
Chapter 2

Network Security

ServerIron software contains many intrusion detection and prevention capabilities. The ServerIron can be configured to defend against a variety of TCP SYN attacks, Denial of Service (DoS) attacks, and Smurf attacks.

TCP SYN Attacks

TCP SYN attacks disrupt normal traffic flow by exploiting the way TCP connections are established.

When a normal TCP connection occurs, the connecting host first sends a TCP SYN packet to the destination host. The destination host (actually the ServerIron, acting as an intermediary between the source and destination hosts) responds with a SYN ACK packet. The connecting host then returns an ACK packet. This process, known as a "TCP three-way handshake", establishes the TCP connection.

A TCP SYN attack floods a host with TCP SYN packets. For each of these TCP SYN packets, the ServerIron responds with a SYN ACK packet and adds an entry to its session table. However, no ACK packet is actually sent back, so the connection is incomplete. If the attacker sends enough TCP SYN packets, the session table fills up with incomplete connections, and service can be denied to legitimate TCP connections.

syn-proxy

IP TCP Syn-Proxy

Configure the **ip tcp syn-proxy** command as shown in the following:

1. Configure syn-proxy in the global mode.

```
ServerIron(config)# ip tcp syn-proxy
```

Syntax: ip tcp syn-proxy

2. Enable syn-proxy on each interface handling inbound SYN requests (no change here).

```
ServerIron(config)#interface e 3/1
ServerIron(config-if-3/1)# ip tcp syn-proxy in
```

Usage Guidelines

- The default value for a valid ACK time is 32 seconds and is not user configurable.
- If you enter a value, it is ignored. The command remains in the config file the way you enter it, in case you need to downgrade to the previous release.
- ServerIron may accept the ACK during 33 seconds to 64 seconds due to the syn-proxy algorithm, but it does not accept the ACK after 64 seconds.

- If you enter a value for the **ip tcp syn-proxy <value>** command from the CLI or upgrade from an older release such as 09.4.x to 09.5.2a with the **ip tcp syn-proxy <value>** command in the config file, you receive the following warning message:

```
Warning: The value 10 is being ignored.
        Default ACK validate time of 32 seconds will be used.
        To change the MSL value, issue 'server msl <value>'.
```

Granular Application of Syn-Proxy Feature

This feature applies to ServerIron ADX Syn-Proxy. When this feature is enabled, traffic destined to a virtual server IP is denied if the destination port is not defined under any of the virtual server definitions.

This feature prevents ServerIron ADX from responding with TCP SYN-ACK to TCP SYN for ports not defined under VIP.

Use the following command to validate traffic against a configured virtual port:

```
ServerIron(config)# server syn-cookie-check-vport
```

Syntax: [no] server syn-cookie-check-vport

syn-def

Introduction

Use SYN-def (also known as SYN-Defense) to protect the hosts behind the ServerIron (not the ServerIron itself) by the ServerIron to complete the TCP three-way handshake on behalf of a connecting client. With , the SYNs are forwarded to servers and hosts. There is no SYN-cookie functionality with SYN-def.

show server traffic

Use the **show server traffic** command to display information about the number of times the incomplete connection threshold was reached:

```
ServerIron# show server traffic
Client->Server      =          0  Server->Client      =          0
Drops               =          0  Aged                 =          0
Fw_drops            =          0  Rev_drops            =          0
FIN_or_RST          =          0  old-conn             =          0
Disable_drop        =          0  Exceed_drop          =          0
Stale_drop          =          0  Unsuccessful         =          0
TCP SYN-DEF RST    =          0  Server Resets      =          0
Out of Memory       =          0  Out of Memory        =          0
```

The last line contains information relevant to the incomplete connection threshold. The TCP SYN-DEF RST field displays the number of times the incomplete connection threshold was reached. The Server Resets field displays the number of times the ServerIron sent a TCP RESET packet to the destination real server.

syn-def-dont-send-ack

The SYN-def feature allows the ServerIron to complete the TCP three-way handshake on behalf of a connecting client. When a connecting client sends a TCP SYN to a server, the ServerIron forwards the SYN to the real server, then forwards the SYN ACK from the server to the client. Next, the ServerIron sends an ACK to the real server,

completing the three-way handshake on behalf of the connecting client. This action allows the real server to move the connection from its pending connection queue to its established (and much larger) connection queue.

Starting with Release , use the **server syn-def-dont-send-ack** command to prevent the ServerIron from sending the ACK to the real server to complete the three-way handshake.

EXAMPLE:

```
ServerIron ADX(config)#server syn-def-dont-send-ack
```

show server debug

Use the **show server debug** command to display information about the configuration, as shown in the following example:

```
SLB-chassis1/1#show server debug
```

```
Generic Deug Info
BP Distribution      =          Enabled      =          No
No of BPs           =          3      No of Partner BPs   =          0
Partner Chassis MAC = 0000.0000.0000
Partner BP1 MAC     = 0000.0000.0000      Partner BP2 MAC     = 0000.0000.0000
Partner BP3 MAC     = 0000.0000.0000      Partner BP4 MAC     = 0000.0000.0000
Partner BP5 MAC     = 0000.0000.0000      Partner BP6 MAC     = 0000.0000.0000
```

```
Server Load Balancing Debug Info
Total Get           =          3      Total Free           =          0
Get Fails           =          0      Get Buffer failure   =          0
Forward Sp          =          0      Reverse Sp          =          0
Bad creates         =          0      TCP Resets          =          0
Fw resets           =          0      Rev Resets          =          0
Double Free         =          0      Error               =          0
Free inv Sess Idx   =          0      Free list Idx inv   =          0
Cache-Reassigns     =          0      Trans-Denied        =          0
Multi Path Fwd Use  =          0      Multi Path Rev Use  =          0
Bad non-owner       =          0      Select Fwall        =          0
FTP-trans-error     =          0      Cache track-error   =          0
Fw tcp inside move  =          0      Fw udp inside move  =          0
Fw SYNC delayed    =          0      ownership contention =          0
FW stale to conns   =          0      FW stale to delq con =          0
FW stale from conns =          0      FW stale from delq c =          0
FW stale from nuke c =          0      Sac frwds           =          0

Unxpectd udata     =          0      Unxpectd udata(def) =          0
Client->Server      =          0      Server->Client       =          0
Drops              =          0      Aged                =          0
Fw_drops           =          0      Rev_drops           =          0
FIN_or_RST         =          0      old-conn            =          0
Disable_drop       =          0      Exceed_drop         =          0
Stale_drop         =          0      Unsuccessful        =          0
SYN def/proxy RST  =          0      Server Resets       =          0
Out of Memory      =          0      Out of Memory       =          0
last conn rate     =          0      max conn rate       =          0
last TCP attack rate =          0      max TCP attack rate =          0
fast vport found   =          0      fast vport n found  =          0
Fwd to non-static FI =          0      Dup stale SYN       =          0

TCP forward FIN     =          0      TCP reverse FIN     =          0
Fast path FWD FIN   =          0      Fast path REV FIN   =          0
Fast path SLB SYN   =          0      Dup SYN after FIN   =          0
Duplicate SYN       =          0      Duplicate sessions   =          0
```

