the full Strike List name:

```
admin@storm (group:1)% strike members Strike Level 1
[+] showing members of the 'Strike Level 1' strike
list
...
```

Show Details for Strikes in a Strike List

To show detailed information for individual Strikes in a specific Strike List:

```
admin@storm (group:1)% strike members Strike Level 1
[+] showing details for strikes in strike list 'Strike Level
1'...
...
```

Note: This command can take a long time to display results.

Copying a Strike List

To copy the Strike List My Strike List to My New Strike List:

```
admin@storm (group:1)% strike copy My Strike List to My New Strike
List
[+] copying strike list 'My Strike List' to 'My New Strike List'
```

Renaming a Strike List

To rename the Strike List My Strike List to My Old Strike List:

```
admin@storm (group:1)% strike rename My Strike List to My Old Strike
List
[+] renaming strike list 'My Strike List' to 'My Old Strike List'
```

Deleting a Strike List

```
admin@storm (group:1)% strike delete My Old Strike
List
[+] deleting strike list 'My Old Strike List'
```

Searching for Strikes

To search for Strikes, you can specify one or more simple substrings and/or a more complex query involving one or more values for one or more Strike Data fields. lists the available Strike Data fields and their shortcuts.

Data Fields

Data Field Name	Data Field Shortcut
category	c

Data Field Name	Data Field Shortcut
description	desc
direction:	dir
id:	i
keyword	k
name	n
protocol	p
reference	r
strikepath:	s

By default, all fields are searched; so a query of dns would search all fields while a query of category:dns or c:dns would search only the Category field.

To search for Strikes with the word apache in any field:

```
admin@storm (group:1)% strike info
apache
```

To search for Strikes containing both apache and acl in any field:

```
admin@storm (group:1)% strike info apache
acl
```

To search for Strikes containing both apache and acl in any field and xss in the Keyword field:

```
admin@storm (group:1)% strike info apache acl
keyword:xss
```

To search for Strikes containing both apache and acl in any field, xss in the Keyword field, and username or password in the Name field:

```
admin@storm (group:1)% strike info apache acl keyword:xss
name:username,password
```

Creating a Strike List

To create a Strike List using Strikes matching the query terms tippingpoint and dns in any Strike Data field:

```
admin@storm (group:1)% strike create My Strike List using tippingpoint
dns
[+] query returned 3 strikes
[+] adding strikes to strike list 'My Strike List'
[+] saving strike list 'My Strike List'
```

Appending Strikes to a Strike List

To append Strikes matching the category ids to an existing Strike List named My Strike List:

```
admin@storm (group:1)% strike append c:ids to My Strike List
[+] adding strike '/strikes/exploits/ids/ms11_040_tmg_firewall_client_dns_
lookup_overflow.xml' to 'My Strike List'
[+] adding strike '/strikes/exploits/ids/realsecure_icq_sam_1.xml' to 'My
Strike List'
[+] adding strike '/strikes/exploits/ids/realsecure_icq_sam_2.xml' to 'My
Strike List'
[+] adding strike '/strikes/exploits/ids/snort_boping_generic.xml' to 'My
Strike List'
[+] adding strike '/strikes/exploits/ids/snort_boping_xort_1.xml' to 'My Strike
List'
[+] adding strike '/strikes/exploits/ids/snort_boping_xwings_1.xml' to 'My
Strike List'
[+] adding strike '/strikes/exploits/ids/snort_boping_xwings_2.xml' to 'My
Strike List'
[+] adding strike '/strikes/exploits/ids/snort_dcerpc_trirat.xml' to 'My Strike
List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
format_string.xml' to 'My Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
overflow.xml' to 'My Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' to 'My Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_xss.xml'
to 'My Strike List'
[+] added 12 strikes to strike list 'My Strike List'
[+] saving strike list 'My Strike List'
```

To append Strikes matching category:ids and xss to multiple Strike Lists:

```
admin@storm (group:1)% strike list strike list
1: My New Strike List
2: My Other Strike List
3: My Strike List
admin@storm (group:1)% strike append c:ids xss to 1-3
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' to 'My New Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_xss.xml'
to 'My New Strike List'
[+] added 2 strikes to strike list 'My New Strike List'
[+] saving strike list 'My New Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' to 'My Other Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_xss.xml'
to 'My Other Strike List'
[+] added 2 strikes to strike list 'My Other Strike List'
[+] saving strike list 'My Other Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' to 'My Strike List'
[+] adding strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_xss.xml'
to 'My Strike List'
[+] added 2 strikes to strike list 'My Strike List'
[+] saving strike list 'My Strike List'
```

Removing Strikes from a Strike List

To remove Strikes containing the word realsecure from a Strike List name My Strike List:

```
admin@storm (group:1)% strike remove realsecure from My Strike List
[+] removing strike '/strikes/exploits/ids/realsecure_icq_sam_1.xml' from 'My
Strike List'
[+] removing strike '/strikes/exploits/ids/realsecure_icq_sam_2.xml' from 'My
Strike List'
[+] removed 2 strikes from strike list 'My Strike List'
[+] saving strike list 'My Strike List'
```

To remove Strikes matching category:ids and xss from multiple Strike Lists:

```
admin@storm (group:1)% strike list strike list
1: My New Strike List
2: My Other Strike List
3: My Strike List
admin@storm (group:1)% strike remove c:ids xss from 1-3
[+] removing strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' from 'My New Strike List'
[+] removing strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_
xss.xml' from 'My New Strike List'
[+] removed 2 strikes from strike list 'My New Strike List'
[+] saving strike list 'My New Strike List'
[+] removing strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' from 'My Other Strike List'
[+] removing strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_
xss.xml' from 'My Other Strike List'
[+] removed 2 strikes from strike list 'My Other Strike List'
[+] saving strike list 'My Other Strike List'
[+] removing strike '/strikes/exploits/ids/tippingpoint_reverse_dns_lookup_
xss.xml' from 'My Strike List'
[+] removing strike '/strikes/exploits/ids/tippingpoint_telnet_audit_log_
xss.xml' from 'My Strike List'
[+] removed 2 strikes from strike list 'My Strike List'
[+] saving strike list 'My Strike List'
```

Running an Existing Security Test

You can run a security test in several different ways. If the test already exists, simply run it using the `test` command. (See the `test` command for more details and options on using the `test` command).

```
admin@storm (group:1)% test run Security Strike
Level 1
```

Running a Dynamic Security Test

If you have not yet created a security test to run a given Strike List or set of Strikes, you can do so dynamically using the Enhanced Shell.

To create and run a security test using an existing Strike List:

```
admin@storm (group:1)% strike run My Strike
List
```

To create and run a security test using a Strike query that creates a new Strike List:

```
admin@storm (group:1)% strike run My Strike List using apache acl keyword:xss
name:username,password
[+] query returned 2 strikes
[+] adding strikes to strike list 'My Strike List'
[+] saving strike list 'My Strike List'
[+] preparing to run test 'admin - Security tM1KN3ZVBd'
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'Default evasion settings'
using group 1 in async mode
[+] now running test [1185] 'admin - Security tM1KN3ZVBd'
```

To create and run an arbitrarily-named security test using a Strike query:

```
admin@storm (group:1)% strike run apache acl keyword:xss
name:username,password
[+] query returned 2 strikes
[+] adding strikes to strike list 'admin - Strikelist 7Z5p88PtLE'
[+] saving strike list 'admin - Strikelist 7Z5p88PtLE'
[+] preparing to run test 'admin - Security 7Z5p88PtLE'
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'Default evasion settings'
using group 1 in async mode
[+] now running test [1187] 'admin - Security 7Z5p88PtLE'
```

To run a security test series using different Strike Lists:

```
admin@storm (group:1)% strike list strike list
1: My New Strike List
2: My Other Strike List
3: My Strike List
admin@storm (group:1)% strike run 1-3
[+] creating test series 'admin - Test Series E17FLv5DNz'...
[+] creating test using strike list 'My New Strike List'
[+] adding 'admin - Security pbTv5YKLVX' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'Default evasion settings'
[+] creating test using strike list 'My Other Strike List'
[+] adding 'admin - Security R6VTUHoLkk' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'Default evasion settings'
[+] creating test using strike list 'My Strike List'
[+] adding 'admin - Security 1T5s19HbGF' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'Default evasion settings'
[+] beginning test series run...
[+] now running test series [6] 'admin - Test Series
E17FLv5DNz'
```

To run a test series using multiple Strike Lists, Network Neighborhoods, and Evasion Profiles:

```
admin@storm (group:1)% strike list strike list
1: My New Strike List
2: My Other Strike List
3: My Strike List
admin@storm (group:1)% nn list switching
[+] showing network neighborhoods created by all users matching
'switching'
1: BreakingPoint IPv6 Switching
2: BreakingPoint Resiliency Switching
3: BreakingPoint Switching
admin@storm (group:1)% ep list rpc.*segments
[+] showing evasion profiles created by all users matching
'rpc.*segments'
1: RPC: 1-byte TCP segments
2: RPC: 2-byte TCP segments
admin@storm (group:1)% strike run 1 2 nn 3 1 ep *
[+] creating test series 'admin - Test Series I0Nt7G6Xaz'...
[+] creating test using strike list 'My New Strike List'
[+] adding 'admin - Security 8HyZZ1oXPg' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'RPC: 1-byte TCP segments'
[+] adding 'admin - Security htm8HDIQjS' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'RPC: 2-byte TCP segments'
[+] creating test using strike list 'My Other Strike List'
[+] adding 'admin - Security gL3bLTtogq' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'RPC: 1-byte TCP segments'
[+] adding 'admin - Security 3kfICKLHZ3' to test series
with network neighborhood 'BreakingPoint Switching'
and evasion profile 'RPC: 2-byte TCP segments'
[+] creating test using strike list 'My New Strike List'
[+] adding 'admin - Security 2543oS2Jgp' to test series
with network neighborhood 'BreakingPoint IPv6 Switching'
and evasion profile 'RPC: 1-byte TCP segments'
[+] adding 'admin - Security jaXchNreDw' to test series
with network neighborhood 'BreakingPoint IPv6 Switching'
and evasion profile 'RPC: 2-byte TCP segments'
[+] creating test using strike list 'My Other Strike List'
[+] adding 'admin - Security A375g6j04v' to test series
with network neighborhood 'BreakingPoint IPv6 Switching'
and evasion profile 'RPC: 1-byte TCP segments'
[+] adding 'admin - Security k41RFdEADh' to test series
with network neighborhood 'BreakingPoint IPv6 Switching'
and evasion profile 'RPC: 2-byte TCP segments'
[+] beginning test series run...
[+] now running test series [9] 'admin - Test Series I0Nt7G6Xaz'
```

Note: Arbitrarily-named tests and Strike Lists are temporary and will be automatically deleted at some point unless you disable the automatic cleanup feature. For more details on the Cleanup feature, see the section.

The Test Command

The `test` command allows you to list or run tests.

```
NAME
test - list or run tests

SHORTCUT: t

CONFIGURATION

test_dir: /home/username/tests
test_mode: async

SYNOPSIS

test [cmd] [args]

test - synonomous with "test list all"
test cancel - cancel all tests running under your username
test cancel <tid> [...] - cancel the specified test(s) with matching <tid>
test copy <name> to <newname> - copy test <name> to <newname>
test delete <names> - delete tests matching <names>
test import <URL> [as <name>] [force] - import test [as <name>] from <URL>
test import <pattern> [force] - import local tests matching <pattern>
test export <name> - export test named <name>
test export <pattern> - export tests matching <pattern>
test list - list all tests created by any user
test list <user|mine|all> - list all tests created by <user|mine|all>
test list <user|mine|all> <query> - list all tests matching <query> created by
<user|mine|all>
test list <query> - list all tests with title matching <query>
test list local <pattern> - list local tests matching <pattern>
test rename <name> to <newname> - rename test <name> to <newname>
test run <tests> [options] - run test(s)
test run <tests> series - run test(s) as a test series

DESCRIPTION

You can specify tests to run via the numbers shown in the test list. Test
options: mode (async or sync) and nn (network neighborhood). Test "patterns"
should contain an asterisk to indicate multiple files.
```

Syntax

Use the following syntax to display a list of all available tests.

```
username@storm (group:1)%
test
```

Note: The commands `test`, `test list`, and `test list all` are synonymous.

Configuration Options for the Test Command

The configuration option `test_dir` is the directory where tests will be saved after an export. The default location is $HOME/tests.

The `test_mode` configuration option indicates the default mode that tests will run in. The `async` mode will cause the test to run in the background, returning you to a shell prompt and interjecting test results when the test is complete. The `sync` mode will show you a progress indicator, but you will not be able to issue any other shell commands while the test is running. Note that this option can be overridden when a test or test series is executed by using the mode option. (Default: `async`)

Listing Tests

If the first argument after the list subcommand is the word mine, then only tests created by you will be returned.

If the first argument after the list subcommand is a valid system username, then only tests created by that user will be returned.

If the first argument after the list subcommand is the word all or does not match anything else above, then tests created by all users will be returned.

To list all tests:

```
username@storm (group:1)% test
list
```

To list tests with appsim in the title:

```
username@storm (group:1)% test list appsim
[+] showing tests created by all users matching
'appsim'
1: AppSim
2: SimpleEngine - AppSim HTTP 10 Packet
3: SimpleEngine - AppSim HTTP 5 Packet
4: SimpleEngine - AppSim HTTP 6 Packet
5: SimpleEngine - AppSim HTTP 7 Packet
6: SimpleEngine - AppSim HTTP 9 Packet
```

Listing Tests on the Local Filesystem

You may also list tests on the local filesystem contained in the test_dir directory. This is convenient for importing tests.

To list tests on the local filesystem with SNAT in the title:

```
username@storm (group:1)% test list local
SNAT
[+] showing local tests matching 'SNAT'
/Users/username/tests/RTSP_SNAT.bpt
/Users/username/tests/SNAT All Strikes.bpt
[+] 2 results
```

Examining Test Details

To examine details of a particular test, use test info:

```
admin@ati22 (group:2)% test info Security Strike Level 3

TEST: Security Strike Level 3

OPTION VALUE
------------ ------------------------------------------------------
description - Security Level 3 targets all high-risk vulnerabilities,
worms, and backdoors. This includes approximately 500
strikes and usually completes in less than three
minutes.
dut - BreakingPoint Default
name - testClient1
neighborhood - BreakingPoint Switching
seedOverride -

COMPONENT: Security1 (Security Strike Level 3)

OPTION VALUE
----------------------- ------------------------------------------------------
---
attackPlan - Strike Level 3
attackPlanIterationDelay - 0
attackPlanIterations - 1
attackProfile - Default evasion settings
attackRetries - 0
delayStart - 00:00:00
description - A component for running exploits and generating hostile
traffic
maxAttacksPerSecond - 0
maxConcurrAttacks - default
maxPacketsPerSecond - 0
name - Security Strike Level 3
randomSeed - 0
```

Running Tests by Name

To run a test using the case-sensitive test name:

```
admin@storm (group:1)% test run Security Strike Level 1
[+] preparing to run test 'Security Strike Level 1
coGQYA3khW'
using group 1 in async mode
[+] now running test [1217] 'Security Strike Level 1
coGQYA3khW'
```

Running Tests by Number

To run a test using a numbered result (for example, Security Strike Level 3):

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike
level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% test run 3
[+] preparing to run test 'Security Strike Level 3
qxsHD8T3Up'
using group 1 in async mode
[+] now running test [1216] 'Security Strike Level 3
qxsHD8T3Up'
```

To run multiple tests, list their numbered results separated by spaces:

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% test run 1 2 3
[+] creating test series 'admin - Test Series KWcz6VOiOK'...
[+] adding 'Security Strike Level 1' to test series
[+] adding 'Security Strike Level 2' to test series
[+] adding 'Security Strike Level 3' to test series
[+] beginning test series run...
[+] now running test series [11] 'admin - Test Series
KWcz6VOiOK'
```

You may also run a range of multiple tests if the numbered results are sequential:

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% test run 1-3
[+] creating test series 'admin - Test Series ubpL0XLrVd'...
[+] adding 'Security Strike Level 1' to test series
[+] adding 'Security Strike Level 2' to test series
[+] adding 'Security Strike Level 3' to test series
[+] beginning test series run...
[+] now running test series [10] 'admin - Test Series
ubpL0XLrVd'
```

You may also combine one or more individual test result numbers with one or more ranges of test result numbers, in any order:

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% test run 5 1-3
[+] creating test series 'admin - Test Series ePYIeKQBiI'...
[+] adding 'Security Strike Level 5' to test series
[+] adding 'Security Strike Level 1' to test series
[+] adding 'Security Strike Level 2' to test series
[+] adding 'Security Strike Level 3' to test series
[+] beginning test series run...
[+] enabling malware strikes
[+] now running test series [12] 'admin - Test Series
ePYIeKQBiI'
```

Running Tests Synchronously or Asynchronously

Regardless of the `test_mode` configuration setting, you can always force the execution of a test to run synchronously or asynchronously using the mode option anywhere after the test name or test result numbers:

```
admin@storm (group:1)% test run Security Strike Level 1 mode
sync
[+] preparing to run test 'Security Strike Level 1 Ie5slziTZT'
using group 1 in sync mode

Security Strike Level 1 Ie5slziTZT |========== | 27 %
```

Running Tests with a Different Network Neighborhood

By default, tests will run with whichever Network Neighborhoods they were originally configured to use. However, you can change the Network Neighborhood using the `nn` option:

```
admin@storm (group:1)% test run Security Strike Level 1 nn BreakingPoint IPv6
Switching
[+] preparing to run test 'Security Strike Level 1 0mqyA3mpH7'
with network neighborhood 'BreakingPoint IPv6 Switching'
using group 1 in async mode
[+] now running test [1229] 'Security Strike Level 1 0mqyA3mpH7'
```

You can also indicate Network Neighborhoods by number:

```
admin@storm (group:1)% nn list switching
[+] showing network neighborhoods created by all users matching
'switching'
1: BreakingPoint IPv6 Switching
2: BreakingPoint Resiliency Switching
3: BreakingPoint Switching
admin@storm (group:1)% test run Security Strike Level 1 nn 1
[+] preparing to run test 'Security Strike Level 1 0mqyA3mpH7'
with network neighborhood 'BreakingPoint IPv6 Switching'
using group 1 in async mode
[+] now running test [1229] 'Security Strike Level 1 0mqyA3mpH7'
```

Using multiple numbered results also works here, as well:

```
admin@storm (group:1)% test run Security Strike Level 1 nn 3 1
[+] creating test series 'admin - Test Series rJ91JvQkQp'...
[+] adding 'Security Strike Level 1' to test series
with network neighborhood 'BreakingPoint Switching'
[+] adding 'Security Strike Level 1' to test series
with network neighborhood 'BreakingPoint IPv6 Switching'
[+] beginning test series run...
[+] now running test series [13] 'admin - Test Series
rJ91JvQkQp'
```

Forcing Tests to Run as a Test Series

You can force a test run to run as a test series by placing the keyword series at the end of the command:

```
admin@storm (group:1)% test run Security Strike Level 1 series
[+] creating test series 'admin - Test Series K3CRrE3g9j'...
[+] adding 'Security Strike Level 1' to test series
[+] beginning test series run...
[+] now running test series [14] 'admin - Test Series
K3CRrE3g9j'
```

Canceling a Running Test

To cancel a specific test, get the test ID from the output of port list, then issue the test cancel command with that ID. More than one test ID may be specified, separated by spaces.

```
admin@storm (group:1)% port list 1

SLOT 1 - model: np_ctm_10g, status: ok

0:1 <|1|> (admin) [1215] Stack Scrambler 60h zWVaDOFI5n
(0.00%)
1:2 <|1|> (admin) [1215] Stack Scrambler 60h zWVaDOFI5n
(0.00%)
2:3 <|1|> (admin) [1215] Stack Scrambler 60h zWVaDOFI5n
(0.00%)
3:4 <|1|> (admin) [1215] Stack Scrambler 60h zWVaDOFI5n
(0.00%)

admin@storm (group:1)% test cancel 1215
[+] cancelling test: [1215] Stack Scrambler 60h zWVaDOFI5n

[+] test complete: [1215] Stack Scrambler 60h zWVaDOFI5n

Stack Scrambler 122: CANCELED
rxFrames: 1073041
txFrames: 1241739
```

To cancel all of your running tests (for instance, only tests running under your own username):

```
admin@storm (group:1)% test
cancel
```

If your user account is in the admin group, you can also cancel the tests of others using the force option:

```
admin@storm (group:1)% test cancel 1215
force
```

Importing Tests

To import a test from the local filesystem, you may either indicate the full path or just the file name, with or without the file extension (bpt is assumed):

```
admin@storm (group:1)% test import /Users/username/tests/RTSP_
SNAT.bpt
[+] importing /Users/username/tests/RTSP_SNAT.bpt as RTSP_SNAT
upload |=======================================| 100 %
```

To rename a test when you import it:

```
admin@storm (group:1)% test import /Users/username/tests/RTSP_SNAT.bpt as My
New Test
[+] importing /Users/username/tests/RTSP_SNAT.bpt as My New Test
upload |=======================================| 100 %
```

To force a test to be imported, even if there are conflicts, use the force option:

```
admin@storm (group:1)% test import /Users/username/tests/RTSP_SNAT.bpt as My
New Test force
[+] importing /Users/username/tests/RTSP_SNAT.bpt as My New Test (overwrite
forced)
upload |=======================================| 100 %
```

Exporting Tests

To export a test by name:

```
admin@storm (group:1)% test export My Amazing Test
[+] exporting 'My Amazing Test' to '/Users/username/tests/My Amazing
Test.bpt'
```

To export a test by number:

```
admin@storm (group:1)% test list amazing
[+] showing tests created by all users matching 'amazing'
1: My Amazing Test
admin@storm (group:1)% test export 1
[+] exporting 'My Amazing Test' to '/Users/username/tests/My Amazing
Test.bpt'
```

To export multiple tests by number:

```
admin@storm (group:1)% test list appsim
[+] showing tests created by all users matching 'appsim'
1: AppSim
2: SimpleEngine - AppSim HTTP 10 Packet
3: SimpleEngine - AppSim HTTP 5 Packet
4: SimpleEngine - AppSim HTTP 6 Packet
5: SimpleEngine - AppSim HTTP 7 Packet
6: SimpleEngine - AppSim HTTP 9 Packet
admin@storm (group:1)% test export 2-6
[+] exporting 'SimpleEngine - AppSim HTTP 10 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 10 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 5 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 5 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 6 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 6 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 7 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 7 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 9 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 9 Packet.bpt'
```

To export one or more tests by regular expression pattern:

```
admin@storm (group:1)% test export SimpleEngine*
[+] exporting tests created by all users matching 'SimpleEngine.*'
[+] exporting 'SimpleEngine - AppSim HTTP 10 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 10 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 5 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 5 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 6 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 6 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 7 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 7 Packet.bpt'
[+] exporting 'SimpleEngine - AppSim HTTP 9 Packet' to
'/Users/username/tests/SimpleEngine - AppSim HTTP 9 Packet.bpt'
[+] 5 tests exported
```

Copying a Test

To copy a test by name:

```
admin@storm (group:1)% test copy My Amazing Test to Another Amazing
Test
[+] copying test 'My Amazing Test' to 'Another Amazing Test'
```

To copy a test using a numbered result:

```
admin@storm (group:1)% test list amazing
[+] showing tests created by all users matching 'amazing'
1: My Amazing Test
admin@storm (group:1)% test copy 1 to Another Amazing
Test
[+] copying test 'My Amazing Test' to 'Another Amazing
Test'
```

To force a copy operation to overwrite an existing test, use the force keyword:

```
admin@storm (group:1)% test copy My Amazing Test to Another Amazing Test
[!] there is already a test named 'Another Amazing Test'; use 'force' to
overwrite
admin@storm (group:1)% test copy My Amazing Test to Another Amazing Test force
[+] copying test 'My Amazing Test' to 'Another Amazing Test'
```

Renaming a Test

To rename a test by name:

```
admin@storm (group:1)% test rename Terrible Test Name to Great Test
Name
[+] renaming test 'Terrible Test Name' to 'Great Test Name'
```

To rename a test using a numbered result:

```
admin@storm (group:1)% test list terrible
[+] showing tests created by all users matching
'terrible'
1: Terrible Test Name
admin@storm (group:1)% test rename 1 to Great Test Name
[+] renaming test 'Terrible Test Name' to 'Great Test
Name'
```