No Response to Non-Syn First Packet of a TCP flow

ServerIron can remain passive for non-SYN packet in the beginning of the flow. The default behavior is to send TCP RESET to client when a non-SYN packet is received in the beginning.

By default, when ServerIron ADX receives TCP packet that is destined to VIP and there is no session match then it sends TCP reset to the sender. However, if one desires to remain passive then the above feature can be enabled.

To not send the reset packet, use the following command:

```
ServerIron(config)# server reset-on-syn-only
```

To remove the configuration, use the following command:

```
ServerIron(config)# no server reset-on-syn-only
```

Syntax: [no] server reset-on-syn-only

Prioritizing Management Traffic

ServerIron ADX software allows the system to prioritize traffic destined to the management IP address in order to facilitate uninterrupted access to the ServerIron switch even under heavy load conditions. This feature allows you to prioritize management traffic based on the following:

1. Client IP address/subnet
2. Protocol (TCP/UDP/IP) and
3. TCP or UDP port number

With this feature turned on, the specified traffic is directly forwarded to the Management Module in hardware. In the following example, traffic from the source subnet 1.1.1.1 and destined to management IP 10.45.16.104 for TCP port 22 (SSH) is prioritized:

```
ServerIron(config)# server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0
10.45.16.104 6 22
```

Syntax: server prioritize-mgmt-traffic <source ip> <mask> <destination ip> [<protocol>] [<port>]

The <source ip> variable specifies the Source IP address.

The <mask> variable specifies the Mask for the source IP address.

The <destination ip> variable specifies the Destination management IP address. The destination IP address must already be configured on the ServerIron ADX. If the IP address is not configured, the command is rejected.

The <protocol> variable specifies any protocol.

The <port> variable specifies a TCP or UDP port.

It is also possible to prioritize management traffic from any source ip as shown in the example below:

```
ServerIron(config)# server prioritize-mgmt-traffic any 10.45.16.104 6 22
```

Syntax: [no] server prioritize-mgmt-traffic any <destination ip> [<protocol>] [<port>]

NOTE: The prioritizing management traffic feature should not be enabled for a ServerIron ADX router VE address if this interface is used for source-NAT as that would break the SLB traffic flow.

See the following examples:

Prioritization of TCP port 80 traffic to management IP 200.1.1.1

```
ServerIron# server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0 200.1.1.1 6 80
```

Prioritization of TCP port 80 traffic to management IP 200.1.1.1 from any source IP address

```
ServerIron# server prioritize-mgmt-traffic any 200.1.1.1 6 80
```

Prioritization of UDP port 2222 traffic to management IP 200.1.1.1

```
ServerIron# server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0 200.1.1.1 17 2222
```

Prioritization of IP protocol 89 (OSPF) traffic to management IP 200.1.1.1

```
ServerIron# server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0 200.1.1.1 89
```

Protection Against Attack in Hardware

ServerIron ADX allows for protection against attack in hardware without impacting MP or BP CPU utilization. Configure the **server the drop-all-mgmt-access** command to drop all traffic destined to a specified management IP address.

The following command drops all traffic destined to the management IP address 10.45.16.104:

```
ServerIron(config)# server drop-all-mgmt-access 10.45.16.104
```

Syntax: [no] server drop-all-mgmt-access <destination ip>

NOTE: For a router, the destination IP address is the physical or ve interface IP address. For a switch, the destination IP address is the management IP address.

The server drop-all-mgmt-access feature when used in combination with the server prioritize-mgmt-traffic feature allows you to prioritize valid traffic while blocking unwanted traffic destined to the management IP address.

For example, with the following configuration, only ssh, telnet and http traffic destined to management IP address 10.45.16.104 will be prioritized and all other traffic destined to 10.45.16.104 will be dropped.

```
ServerIron(config)#server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0
10.45.16.104 6 22
ServerIron(config)#server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0
10.45.16.104 6 23
ServerIron(config)#server prioritize-mgmt-traffic 1.1.1.1 255.255.255.0
10.45.16.104 6 80
ServerIron(config)#server drop-all-mgmt-access 10.45.16.104
```

Peak BP Utilization with TRAP

Show CPU-Utilization Command Enhancement

The **show cpu-utilization** command displays CPU utilization peaks since the system boot or the last reset of counters (using the **clear cpu utilization** command).

The command, **clear cpu-utilization**, on both the MP and the BP is used to reset the counter.

BP Utilization Threshold

The **bp-utilization-threshold** command allows you to specify a threshold for BP CPU utilization. Define this command under the global configuration mode.

When the threshold is exceeded, the event is logged and a trap is sent. The log and trap are rate-limited to one per two minutes.

The command takes a percentage string as parameter. For example:

```
ServerIron(config)# bp-utilization-threshold 80.5%
```

Syntax: bp-utilization-threshold <percentage>

MP Utilization Threshold

The **mp-utilization-threshold** command specifies a threshold for BP CPU utilization. Define this command under the global configuration mode.

When the threshold is exceeded, the event is logged and a trap is sent. The log and trap are rate-limited to one every two minutes.

The command takes a percentage string as parameter. For example:

```
ServerIron(config)# mp-utilization-threshold 80.5%
```

Syntax: mp-utilization-threshold <percentage>

Transaction Rate Limit (TRL)

Transaction Rate Limit, allows the ServerIron ADX to monitor and limit traffic from any one IP address.

Understanding Transaction Rate Limit

Transaction Rate Limit counts the number of transactions received from any one IP address. If the transaction count exceeds a specified threshold value, traffic from that IP address is held and not processed for a specified number of minutes.

Transaction rate limit provides the flexibility to specify different configurations for different clients, based on the client IP address/prefix.

Transaction rate limit provides the following benefits:

- Ability to apply a default transaction rate limit value to all clients, while maintaining an exception list.
- Ability to apply a different transaction rate limit rate per client IP or prefix.
- Ability to exclude specific IP addresses or prefixes from transaction rate limit and maintain an exclude list.
- Ability to apply transaction rate limit to traffic coming to a specific VIP only.
- Ability to operate on a per VIP basis, whereby a different rate limit can be applied to traffic coming to a different VIP.

Configuring Transaction Rate Limit

To enable transaction rate limit, you must configure parameters for each client address/prefix and apply the transaction rate limit configuration to a specific VIP.

Prerequisites

Before you can configure transaction rate limit, you must configure a virtual server. The following example shows how to configure a virtual server:

```
ServerIron> enable
ServerIron# config t
ServerIron(config)# server virtual-name-or-ip bwVIP 1.1.1.33
```

Syntax: [no] server virtual-name-or-ip <vip-name-or-address> <ip address>

Configure Transaction Rate Limit Rule Set

The transaction rate limit parameters are grouped into a set and each set is associated with a name. To create a set of transaction rate limit rules, follow these steps:

1. Enable privileged EXEC mode.

```
ServerIron> enable
```

2. Enter global configuration mode.

```
ServerIron# configure terminal
```

3. Configure name of a transaction rate limit rule set and enter client transaction rate limit configuration mode.

```
ServerIron(config)#client-trans-rate-limit tcp TRL1
```

Syntax: [no] client-trans-rate-limit tcp | udp | icmp <name>

4. Specify the trl keyword for client subnet and set connection rate.

```
ServerIron(config-client-trl)# trl 100.1.1.0 255.255.255.0 monitor-interval 3
conn-rate 10 hold-down-time 1
```

Syntax: [no] trl <client-IP> <client-mask> monitor-interval <mon-value> conn-rate <con-value> hold-down-time <hold-down-value>

Configure Transaction Rate Limit to Exclude a Client

You can configure a client address/prefix to be excluded from transaction rate limiting within a transaction rate limit configuration group.

To exclude a client from transaction rate limit, follow these steps:

1. Enable privileged EXEC mode.

```
ServerIron> enable
```

2. Enter global configuration mode.

```
ServerIron# configure terminal
```

3. Specify the name of the transaction rate limit rule set and enter client transaction rate limit configuration mode.

```
ServerIron(config)# client-trans-rate-limit tcp TRL1
```

Syntax: [no] client-trans-rate-limit tcp | udp | icmp <name>

4. Specify the **trl** parameter for the client subnet and the **exclude** keyword.

```
ServerIron(config-client-trl)# trl 100.1.1.0 255.255.255.0 exclude
```

Syntax: [no] trl <client-ip> <client-mask> exclude

Configure a Transaction Rate Limit Default

You can specify a default transaction rate limit configuration for all other clients that are not explicitly configured. To create a transaction rate limit default for a group, follow these steps:

1. Enable privileged EXEC mode.

```
ServerIron> enable
```

2. Enter global configuration mode.

```
ServerIron# configure terminal
```

3. Specify name of transaction rate limit rule set and enter client transaction rate limit configuration mode.

```
ServerIron(config)# client-trans-rate-limit tcp TRL1
```

Syntax: [no] client-trans-rate-limit tcp | udp | icmp <name>

4. Specify the default trl parameter for this group.

```
ServerIron(config-client-trl)# trl default monitor-interval 3 conn-rate 10 hold-down-time 1
```

Syntax: [no] trl default monitor-interval <mon-value> conn-rate <con-value> hold-down-time <hold-down-value>

Configure Transaction Rate Limit for Pass Through Traffic

You can configure transaction rate limit for traffic that is not going to a virtual server. You can configure only one group for pass through traffic.

To create a transaction rate limit group for pass through traffic, follow these steps:

1. Enable privileged EXEC mode.

```
ServerIron> enable
```

2. Enter global configuration mode.

```
ServerIron# configure terminal
```

- Specify name of BW rule set and enter client bandwidth configuration mode.

```
ServerIron(config)# client-trans-rate-limit tcp default
```

Syntax: [no] client-trans-rate-limit tcp | udp | icmp default

- Specify the trl parameter for the client subnet and set a connection rate.

```
ServerIron(config-client-trl)#trl 100.1.1.0 255.255.255.0 monitor-interval 3
conn-rate 10 hold-down-time 1
```

Syntax: [no] trl <client-IP> <client-mask> monitor-interval <mon-value> conn-rate <con-value> hold-down-time <hold-down-value>

Configure TCP/UDP Port Number for an Interface

To apply transaction rate limiting to TCP traffic coming into port 80 on interface 1/1, follow these steps:

- Enable privileged EXEC mode.

```
ServerIron> enable
```

- Enter global configuration mode.

```
ServerIron# configure terminal
```

- Specify the interface and enter interface-configuration mode.

```
ServerIron(config)# interface ethernet 1/1
```

- Specify the **tcp** traffic type and transaction rate on port 80 of interface 1/1.

```
ServerIron(config-if-1/1)# ip tcp trans-rate 80
```

Syntax: ip tcp | udp trans-rate <ports>

Syntax: ip icmp trans-rate

NOTE: The <ports> parameter specifies one or more TCP or UDP ports to monitor. You can monitor up to four ports.