To force a rename operation to overwrite an existing test, use the force keyword:

```
admin@storm (group:1)% test list terrible
[+] showing tests created by all users matching 'terrible'
1: Terrible Test Name
admin@storm (group:1)% test rename 1 to Great Test Name
[!] there is already a test named 'Great Test Name'; use 'force' to
overwrite
admin@storm (group:1)% test rename 1 to Great Test Name force
[+] renaming test 'Terrible Test Name' to 'Great Test Name'
```

Deleting Tests

To delete a test by name:

```
admin@storm (group:1)% test delete Another Amazing
Test
[+] deleting test 'Another Amazing Test'
```

To delete one or more tests using numbered results:

```
admin@storm (group:1)% test list mine test
[+] showing tests created by admin matching
'test'
1: Another Amazing Test
2: Great Test Name
3: My New Test
admin@storm (group:1)% test delete 1-3
[+] deleting test 'Another Amazing Test'
[+] deleting test 'Great Test Name'
[+] deleting test 'My New Test'
[+] 3 results
```

The Testseries Command

The testseries command allows you to manage test series.

```
NAME

testseries - manage test series

SHORTCUT: ts

SYNOPSIS

testseries [cmd] [args]

testseries - synonomous with "testseries list all"
testseries append <tests> to <name> - append <tests> to test series <name>
testseries copy <name> to <newname> - copy test series <name> to <newname>
testseries create <name> [using <tests>] - create test series <name> using
<tests>
testseries delete <names> - delete test series matching <names>
testseries list - list all test series created by any user
testseries list <user|mine|all> - list all test series created by
<user|mine|all>
testseries list <user|mine|all> <query> - list all test series matching <query>
created by <user|mine|all>
testseries list <query> - list all test series with title matching <query>
testseries members <name> - lists tests in test series <name>
testseries remove <tests> from <name> - remove <tests> from test series <name>
testseries rename <name> to <newname> - rename test series <name> to <newname>
testseries run <testseries> [options] - run test series

DESCRIPTION

You can specify test series to run via the numbers shown in the test series
list. Test series options: mode (async or sync) and nn (network
neighborhood).
```

Listing Test Series

If the first argument after the list subcommand is the word mine, then only test series created by you will be returned.

If the first argument after the list subcommand is a valid system username, then only test series created by that user will be returned.

If the first argument after the `list` subcommand is the word all or does not match anything else above, then test series created by all users will be returned.

To list all test series:

```
admin@storm (group:1)% testseries
list
```

To list test series with http in the title:

```
admin@storm (group:1)% testseries list http
[+] showing test series created by all users matching
'http'
1: BreakingPoint IPS Test - HTTP Attack Evasion
```

Running a Test Series

You can run a test series the same way you would run a test, optionally indicating a mode (`sync` or `async`) and one or more Network Neighborhoods:

```
admin@storm (group:1)% testseries run My Test Series mode sync nn
3 1
```

Note that if you specify more than one test series names (using numbered results) or more than one Network Neighborhood (using numbered results), the testseries will automatically be forced to run in `sync` mode.

Canceling a Running Test Series

You can cancel a test series using `test cancel`:

```
admin@storm (group:2)% port list 1

SLOT 1 - model: np_ctm_10g, status: ok

0:1 <|1|> (admin) [19] BreakingPoint IPS Test - FTP Attack Evasion
(13.85%)
1:2 <|1|> (admin) [19] BreakingPoint IPS Test - FTP Attack Evasion
(13.85%)
2:3 <|1|> (admin) [19] BreakingPoint IPS Test - FTP Attack Evasion
(13.85%)
3:4 <|1|> (admin) [19] BreakingPoint IPS Test - FTP Attack Evasion
(13.85%)

admin@storm (group:2)% test cancel 19
[+] cancelling test: [19] BreakingPoint IPS Test - FTP Attack Evasion

[+] test series complete
BreakingPoint IPS Test - FTP Attack Evasion: CANCELED OVERALL
FTP Strikes No Evasion: CANCELED
FTP Strikes Level 1 Evasion: CANCELED
FTP Strikes Level 2 Evasion: CANCELED
FTP Strikes Level 3 Evasion: CANCELED
FTP Strikes Level 4 Evasion: CANCELED
FTP Strikes Level 5 Evasion: CANCELED
FTP Strikes Level 6 Evasion: CANCELED
```

Creating a Test Series

To create a test series using a test name:

```
admin@storm (group:1)% testseries create My Test Series using Security Strike
Level 1
[+] creating test series 'My Test Series'
[+] adding test 'Security Strike Level 1' to test series
[+] saving test series
```

To create a test series using one or more numbered results instead of a test name:

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% testseries create My Test Series using
1-5
[+] creating test series 'My Test Series'
[+] adding test 'Security Strike Level 1' to test series
[+] adding test 'Security Strike Level 2' to test series
[+] adding test 'Security Strike Level 3' to test series
[+] adding test 'Security Strike Level 4' to test series
[+] adding test 'Security Strike Level 5' to test series
[+] saving test series
```

Adding Tests to a Test Series

To append a test to an existing test series using the test name:

```
admin@storm (group:1)% testseries append Security All Strikes to My Test
Series
[+] adding test 'Security All Strikes' to test series 'My Test Series'
[+] added 1 test to test series 'My Test Series'
[+] saving test series 'My Test Series'
```

To append one or more tests to an existing test series using numbered results instead of test names:

```
admin@storm (group:1)% test list all all strikes
[+] showing tests created by all users matching 'all strikes'
1: Security All Strikes
2: Security All Strikes 1-byte TCP Segments, duplicate last packet
3: Security All Strikes 1-byte TCP Segments, interleaved duplicate segments
with invalid TCP checksums
4: Security All Strikes 1-byte TCP Segments, interleaved duplicate segments
with null TCP control flags
5: Security All Strikes 1-byte TCP Segments, interleaved duplicate segments
with out-of-window sequence numbers
6: Security All Strikes 1-byte TCP Segments, interleaved duplicate segments
with requests to resync sequence numbers mid-stream
7: Security All Strikes 1-byte TCP Segments, Ordered
8: Security All Strikes 1-byte TCP Segments, Out of order 1 byte segments
9: Security All Strikes 16-byte IP Fragments, Favor New
10: Security All Strikes 16-byte IP Fragments, Favor Old
11: Security All Strikes 24-byte IP Fragments, Ordered
12: Security All Strikes 8-byte IP Fragments, Ordered
13: Security All Strikes 8-byte IP Fragments, Random Order
14: Security All Strikes 8-byte IP Fragments, Reverse Order
admin@storm (group:1)% testseries append 1-3 to My Test Series
[+] adding test 'Security All Strikes' to test series 'My Test Series'
[+] adding test 'Security All Strikes 1-byte TCP Segments, duplicate last
packet' to test series 'My Test Series'
[+] adding test 'Security All Strikes 1-byte TCP Segments, interleaved
duplicate segments with invalid TCP checksums' to test series 'My Test Series'
[+] added 3 tests to test series 'My Test Series'
[+] saving test series 'My Test Series'
```

Listing Tests in a Test Series

To show individual tests in a specific test series using the name of the test series:

```
admin@storm (group:1)% testseries members BreakingPoint IPS Test - FTP Attack
Evasion
[+] showing tests in the 'BreakingPoint IPS Test - FTP Attack Evasion' test
series
1: FTP Strikes No Evasion
2: FTP Strikes Level 1 Evasion
3: FTP Strikes Level 2 Evasion
4: FTP Strikes Level 3 Evasion
5: FTP Strikes Level 4 Evasion
6: FTP Strikes Level 5 Evasion
7: FTP Strikes Level 6 Evasion
```

To show individual tests in a specific test series using a numbered result instead of the name of the test series:

```
admin@storm (group:1)% testseries list ftp
[+] showing test series created by all users matching 'ftp'
1: BreakingPoint IPS Test - FTP Attack Evasion
admin@storm (group:1)% testseries members 1
[+] showing tests in the 'BreakingPoint IPS Test - FTP Attack Evasion' test
series
1: FTP Strikes No Evasion
2: FTP Strikes Level 1 Evasion
3: FTP Strikes Level 2 Evasion
4: FTP Strikes Level 3 Evasion
5: FTP Strikes Level 4 Evasion
6: FTP Strikes Level 5 Evasion
7: FTP Strikes Level 6 Evasion
```

Removing Tests from a Test Series

To remove a test from a test series:

```
admin@storm (group:1)% test list strike level
[+] showing tests created by all users matching 'strike level'
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% testseries create junk using 1-5
[+] creating test series 'junk'
[+] adding test 'Security Strike Level 1' to test series
[+] adding test 'Security Strike Level 2' to test series
[+] adding test 'Security Strike Level 3' to test series
[+] adding test 'Security Strike Level 4' to test series
[+] adding test 'Security Strike Level 5' to test series
[+] saving test series
admin@storm (group:1)% testseries copy junk to other junk
[+] copying test series 'junk' to 'other junk'
admin@storm (group:1)% testseries
[+] showing test series created by all users
1: BreakingPoint IPS Test - Attack Evasion IP Fragmentation
2: BreakingPoint IPS Test - Attack Evasion TCP Segmentation
3: BreakingPoint IPS Test - FTP Attack Evasion
4: BreakingPoint IPS Test - HTTP Attack Evasion
5: BreakingPoint IPS Test - RPC Attack Evasion
6: junk
7: other junk
admin@storm (group:1)% testseries members 6
[+] showing tests in the 'junk' test series
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
4: Security Strike Level 4
5: Security Strike Level 5
admin@storm (group:1)% testseries remove 4-5 from 6 7
[+] removing test 'Security Strike Level 5' from test series 'junk' position 5
[+] removing test 'Security Strike Level 4' from test series 'junk' position 4
[+] removed 2 tests from test series 'junk'
[+] saving test series 'junk'
[+] removing test 'Security Strike Level 5' from test series 'other junk'
position 5
[+] removing test 'Security Strike Level 4' from test series 'other junk'
position 4
[+] removed 2 tests from test series 'other junk'
[+] saving test series 'other junk'
admin@storm (group:1)% testseries members 6
[+] showing tests in the 'junk' test series
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
admin@storm (group:1)% testseries members 7
[+] showing tests in the 'other junk' test series
1: Security Strike Level 1
2: Security Strike Level 2
3: Security Strike Level 3
admin@storm (group:1)% testseries remove Security Strike Level 3 from junk
[+] removing test 'Security Strike Level 3' from test series 'junk' position 3
[+] removed 1 test from test series 'junk'
[+] saving test series 'junk'
admin@storm (group:1)% testseries members junk
```

Copying a Test Series

To copy a test series:

```
admin@storm (group:1)% testseries list
[+] showing test series created by all users
1: BreakingPoint IPS Test - Attack Evasion IP Fragmentation
2: BreakingPoint IPS Test - Attack Evasion TCP Segmentation
3: BreakingPoint IPS Test - FTP Attack Evasion
4: BreakingPoint IPS Test - HTTP Attack Evasion
5: BreakingPoint IPS Test - RPC Attack Evasion
6: My Test Series
admin@storm (group:1)% testseries copy My Test Series to My New Test
Series
[+] copying test series 'My Test Series' to 'My New Test Series'
admin@storm (group:1)% testseries copy 6 to My Other New Test Series
[+] copying test series 'My Test Series' to 'My Other New Test Series'
```

Renaming a Test Series

To rename a test series:

```
admin@storm (group:1)% testseries list
[+] showing test series created by all users
1: BreakingPoint IPS Test - Attack Evasion IP Fragmentation
2: BreakingPoint IPS Test - Attack Evasion TCP Segmentation
3: BreakingPoint IPS Test - FTP Attack Evasion
4: BreakingPoint IPS Test - HTTP Attack Evasion
5: BreakingPoint IPS Test - RPC Attack Evasion
6: My New Test Series
7: My Other New Test Series
8: My Test Series
admin@storm (group:1)% testseries rename My Test Series to My Old Test
Series
[+] renaming test series 'My Test Series' to 'My Old Test Series'
admin@storm (group:1)% testseries rename 7 to My Amazing Test Series
[+] renaming test series 'My Other New Test Series' to 'My Amazing Test
Series'
```

Deleting a Test Series

To delete a test series:

```
admin@storm (group:1)% testseries list
[+] showing test series created by all users
1: BreakingPoint IPS Test - Attack Evasion IP
Fragmentation
2: BreakingPoint IPS Test - Attack Evasion TCP
Segmentation
3: BreakingPoint IPS Test - FTP Attack Evasion
4: BreakingPoint IPS Test - HTTP Attack Evasion
5: BreakingPoint IPS Test - RPC Attack Evasion
6: My Amazing Test Series
7: My New Test Series
8: My Old Test Series
admin@storm (group:1)% testseries delete My Old Test
Series
[+] deleting test series 'My Old Test Series'
admin@storm (group:1)% testseries delete 6 7
[+] deleting test series 'My Amazing Test Series'
[+] deleting test series 'My New Test Series'
[+] 2 results
```

The User Command

The `user` command allows you to add, delete, or list users.

```
NAME

user - add, delete, or list users

SHORTCUT: u

SYNOPSIS

user [cmd] [args]

user - synonomous with "user list"
user add <userid> [<opt> <val> ...] - create user <userid> with
options
user delete <userid> - delete user <userid>
user list [userid] - list one or more users
user modify <userid> <option> <value> - modify the settings for
<userid>

DESCRIPTION

Valid options: name, email, password, group
```

The commands `user` and `user list` are synonymous.

Listing User Accounts

To list user accounts:

```
username@storm (group:1)% user list

USER NAME EMAIL GROUP
----- ----- ---------------------- -----
admin Admin admin@example.com admin
username username username@example.com
user
```

Listing Specific User Accounts

To list specific user accounts:

```
username@storm (group:1)% user list
username

USER NAME EMAIL GROUP
----- ----- ---------------------- -----
username username username@example.com
user
```

Adding a New User Account

Note: You must be an administrator to add a new user account.

To add and verify a new user account:

```
admin@storm (group:1)% user add jdoe
[+] adding user 'jdoe'
admin@storm (group:1)% user list jdoe

USER NAME EMAIL GROUP
----- ----- ---------------------- --
---
jdoe jdoe jdoe@example.com user
```

Adding a New User Account with Details

To add a new user account and provide a user name, email address, password, and group:

```
admin@storm (group:1)% user add jdoe name Jane Doe email jdoe@example.com
password abc123 group admin
[+] adding user 'jdoe'
admin@storm (group:1)% user list jdoe

USER NAME EMAIL GROUP
----- -------- ---------------------- -----
jdoe Jane Doe jdoe@example.com admin
```

Note: Only the options that you want to set need to be provided.

Modifying an Existing User Account

Note: You must be an administrator to modify a user account.

To modify one (or more) user account details:

```
admin@storm (group:1)% user list jdoe

USER NAME EMAIL GROUP
----- -------- ---------------------- -----
jdoe Jane Doe jdoe@example.com admin

admin@storm (group:1)% user modify jdoe group
user
[+] setting new group value for 'jdoe'
admin@storm (group:1)% user list jdoe

USER NAME EMAIL GROUP
----- -------- ---------------------- -----
jdoe Jane Doe jdoe@example.com user
```

Deleting a User Account

Note: You must be an administrator to delete a user account.

To delete a user account:

```
admin@storm (group:1)% user delete
jdoe
[+] deleting user 'jdoe'
```

Modules

The Enhanced Shell is extensible and uses "modules" that provide additional functionality. Modules can be installed manually or via a local or remote (HTTP-accessible) module repository.

Repositories

By default, the repository your Enhanced Shell installation will use is the same as the source you installed the original files from. For example, if you performed a local install from `/Users/username/src/esh`, then the repository definition in `/Users/username/.bpsh/etc/repo.conf` would look like this:

```
set bps_repositories {
local {
description "Local File Repository"
method local
url
"file:///Users/username/src/esh/lib/modules"
}
}
```

A remote repository would have an HTTP URL instead of a file-based URL, but is otherwise quite similar.

Example

```
set bps_repositories {
strikecenter.ixiacom.com {
description "BreakingPoint Strike Center"
method remote
url
"https://strikecenter.ixiacom.com/esh/lib/modules"
}
}
```

To manually add another repository:

```
set bps_repositories {
strikecenter.ixiacom.com {
description "BreakingPoint Strike Center"
method remote
url
"https://strikecenter.ixiacom.com/esh/lib/modules"
}
myrepo {
description "My Very Own Repository"
method remote
url "http://server.example.com/esh/lib/modules"
}
}
```

Note: There are currently no mechanisms within the Enhanced Shell itself to add, view, or otherwise manipulate repository configurations.

Example Module Code

In a file named `helloworld.tcl`, located in `lib/modules` within a module repository:

```
# description: Hello World Module

# this is the command (or list of commands) to be eval'ed after initial module
installation
set bps_postinstall InstallHelloWorldResources

# this is the command (or list of command) to be eval'ed before module
uninstallation
AddPreUninstallCommand helloworld UninstallHelloWorldResources

# set one or more config variables for the module here
dict set bps_config_defaults helloworld_foo "bar"

#
# helloworld
#

proc helloworld {args} {

#| shortcut {
#| alias hw
#| command {helloworld {*}$args}
#| }
#| summary {
#| args {[say <something>]}
#| description {say hello world or whatever}
#| }
#| invocations {
#| {
#| args {}
#| description {synonymous with "helloworld say hello"}
#| }
#| {
#| args {say <something>}
#| description {say whatever}
#| }
#| }
#| verbose {
#| Give long-winded explanation that no one will read here.
#| }

# print our config variable to show it works
set hw_var [GetConfigOption helloworld_foo]
puts "config variable 'helloworld_foo' has a value of: $hw_var"

set command [lindex $args 0]
set arguments [lrange $args 1 end]

if { $command eq {} } { set command "say" }
if { $arguments eq {} } { set arguments "hello" }

switch $command {
"s" -
"say" { SayStuff {*}$arguments }
default { ShowVerboseHelp helloworld }
}
```

Example Module Code (continued)

```
return
}

#
# SayStuff
#

proc SayStuff {args} {
puts "you can haz $args"
}

#
# InstallHelloWorldResources
#

proc InstallHelloWorldResources {} {
# if you have one or more external script(s) to install,
# you could use a proc like this to do all your installations

# InstallResources expects each resource to be relative to
lib/modules/resources/helloworld/
#
# first arg: the name of this module (usually)
#
# each pair of arguments after that:
# first arg in pair: the install directory relative to $HOME/.bpsh/
# second arg in pair: the path/filename of the resource to install
InstallResources helloworld bin helloworld
}

#
# UninstallHelloWorldResources
#

proc UninstallHelloWorldResources {} {
# if you install one or more external script(s),
# you should be courteous enough to provide
# a way to remove them

# uses the same arguments as InstallResources
UninstallResources helloworld bin helloworld
}
```

This page intentionally left blank.

CHAPTER 25 BPS Robot Framework

Robot Framework is a generic test automation framework for acceptance testing and acceptance test-driven development (ATDD). It has an easy-to-use tabular test data syntax and it utilizes the keyword-driven testing approach.

BreakingPoint allows you to use Robot APIs to configure and execute tests and check the test results. You can leverage these Robot APIs to integrate and run BreakingPoint tests from your own Robot Framework.

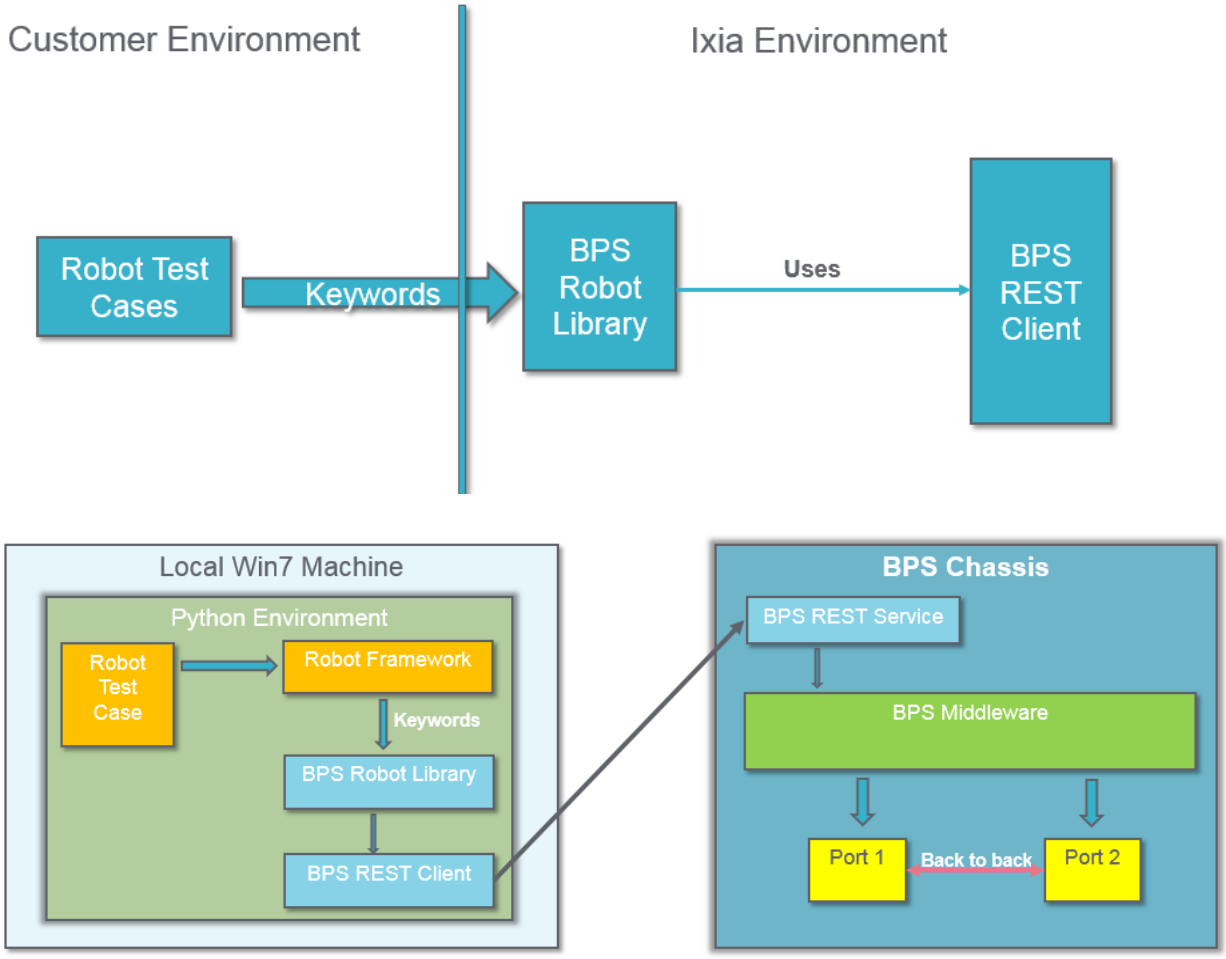
Instructions for installing a Robot Framework can be found at: http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#installation-instructions.

For information about the Robot Framework preconditions (Python environment and pip) please see:

- http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#preconditions.
- BPS/Robot Framework and Example Setup
- BPS Robot Framework Functionality and Keywords
- Example Test Cases
  - Upload and Run Test and Download Report
  - Run Test and Poll for Statistics

High Level Overview and Example Setup

The following diagrams provide an overview of the BPS Robot Framework and example test setup.



BPS Robot Framework Functionality and Keywords

The following tables describe BPS Robot Framework functionality and keywords.

Functionality	Keyword	Description
User Authentication	Login	Login user to chassis
	Logout	Logout user from chassis
Config Handling	Upload Config	Uploads given config file to chassis
	Download Config	Downloads given config from chassis
Ports Handling	Reserve Ports	Reserves the given port list
	Unreserve Ports	Un-reserves the given ports

Functionality	Keyword	Description
Run / Stop Tests	Run Test	Runs the test with the given name
	Stop Test	Stops the given test

Functionality	Keyword	Description
Statistics	Get Test Progress	Get progress for running test
	Get Test Result	Get result for given test
	Download Report	Downloads report for given test
	Check Test Result	Checks if the BPS test has passed or failed
	Add Stat Watch	Add statistic to watched stats list
	Remove Stat Watch	Remove statistic from watched stats list
	Run Test	Runs the test with the given name
	Watch Stats	Gets the values for the watched stats

Example Test Cases

The following test cases display the Robot Framework test flow followed by an example of the code required to implement the test.

Upload and Run Test and Download Report

==========Contents of: upload_run_test_malware.robot==========

*** Settings ***

Documentation Example test cases using the keyword-driven testing approach.

All tests contain a workflow constructed from keywords in

... ``BPSRobotLibrary.py``.

Library BuiltIn

Library BpsRobotLibrary.py

*** Variables ***

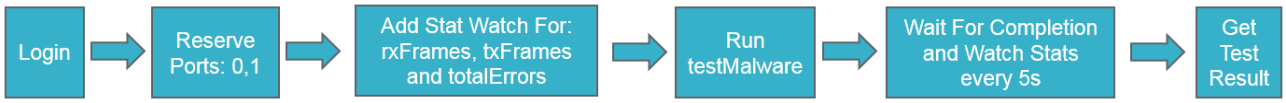
```
${chassis} bps_unit1
${username} admin
${password} admin
*** Test Cases ***
Upload And Run Test And Download Report
######################
# Login #
######################
Log To Console Login
Login chassis=${chassis} username=${username} password=${password}