Apply Transaction Rate Limit to a VIP

After configuring transaction rate limit, you must bind transaction rate limit to a VIP. To enable transaction rate limit, follow these steps:

- Enable privileged EXEC mode.

```
ServerIron> enable
```

- Enter global configuration mode.

```
ServerIron# configure terminal
```

- Specify **server virtual-name-or-ip** command and VIP name to enter virtual server configuration mode.

```
ServerIron(config)# server virtual-name-or-ip bwVIP
```

Syntax: [no] server virtual-name-or-ip <name-or-address>

- Specify the BW parameter and BW rule set.

```
ServerIron(config-vs-bwVIP)# client-trans-rate-limit trl
```

Syntax: [no] client-trans-rate-limit <name>

Download Transaction Rate Limit Configuration from a TFTP Server. (Optional)

When a Transaction Rate Limit configuration becomes very large, you can download the configuration from a TFTP server.

The following example shows how to download a Transaction Rate Limit configuration from a TFTP server:

```
ServerIron(config)# server trl tftp 100.1.1.1 test.trl 2
```

Syntax: server trl tftp <ip-address> <trl_config_file_name> <retry_count>

Specify the following values:

<ip_address> —IP address of the TFTP server.

<trl_config_file_name> —File name of Transaction Rate Limit configuration.

<retry_count> —Retry number for the download.

Verify that the Transaction Rate Limit configuration file is in the following format:

```
client-trans-rate-limit tcp trl101
trl 10.2.24.0/24 monitor-interval 50 conn-rate 100 hold-down-time 60
trl 10.2.24.10/32 exclude
```

NOTE: This is the same format as the **show running-configuration** command generates:

Transaction Rate Limit Command Reference

This section describes the syntax, semantics, and usage for each transaction rate limit command. This section contains the following sections:

- client-trans-rate-limit
- trl

client-trans-rate-limit

Use the **client-trans-rate-limit** command in the global configuration mode to configure a transaction rate limit rule name and traffic type.

Syntax: client-trans-rate-limit {icmp <name> | default} | {tcp <name> | default} | {udp <name> | default}

icmp - Specifies ICMP transaction rate limit for client subnet.

tcp - Specifies TCP transaction rate limit for client subnet.

udp - Specifies UDP transaction rate limit for client subnet.

<name> - Specifies the name for this configuration.

default - Specifies default.

trl

Use the **trl** command in the global configuration client-trl mode to configure transaction rate limit rules.

Syntax: trl {default | <client-IP> <client-mask> {exclude | monitor-interval <monitor-value> conn-rate <connection-value> hold-down-time <hold-down-value>}}

default - Specifies default transaction rate limit parameter.

<client-IP> - Specifies client subnet.

<client-mask> - Specifies client mask.

exclude - Specifies to exclude the prefix from transaction rate limit.

monitor-interval - Specifies time interval for monitoring in 100ms.

<monitor-value> - Specifies value of time interval for monitoring.

conn-rate - Specifies connection rate.

<connection-value> - Specifies value of connection rate for client.

hold-down-time - Specifies time for holding down source.

<hold-down-value> - Specifies hold down time in minutes.

Command Modes

Global configuration mode.

global trl

If TRL per client subnet is not needed, Global TRL can be used to create a configuration to apply to all the incoming traffic.

Use **ip [tcp | udp | icmp] trans-rate** to enable TRL on the ServerIron for TCP, UDP, or ICMP traffic. If any more than a specified number packets per second come from the same IP address over a specified interval, then all traffic from that IP address is held down for a specified number of minutes.

Syntax: [no] ip [tcp | udp | icmp] trans-rate monitor-interval <interval> conn-rate <rate> hold-down-time <minutes>

monitor-interval <interval>—Amount of time used to measure incoming traffic. This parameter is specified in increments of 100ms. For example, to measure traffic over a 1 second interval, you would specify 10 for this.

conn-rate <rate>—Threshold for the number of connections per second from any one IP address. Traffic exceeding this rate over the specified interval is subject to hold down.

hold-down-time <minutes>—Number of minutes that traffic from an IP address that has sent packets at rate higher than the configured threshold is to be held down.

EXAMPLE:

```
ServerIron(config)# ip tcp trans-rate monitor-interval 600 conn-rate 100 hold-down-time 5
```

This command configures the ServerIron to monitor incoming TCP traffic. If more than 100 TCP connections per second arrive from the same IP address over a 6-second interval (600 X 100ms), then all TCP traffic from that IP address is held down for 5 minutes.

To apply TRL to TCP traffic coming into port 80 on interface 1/1.

```
ServerIron(config)# interface ethernet 1/1
ServerIron(config-if-1/1)# ip tcp trans-rate 80
```

where <ports> sets one or more TCP or UDP ports to monitor. With TRL, the ServerIron can monitor up to 4 specific ports. The ServerIron can also monitor traffic to all the ports by configuring the default port.

TRL Plus Security ACL-ID

Even though TRL is applied to an interface and effects all traffic received on this interface, with the **security acl-id** <acl-num> command TRL can be applied only to specific traffic coming in on that interface. See "security acl-id" on page 2-24.

Transaction Rate Limit Hold-down Value

if you configure "hold down 0," the incoming request is not held down. Instead it generates a log.

Displaying IP Address with Held Down Traffic

To display a list of IP addresses whose traffic has been held down, enter commands such as the following:

```
ServerIron# rconsole 2 1
ServerIron2/1 #show security holddown

source          destination    vers attempt start      last      HD time
192.168.2.30    Any tcp        ???? 0    000ab6ae 00000000  Y  9
192.168.2.40    Any tcp        ???? 0    000ab6ea 00000000  Y  9
```

Syntax: rconsole <slotnum> <cpunum>

Syntax: show security holddown

The following table lists the output from the **show security holddown** command:

Table 2.1: Output from the show security holddown command

Field	Description
source	Source IP address that is currently being held down
destination	TCP, UDP, or ICMP depending on the type of traffic sent by the client.
vers	Used by Brocade Technical Support.
attempt	Number of connection attempts made by the client during the current monitoring interval.
start	Time stamp representing the start of the monitoring interval.
last	Time stamp representing the last time the ServerIron received a connection request from the client.
HD	Whether the IP address is currently being held down. Y indicates that the address is being held down. N indicates that it isn't.
time	Time remaining for this IP address to be held down, if the HD field contains Y.

Refusing New Connections from a Specified IP Address

Use the **security hold-source-ip** command to refuse new connections from a specified IP address for a specified amount of time. This feature applies to all TCP, UDP, and ICMP traffic originating from the specified IP address.

Syntax: [no] security hold-source-ip <ip-address> <minutes>

EXAMPLE:

To configure the ServerIron to refuse connections from 192.168.9.210 for 20 minutes, enter:

```
ServerIron(config)# security hold-source-ip 192.168.9.210 20
```

To display the IP addresses from which connections are currently being refused:

```
ServerIron# rconsole 2 1
ServerIron2/1 # show security holddown

source          destination    vers attempt start      last      HD time
192.168.2.30    Any tcp       ???? 0      000ab6ae 00000000  Y  9
192.168.2.40    Any tcp       ???? 0      000ab6ea 00000000  Y  9
```

The IP addresses for which connections are being refused are displayed in the source column.

HTTP TRL

This section describes how to use the HTTP Transaction Rate Limiting (TRL) feature with ServerIron devices.

Overview of HTTP TRL

HTTP TRL provides HTTP transaction rate limiting for SSL and HTTP traffic, based on a customer ID. Existing ServerIron TRL features, which are based on source IP addresses, are inadequate in environments where a client

is identified by an application user ID. HTTP TRL allows you to prevent per-client over subscription by allowing you to configure features, such as transaction and connection rate limiting, based on customer IDs.

With HTTP TRL, the rate limit configuration for each customer is grouped into a set. Each of these groups can be applied to multiple VIPs. A counter is maintained on per-VIP basis. When a client request is received, the client customer ID is extracted and decoded. A table lookup is performed on the customer ID and, if the client is subjected to a rate limit, a session lookup is done to locate the current connection information.

For each BP, the current counter is checked against the configuration. If the limit is exceeded, the configured action occurs.

HTTP TRL Features

Before you configure HTTP TRL, you should be aware of the following benefits and restrictions for this feature.

- The customer ID is contained within the HTTP header, is alphanumeric, and can be up to 101 characters in length.
- Maximum customer ID entries is 35K.
- Customer ID entries can be manually configured or have dynamic upload support.
- All customer connections are supported on a single VIP with support for up to 10K connections.
- Customer report response times can run up to 120 seconds before they timeout at the gateway tier.
- Rate-limiting functionality must support rate over time and total connections, based on customer ID.
- Max-conn currently works only for HTTP1.0.
- This feature supports http redirect, or drop client response actions once rate-limit has been exceeded.
- This feature provides event and threshold alert monitoring and notification, based on specific customer connection SLAs.

Configuring HTTP TRL

This section describes how to configure the HTTP TRL feature.

Configuring HTTP TRL Client

Use the following procedures to configure the HTTP TRL client rate limit and the client maximum connection.

Configuring HTTP TRL Client Rate Limit

To configure the HTTP TRL client rate limit, follow these steps:

1. Define an HTTP TRL policy.

```
ServerIron(config)# http-trl-policy p1
```

Syntax: [no] http-trl-policy <policy-name>

2. Configure an HTTP TRL client rate limit.

```
ServerIron(config-http-trl-p1)# client-name c1 monitor-interval 1 10 20 0
```

Syntax: [no] client-name <client-name> monitor-interval <interval-value> <warning-rate> <shutdown-rate> <holddown-interval>

For more detailed command information, see “client-name <client-name> monitor-interval” on page 2-21.

3. Configure the action to take if a client exceeds the configured rate limit (optional).

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action reset
```

Syntax: [no] client-name <client-name> exceed-action reset

Configuring HTTP TRL Client Maximum Connection

To configure HTTP TRL client maximum connection, follow these steps:

1. Define an HTTP TRL policy.

```
ServerIron(config)# http-trl-policy p1
```

Syntax: [no] http-trl-policy <policy-name>

2. Configure an HTTP TRL client maximum connection.

```
ServerIron(config-http-trl-p1)# client-name c1 max-conn 10
```

Syntax: [no] client-name <client-name> max-conn <max-conn-value>

<max-conn-value>—specifies maximum number of connection client can setup.

3. Configure the action to take if a client exceeds the configured maximum connections (optional).

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action reset
```

Syntax: [no] client-name <client-name> exceed-action reset

Configuring HTTP TRL Defaults

Use the following procedures to configure the HTTP TRL default rate limit and the default maximum connection.

Configuring HTTP TRL Default Rate Limit

To configure HTTP TRL default rate limit, follow these steps:

1. Define an HTTP TRL policy.

```
ServerIron(config)# http-trl-policy p1
```

Syntax: [no] http-trl-policy <policy-name>

2. Configure an HTTP TRL default rate limit.

```
ServerIron(config-http-trl-p1)# default monitor-interval 1 10 20 0
```

Syntax: [no] default monitor-interval <interval-value> <warning-rate> <shutdown-rate> <holddown-interval>

3. Configure the action to take if a client exceeds the configured rate limit (optional).

```
ServerIron(config-http-trl-p1)# default exceed-action reset
```

Syntax: [no] default exceed-action reset

Configuring HTTP TRL Default Maximum Connection

To configure HTTP TRL default maximum connection, follow these steps:

1. Define an HTTP TRL policy.

```
ServerIron(config)# http-trl-policy p1
```

Syntax: [no] http-trl-policy <policy-name>

2. Configure an HTTP TRL default maximum connection.

```
ServerIron(config-http-trl-p1)# default max-conn 10
```

Syntax: [no] default max-conn <max-conn-value>

3. Configure the action to take if a client exceeds the configured maximum connection (optional).

```
ServerIron(config-http-trl-p1)# default exceed-action reset
```

Syntax: [no] default exceed-action reset

Sample HTTP TRL Configuration

This section describes how to configure a sample HTTP TRL configuration. This scenario describes all the required steps for configuring HTTP TRL, with notes the optional steps. This configuration consists of four parts:

- Creating an HTTP TRL policy with a client rate limit
- Configuring Layer 4 server load balancing
- Creating a CSW rule and policy with HTTP TRL
- Enabling Layer 7 server load balancing

Create an HTTP TRL Policy with Client Rate Limit

To configure a HTTP TRL policy with client rate limit, follow these steps:

1. Define an HTTP TRL policy.

```
ServerIron(config)# http-trl-policy p1
```

Syntax: [no] http-trl-policy <policy-name>

2. Configure an HTTP TRL client rate limit.

```
ServerIron(config-http-trl-p1)# client-name c1 monitor-interval 1 10 20 0
```

Syntax: [no] client-name <client-name> monitor-interval <interval-value> <warning-rate> <shutdown-rate> <holddown-interval>

3. Configure the action to take if a client exceeds the configured rate limit (optional).

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action reset
```

Syntax: [no] client-name <client-name> exceed-action reset

Configure Layer 4 SLB

To configure Layer 4 SLB, follow these steps:

1. Define a real server (1) with an IP address.

```
ServerIron(config)# server real web1 1.1.1.1
```

Syntax: server real <real-server> <ip-address>

2. Define a real HTTP port on the real server.

```
ServerIron(config-rs-web1)# port http
```

Syntax: port http

3. Define a real server (2) with an IP address.

```
ServerIron(config-rs-web1)# server real web2 1.1.1.2
```

Syntax: server real <vip-name> <ip-address>

4. Define a real HTTP port on the real server and exit.

```
ServerIron(config-rs-web2)# port http
```

Syntax: port http

```
ServerIron(config-rs-web2)# exit
```

Syntax: exit

5. Define a virtual server with an IP address.

```
ServerIron(config)# server virtual-name-or-ip csw-vip 1.1.1.100
```

Syntax: server virtual-name-or-ip <vip-name-or-ip-address> <ip-address>

6. Define a virtual HTTP port on the virtual server.

```
ServerIron(config-vs-csw-vip)#port http
```

Syntax: port http

7. Bind HTTP ports on real servers web1 and web2 to the virtual port HTTP.

```
ServerIron(config-vs-csw-vip)# bind http web1 http web2 http
```

Syntax: bind http <real-server> http <vip-name>

Create a CSW Rule and Policy with HTTP TRL

1. Define a CSW rule to match a pattern in the HTTP header that contains the client name.

```
ServerIron(config)# csw-rule rule1 header Authorization pattern Basic
```

Syntax: csw-rule <rule-name> header <Authentication> pattern <Basic>

2. Define a CSW policy.

```
ServerIron(config)# csw-policy policy1
```

Syntax: csw-policy <policy-name>

3. Specify an action to apply HTTP TRL policy when the CSW rule is matched.

```
ServerIron(config-csw-policy1)# match rule1 http-trl p1
```

Syntax: match <rule-name> http-trl <http-trl-policy-name>

Enable Layer 7 SLB

To configure Layer 7 SLB, follow these steps:

1. Bind the policy to a virtual HTTP port on the virtual server.

```
ServerIron(config-vs-csw-vip)# port http csw-policy policy1
```

Syntax: port http csw-policy <policy-name>

2. Enable CSW on the virtual port.

```
ServerIron(config-vs-csw-vip)# port http csw
```

Syntax: port http csw

Displaying HTTP TRL

This section describes how to display HTTP TRL information.

Display All HTTP TRL Policies

To show all running configurations for HTTP TRL policies, use the following command:

```
ServerIron# show run http-trl-policy all
```

Syntax: show run http-trl-policy all

EXAMPLE:

```
ServerIron# show run http-trl all
!Building configuration...
!Current configuration : 124813 bytes
!
http-trl-policy "my-http-trl-policy-104"
  tftp 50.50.50.105 "http-trl-policy-104.txt"
  client-name "root1" max-conn 1
  client-name "root1" exceed-action reset
  client-name "root10" max-conn 1
  client-name "root10" exceed-action reset
```

```

client-name "root11" max-conn 1
client-name "root11" exceed-action reset
client-name "root12" max-conn 1
client-name "root12" exceed-action reset
client-name "root13" max-conn 1
client-name "root13" exceed-action reset
client-name "root14" max-conn 1
client-name "root14" exceed-action reset
client-name "root15" max-conn 1
client-name "root15" exceed-action reset
client-name "root16" max-conn 1
client-name "root16" exceed-action reset
client-name "root17" max-conn 1
client-name "root17" exceed-action reset...

```

Display HTTP TRL Policy From Index

To show a running configuration for an HTTP TRL policy starting from an index, enter the following command:

```
ServerIron# show run http-trl-policy my-http-trl-policy-104 2
```

Syntax: show run http-trl-policy <policy-name> <index>

EXAMPLE:

```

ServerIron# show run http-trl my-http-trl-policy-104 2
!Building configuration...
!Current configuration : 4261 bytes
client-name "root11" max-conn 1
client-name "root11" exceed-action reset
client-name "root12" max-conn 1
client-name "root12" exceed-action reset
client-name "root13" max-conn 1
client-name "root13" exceed-action reset
client-name "root14" max-conn 1
client-name "root14" exceed-action reset
client-name "root15" max-conn 1
client-name "root15" exceed-action reset
client-name "root16" max-conn 1
client-name "root16" exceed-action reset
client-name "root17" max-conn 1
client-name "root17" exceed-action reset
client-name "root18" max-conn 1
client-name "root18" exceed-action reset
client-name "root19" max-conn 1
client-name "root19" exceed-action reset
client-name "root2" max-conn 1
client-name "root2" exceed-action reset
client-name "root20" max-conn 1...

```

Display HTTP TRL Policy Client

To show a running configuration for an HTTP TRL policy client, enter the following command:

```
ServerIron# show run http-trl-policy my-http-trl-policy-104 root1
```

Syntax: show run http-trl-policy <policy-name> <client-name>

EXAMPLE:

```

ServerIron#show run http-trl my-http-trl-policy-104 root1
!Building configuration...
!Current configuration : 75 bytes

```