######################
# Upload Test #
######################
Log To Console Upload Test
${filename} Set Variable testMalware.bpt
${result}= Upload Config filename=${Filename} force=true
Log To Console ${result}

######################
# Reserve Ports #
######################
Log To Console Reserve Ports
${slot} Set Variable 2
@{ports}= Create List 0 1
Reserve Ports slot=${slot} portlist=@{ports} force=true

######################
# Run Test #
######################
Log To Console Run Test
```

```
${testname} Set Variable testMalware
${testid}= Run Test testname=${testname}
Log To Console ${testid}
######################
# Get Progress #
######################
Log To Console Get Progress
:FOR ${i} IN RANGE 999999
\ ${progress}= Get Test Progress testid=${testid}
\ Log To Console ${progress}
\ Sleep 5s
\ Exit For Loop If ${progress} == 100
Log To Console Test Finished!
######################
# Get Result #
######################
Log To Console Get Result
${result}= Get Test Result testid=${testid}
Log To Console ${result}

######################
# Get Test Report #
######################
Log To Console Get Test Report
${reportname} Set Variable testMalwareReport.pdf
${location} Set Variable ${CURDIR}/Reports
Download Report testid=${testid} reportname=${reportname} location=${location}

######################
# Check Test Result #
######################
```

```
Log To Console Check Test Result
Check Test Result testid=${testid}
Log To Console ${result}

Run And Stop Test
######################
# Login #
######################
Log To Console Login
Login chassis=${chassis} username=${username} password=${password}

######################
# Reserve Ports #
######################
Log To Console Reserve Ports
${slot} Set Variable 1
@{ports}= Create List 0 1
Reserve Ports slot=${slot} portlist=@{ports} force=true

######################
# Run Test #
######################
Log To Console Run Test
${testname} Set Variable testMalware
${testid}= Run Test testname=${testname}
Log To Console ${testid}
Sleep 5s
######################
# Stop Test #
######################
Log To Console Stop Test
```

Stop Test testid=${testid}

Run Test and Poll for Statistics



==========Contents of: run_test_poll_statistics.robot==========

*** Settings ***

Documentation Example test cases using the keyword-driven testing approach.

...

... All tests contain a workflow constructed from keywords in

... ``BPSRobotLibrary.py``.

Library BuiltIn

Library BpsRobotLibrary.py

*** Variables ***

${chassis} bps_unit1

${username} admin

${password} admin

*** Keywords***

Get Watched Stats

[Arguments] ${id}

${watchStats}= Watch Stats testid=${id}

Log To Console ${watchStats}

*** Test Cases ***

Run Test And Poll For Stats

######################

Login

######################

Log To Console Login

Login chassis=${chassis} username=${username} password=${password}

```
######################
# Reserve Ports #
######################
Log To Console Reserve Ports
${slot} Set Variable 2
@{ports}= Create List 0 1
Reserve Ports slot=${slot} portlist=@{ports} force=true
######################
# Add Stat Watch #
######################
Log To Console Stat Watch
${rxFrames} Set Variable ethRxFrames
${txFrames} Set Variable ethTxFrames
${totalErr} Set Variable totalErrored
Add Stat Watch statname=${rxFrames}
Add Stat Watch statname=${txFrames}
Add Stat Watch statname=${totalErr}

######################
# Run Test #
######################
Log To Console Run Test
${testname} Set Variable testMalware
${testid}= Run Test testname=${testname}
Log To Console ${testid}

######################
# Get Stats #
######################
Log To Console Get Stats
:FOR ${i} IN RANGE 999999
```

```
\ ${progress}= Get Test Progress testid=${testid}
\ Log To Console Progress = ${progress}
\ Run Keyword If ${progress}>0 Get Watched Stats ${testid}
\ Sleep 5s
\ Exit For Loop If ${progress} == 100
Log To Console Test Finished!
######################
# Get Result #
######################
Log To Console Get Result
Get Test Result testid=${testid}
```

This page intentionally left blank.

CHAPTER 26 Appendix

This section includes:

Hardware Specifications

The table below details the hardware specifications for the BreakingPoint Storm.

BreakingPoint Storm Hardware Specifications

Hardware Component	Specification
Model	
Dimensions	Height: 5.25 inches (13.3 cm) Width: 17.5 inches (44.4 cm) Depth: 22.4 inches (56.9 cm) Weight: 40 lbs (18.2 kg)
Dual Media Test Interfaces	4 - 10 Gigabit SX/LX fiber-optic ports
Target Control Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
BPS Management Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
Power Requirements	100-240 VAC 4 A at 50-60 Hz Maximum power consumption: 400 Watts

Hardware Component	Specification
Temperature Requirements	Operating: 0° C to 40° C (32° F to 104° F) Non-operating: -20° C to 70° C (-4° F to 158° F)
Humidity Requirements	Humidity: 5% to 95% relative humidity, non-condensing
Altitude Requirements	No degradation up to 13,000 feet
Hardware Accessories	4 - Multi-mode fiber-optic cables 2 - AC cables 2 - DB9 serial cables 4 - XFP 850 nm optical transceivers 2 - RJ-45 Ethernet cables 1 - Front-side AC adapter (International use only)

The table below details the hardware specifications for the BreakingPoint FireStorm.

BreakingPoint FireStorm Hardware Specifications

Hardware Component	Specification
Model	BreakingPoint FireStorm
Dimensions	Height: 7 inches (17.8 cm) Width: 17.4 inches (44.2 cm) Depth: 19.5 inches (49.8 cm) Shipping Weight: 45 lbs (20.4kg) Rack Units: 4 RU
Dual Media Test Interfaces	4 - 10Gb/1Gb Ethernet ports
Target Control Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
BPS Management Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
Power Requirements	100-240 V, 50/60 Hz Maximum power consumption: 1,800 Watts
Temperature Requirements	Operating: 15° C to 35° C (59° F to 95° F) Non-operating: -20° C to 70° C (-4° F to 158° F)
Humidity Requirements	Humidity: 5% to 95% relative humidity, non-condensing
Altitude Requirements	No degradation up to 13,000 feet

The table below details the hardware specifications for the BreakingPoint FireStorm ONE.

BreakingPoint FireStorm ONE Hardware Specifications

Hardware Component	Specification
Model	BreakingPoint FireStorm ONE
Dimensions	Height: 2 inches (5.08 cm) Width: 17.5 inches (44.45 cm) Depth: 25.5 inches (64.77 cm) Shipping Weight: 18.6 lbs (8.5 kg) Rack Units: 1 RU
Dual Media Test Interfaces	4 - 10Gb/1Gb Ethernet ports
Target Control Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
BPS Management Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
Power Requirements	100-240 VAC, 50/60 Hz, 7A max
Wattage	640 W
Temperature Requirements	Operating: 15° C to 35° C (59° F to 95° F) Non-operating: -20° C to 70° C (-4° F to 158° F)
Humidity Requirements	Humidity: 5% to 95% relative humidity, non-condensing
Altitude Requirements	No degradation up to 13,000 feet

The table below details the hardware specifications for the BreakingPoint 20.

BreakingPoint 20 Hardware Specifications

Hardware Component	Specification
Model	
Dimensions	Height: 5.25 inches (13.3 cm) Width: 17.5 inches (44.4 cm) Depth: 22.4 inches (56.9 cm) Shipping Weight: 40 lbs (18.2 kg) Rack Units: 4 RU
Dual Media Test Interfaces	4 - 10GigE/1GigE SFP+ Ethernet ports

Hardware Component	Specification
BPS Management Ports	1 - 10/100/1000 Ethernet interface 1 - DB9 serial interface
Power Requirements	100-240 VAC 4 A at 50-60 Hz Maximum power consumption: 1,800 Watts
Temperature Requirements	Operating: 15° C to 35° C (59° F to 95° F) Non-operating: -20° C to 70° C (-4° F to 158° F)
Humidity Requirements	Humidity: 5% to 95% relative humidity, non-condensing
Altitude Requirements	No degradation up to 13,000 feet
Hardware Accessories	4 - Multi-mode fiber-optic cables 2 - AC cables 2 - DB9 serial cables 4 - XFP 850 nm optical transceivers 2 - RJ-45 Ethernet cables 1 - Front-side AC adapter (International use only)

The table below details the hardware specifications for the Ixia BreakingPoint PerfectStorm.

Ixia BreakingPoint PerfectStorm Hardware Specifications

Hardware Component	Specification
Model	
Dimensions	Height: 19.21 inches (48.79 cm) Width: 19 inches (48.26 cm) Depth: 27.2 inches (69.09 cm) Shipping Weight: 40 lbs (18.14 kg) Rack Units: 11RU
Load Modules	8 – 10GigE SFP+ Ethernet ports 2 – 40GigE QSFP+ Ethernet ports
BPS Management Ports	2 - RJ-45 10/100/1000Mbps Gigabit Ethernet ports
Power Requirements	3 – Single phase, 200-240 VAC, 50-60 Hz circuits 3 – 20A circuit breakers (one for each circuit)
Temperature Requirements	Operating: 5° C to 40° C (41° F to 104° F) Storage: 5° C to 50° C (41° F to 122° F)

Hardware Component	Specification
Humidity Requirements	Humidity: 0% to 85% relative humidity, non-condensing

Load Module Specifications and Supported Transceivers

See the following load module data sheets for detailed specifications such as supported transceivers and cables.

Description
CloudStorm 100GE Application and Security Test Load Module
CloudStorm 25GE Application and Security Test Load Module
CloudStorm 10GE/40GE Application and Security Test Load Module
PerfectStorm 100GE 1-Port Load Module
PerfectStorm 40GE, High-Performance Application and Security Load Modules
PerfectStorm 10GE, High-Performance Application and Security Load Modules

Software Specifications

The table below details the software specifications for the BreakingPoint system.

Software Specifications

Software Component	Specification
Telnet Client	Telnet client running VT100 emulation
Serial Client	Serial client running 115200/8/n/l/none

See the Control Center Overview on page 44 for supported browser information.

Light-Emitting Diodes

The light-emitting diodes (LEDs) status indicators are located on the front of the chassis. See the following table for descriptions of each LED and what each LED color represents.

LED Statuses

LED	Color	Status	Description
Status LED	Amber	Boot-up	The system is booting up.
	Green	Operational	The system is powered on and operational.

LED	Color	Status	Description
Power LED	Green	Powered	The power is on.
	Off	Off	The power is off.
	Blinking (Red/Green)	Busy	Insufficient power source to power on multiple FireStorm units. Disconnect one FireStorm or upgrade power source.
Data Port Activity LED (bottom)	Blinking Green	Passing traffic	The port is passing traffic.
	Off	No traffic is passing through	The port is not passing traffic.
Data Port Link LED (top)	Green	Connected	The port is connected at 10 Gbps (10Gb system) or 1 Gbps (1Gb system) and ready to send data.
	Off	Disconnected	The port is not ready to send data.
	Amber	Connected	This color indicates a link fault.

CLI Commands

The following table lists the CLI commands available for the BPS Management port.

CLI Commands

Command	Description	Sample Syntax
?	Print a list of commands	`?`
? <cmd>	Print help for a command	`? addUser`
addUser	Add a user to the system	`addUser Joe Smith -name Joe -email joe@email.com`
	Add a user to a group	`addUser Joe Smith -name Joe -email joe@email.com -group admin`
exit	Exit the shell	`exit`
help	Print the list of commands with descriptions	`help`
help <cmd>	Print help for a command	`help addUser`

Command	Description	Sample Syntax
networkInfo	Retrieve network setup information	`networkInfo`
passwd	Change the password for the account logged into the BPS Management port	`passwd`
reboot	Reboot the system	`reboot`
removeUser	Delete a user account	`removeUser Joe`
updateNetwork	Configure a network interface	`updateNetwork -dhcp yes -hostname test.ixiacom.int -ip 10.10.10.123 -netmask 24 -gw 10.10.10.1`
updateNetwork	Switch ports on and off	`updateNetwork -http_off value HTTP Off: <true> turns port 80 OFF`
updateUser	Modify a user account	`updateUser joe -name Joseph Smith -email joeS@email.com`
uptime	Display the system's uptime	`uptime`
userInfo	Query a user's information	`userInfo joe`
version	Display the firmware version	`version`

Global Scripts Templates

Global scripts allow you do things like reboot your device, monitor DUT statistics, and create VLANs via firmware control. The following tables list the global scripts for available device types.

Dell PowerConnect 6024

The following tablelists the global scripts for the Dell PowerConnect 6024 device type.

Dell PowerConnect 6024 Global Scripts Templates

Script	Template
VLAN Trunk Create	Expect > Send enable\r Expect # Send conf \r Expect # Send vlan database\r Expect # Send vlan 1-12\r Expect # Send exit\r Expect # Send interface eth g2\r Expect # Send switchport mode trunk\r Expect # Send switchport trunkallowed vlan add 1-12\r Expect # Send exit\r Send exit\r
VLAN Create	Expect> Send enable\r Expect # Send conf \r Expect # Send vlan database\r Expect # Send vlan 1-12\r Expect # Send exit\r Expect # Send exit\r Expect #

Script	Template
VLAN Delete	Expect > Send enable\r Expect # Send conf \r Expect # Send vlan database\r Expect # Send no vlan 1-12\r Expect # Send exit\r Expect # Send exit\r Expect #

Extreme Summit 7i

The following tablelists the global scripts for the Extreme Summit 7i device type.

Extreme Summit 7i Global Scripts Templates

Script	Template
VLAN Create	Send amdin\r Expect password: Send password\r Expect # Send create vlan test\r Expect # Send configure vlan test ipaddress 192.168.1.1/16\r Expect # Send exit\r Expect # Send exit\r Expect #

Script	Template
VLAN Delete	Send amdin\r Expect password: Send password\r Expect # Send delete vlan test\r Expect # Send exit\r Expect # Send exit\r Expect #
Trunk Create	Send amdin\r Expect password: Send password\r Expect # Send config dot1q ethertype 9100\r Expect # Send config jumbo-frame size 1530\r Expect # Send config vlan test tag 50\r Expect # Send config vlan test add port 1-4 untag\r Expect # Send config vlan test add port 31,32 tagged\r Expect # Send exit\r Expect # Send exit\r Expect #

HP ProCurve 7500yl

The following tablelists the global commands available for the HP ProCurve 7500yl device type.

HP ProCurve 7500yl Global Scripts Templates

Script	Template
VLAN Delete	Send r\r Expect Password: Send password\r Expect # Send config t\r Expect # Send no vlan 2\r Expect # Send exit\r Expect # Send exit\r Expect #
VLAN Create	Send r\r Expect Password: Send password\r Expect # Send config t\r Expect # Send vlan 2\r Expect # Send exit\r Expect # Send exit\r Expect #

Third Party Licenses

BreakingPoint contains software that is licensed to Ixia by third parties.

To review the list of third party components, see the BreakingPoint Third Party Software Licenses file which can be found in the same location where you download BreakingPoint software - https://support.ixiacom.com > **Software Downloads** > **BreakingPoint Software**. Note that you will need to log in to access this section of the support website.

This page intentionally left blank.

INDEX

D

H

S

Ixia, a Keysight Business
26601 West Agoura Road
Calabasas, California 91302