```
client-name "root1" max-conn 1
client-name "root1" exceed-action reset
```

Display HTTP TRL Policy Starting from Index

To show a running configuration for an HTTP TRL policy starting from index for a specific number of entries, enter the following command:

```
ServerIron# show run http-trl-policy my-http-trl-policy-104 1 20
```

Syntax: show run http-trl-policy <policy-name> <start-index> <number-of-entries>

EXAMPLE:

```
ServerIron# show run http-trl my-http-trl-policy-104 1 20
!Building configuration...
!Current configuration : 1500 bytes
client-name "root10" max-conn 1
client-name "root10" exceed-action reset
client-name "root11" max-conn 1
client-name "root11" exceed-action reset
client-name "root12" max-conn 1
client-name "root12" exceed-action reset
client-name "root13" max-conn 1
client-name "root13" exceed-action reset
client-name "root14" max-conn 1
client-name "root14" exceed-action reset
client-name "root15" max-conn 1
client-name "root15" exceed-action reset
client-name "root16" max-conn 1
client-name "root16" exceed-action reset
client-name "root17" max-conn 1
client-name "root17" exceed-action reset
client-name "root18" max-conn 1
client-name "root18" exceed-action reset
client-name "root19" max-conn 1
client-name "root19" exceed-action reset
client-name "root2" max-conn 1...
```

Display HTTP TRL Policy Matching a Regular Expression

To show a running configuration for an HTTP TRL policy matching a specific regular expression (regex), enter the following command:

NOTE: The syntax for regex is the same as for piping.

```
ServerIron# show run http-trl-policy my-http-trl-policy-109 regex ot1
```

Syntax: show run http-trl-policy <policy-name> regex <regular expression>

EXAMPLE:

```
ServerIron#show run http-trl my-http-trl-policy-104 regex ot1
!Building configuration...
!Current configuration : 825 bytes
client-name "root1" max-conn 1
client-name "root1" exceed-action reset
client-name "root10" max-conn 1
client-name "root10" exceed-action reset
client-name "root11" max-conn 1
client-name "root11" exceed-action reset
client-name "root12" max-conn 1
client-name "root12" exceed-action reset
```

```

client-name "root13" max-conn 1
client-name "root13" exceed-action reset
client-name "root14" max-conn 1
client-name "root14" exceed-action reset
client-name "root15" max-conn 1
client-name "root15" exceed-action reset
client-name "root16" max-conn 1
client-name "root16" exceed-action reset
client-name "root17" max-conn 1
client-name "root17" exceed-action reset
client-name "root18" max-conn 1
client-name "root18" exceed-action reset
client-name "root19" max-conn 1...

```

Display HTTP TRL Policy Client Index (MP)

To show an HTTP TRL policy client with a starting and ending index, enter the following command on the MP:

```
ServerIron# show http-trl policy my-http-trl-policy-103 0 10
```

Syntax: show http-trl policy <policy-name> <start entry number> <end entry number>

EXAMPLE:

```

ServerIron# show http-trl policy my-http-trl-policy-103 0 10
Policy Name:          my-http-trl-policy-103
                    configured client count: 1
                    total client count: 1
Client name TDSWS/LoadRunner
                    monitor-interval 1
                    warning rate 10
                    shutdown rate 20
                    holdddown interval 0
                    exceed action: drop
                    dynamic No
                    max-conn track session 0
                    trl track session 0

```

Syntax: show http-trl policy <policy-name> <start entry number> <end entry number>

NOTE: This command entered on the MP only displays configuration information and total entry count for this policy. The same command entered on the BP provides traffic status.

Display HTTP TRL Policy Client Index (BP)

To show HTTP TRL policy client with a starting and ending index, use the following command on the BP:

```
ServerIron# show http-trl policy my-http-trl-policy-103 0 10
```

Syntax: show http-trl policy <policy-name> <start entry number> <end entry number>

EXAMPLE:

```
ServerIron# show http-trl policy my-http-trl-policy-103 0 100
Policy Name:          my-http-trl-policy-103
                    configured client count: 1
                    total client count: 2
Client name V E'Vææ\
                    max-conn 50
                    dynamic Yes
                    max-conn track session 1
                    trl track session 0
                    HTTP_TRL_HIT      3278
                    HTTP_TRL_PASS    1613
                    HTTP_MAX_CONN_F 1665
                    HTTP_TRL_DROP    1665
Client name TDSWS/LoadRunner
                    monitor-interval 1
                    warning rate 10
                    shutdown rate 20
                    holddown interval 0
                    exceed action: drop
                    dynamic No
                    max-conn track session 0
                    trl track session 1
                    HTTP_TRL_HIT      66352
                    HTTP_TRL_PASS    39524
                    HTTP_TRL_FAIL    26828
                    HTTP_TRL_DROP    26828

ServerIron2/1 #
ServerIron2/1 #sh http-trl session 90.90.90.103 80 my-http-trl-policy-103
HTTP-MAX: V E'Vææ\ config 50, current 50
HTTP-TRL: TDSWS/LoadRunner, config 2, attamp 3, hold 0, 1st 3089554, last 3092565
```

Display HTTP TRL Policy for all Client Entries (BP)

To display HTTP TRL policy information for all client entries, enter the following command on the BP:

```
ServerIron2/1# show http-trl resource
```

EXAMPLE:

```
ServerIron2/1#show http-trl resource
Maximum client entry: 35000
Free client entry: 0
Total allocated client entry: 35000
Total freed client entry: 0
Maximum allocated client entry: 35000
Maximum client entry: 35000
Double free client entry: 0
Invalid free client entry: 0
Failed allocate client entry: 0
Double allocated client entry: 0
```

Downloading an HTTP TRL Policy through TFTP

To download an HTTP TRL policy using TFTP, enter a command similar to the following:

```
ServerIron(config-http-trl-p1)# tftp 100.1.1.1 http-trl-config.txt
```

Syntax: tftp <tftp-server-addr> <config-file-name>

NOTE: You can save this command with write memory to automatically initiate a download for this policy after you reload. If you configure more than one policy for TFTP download, and a policy fails the download, the ServerIron does NOT retry, and the subsequent policy does not initiate a download. You must manually issue the command to do a TFTP download.

NOTE: When the total number of HTTP TRL entries exceeds 10k, the **show run time config** command cannot display an http trl-related configuration. You must use a text file to manage it.

NOTE: When any HTTP TRL policy client entry exceeds 1K, the **show run time config** command cannot display a detailed client entry for the HTTP TRL policy.

HTTP TRL Policy Commands

NOTE: You must configure client HTTP TRL before you configure the client exceed-limit

client-name <client-name> monitor-interval

Use the **client-name <client-name> monitor-interval** option in the **http-trl-policy** configuration mode to set client rate limiting parameters.

Syntax: [no] client-name <client-name> monitor-interval <interval-value> <warning-rate> <shutdown-rate> <holddown-interval>

<interval-value>—specifies monitoring window in 100 ms unit.

<warning-rate>—specifies HTTP connection rate (per second) that causes a warning if exceeded.

<shutdown-rate>—specifies HTTP connection rate (per second) that causes a client to hold down.

<holddown-interval>—specifies the length of hold down period, if client exceeds rate limit in term of minutes.

NOTE: Value 0 means do not hold down. Hold down holds all traffic.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# client-name c1 monitor-interval 1 10 20 0
```

client-name <client-name> max-conn

Use the **client-name <client-name> max-conn** option in the **http-trl-policy** configuration mode to set client maximum connection parameters.

Syntax: [no] client-name <client-name> max-conn <max-conn-value>

<max-conn-value>—specifies maximum number of connections client can setup.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# client-name c1 max-conn 10
```

NOTE: You must set the client HTTP max-conn configuration before you configure the client exceed-action.

NOTE: Max-conn currently supports only HTTP/1.0.

client-name <client-name> exceed-action

Use the **client-name <client-name> exceed-action** option in the **http-trl-policy** configuration mode to set the action to take if a client exceeds the configured rate limit.

Syntax: [no] client-name <client-name> exceed-action [reset | drop]

[reset | drop]—specifies client request be reset or dropped if exceeds limit.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action [reset]
```

Syntax: [no] client-name <client-name> exceed-action redirect <domain> <url> [port]

<domain> and <url>—specifies client request to be redirected to this new URL, if limit is exceeded.

NOTE: Use an asterisk (*) to keep the same domain or url. This does not apply if the client is using HTTP 1.0.

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action redirect * /new  
exceed.html http
```

NOTE: The same domain is used in the incoming packet.

The optional [port] specifies the new TCP port number for the redirected URL.

```
ServerIron(config-http-trl-p1)# client-name c1 exceed-action redirect www.yahoo.com  
exceed.html http
```

default monitor-interval

Use the **default monitor-interval** option in the **http-trl-policy** configuration mode to set default rate limiting parameters.

Syntax: [no] default monitor-interval <interval-value> <warning-rate> <shutdown-rate> <holddown-interval>

- <interval-value>—specifies monitoring window in 100 ms unit.
- <warning-rate>—specifies HTTP connection rate (per second) that causes a warning if exceeded.
- <shutdown-rate>—specifies HTTP connection rate (per second) that causes a client to hold down.
- <holddown-interval>—specifies the length of hold down period, if client exceeds rate limit in term of minutes.

NOTE: Value 0 means do not hold down. Hold down holds all traffic.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# default monitor-interval 1 10 20 0
```

default max-conn

Use the **default max-conn** option in the **http-trl-policy** configuration mode to set default maximum connection parameters.

Syntax: [no] default max-conn <max-conn-value>

<max-conn-value>—specifies maximum number of connections client can setup.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# default max-conn 10
```

NOTE: Max-conn currently supports only HTTP/1.0.

default exceed-action

Use the **default exceed-action** option in the **http-trl-policy** configuration mode to set the action to take if a default exceeds the configured rate limit.

Syntax: [no] default exceed-action [reset | drop]

[reset | drop]—specifies default request be reset or dropped if the limit is exceeded.

EXAMPLE:

```
ServerIron(config-http-trl-p1)# default exceed-action [reset | drop]
```

Syntax: [no] default exceed-action redirect <domain> <url> [port]

<domain> and <url>—specifies client request to be redirected to this new URL, if limit is exceeded.

NOTE: Use an asterisk (*) to keep the same domain or url.

```
ServerIron(config-http-trl-p1)# default exceed-action redirect * /new/exceed.html
http
```

NOTE: The same domain is used in the incoming packet.

The optional [port] specifies the new TCP port number for the redirected URL.

```
ServerIron(config-http-trl-p1)# default exceed-action redirect www.yahoo.com /
exceed.html http
```

Logging for DoS Attacks

The following sections describe how to enable logging of DoS attacks.

Configuration Commands

Use the following commands to enable logging of TCP connection rate and attack rate.

Syntax: [no] ip tcp conn-rate <rate> attack-rate <rate>

Syntax: [no] ip tcp conn-rate-change <percentage> attack-rate <percentage>

Syntax: [no] server max-conn-trap <seconds>

Parameters

The **conn-rate** <rate> parameter specifies a threshold for the number of global TCP connections per second that are expected on the ServerIron. A global TCP connection is defined as any packet that requires session processing. For example, 1 SLB, 1 TCS, and 1 SYN-Guard connection would equal 3 global TCP connections, since there are three different connections that require session processing.

NOTE: The ServerIron ADX counts only the new connections that remain in effect at the end of the one second interval. If a connection is opened and terminated within the interval, the ServerIron ADX does not include the connection in the total for the server.

The **attack-rate** <rate> parameter specifies a threshold for the number of TCP SYN attack packets per second that are expected on the ServerIron.

Syslog entries are generated under the following circumstances:

- If the connection rate or attack rate on the ServerIron reaches 80% of the configured threshold.
- If the connection rate or attack rate is still between 80% and 100% of the configured threshold 6 minutes after the last message.
- If the connection rate or attack rate exceeds 100% of the configured threshold.
- If the connection rate or attack rate exceeds 100% of the configured threshold, and has gone up by the configured rate change percentage.
- One minute after the last message indicating that the connection rate or attack rate still exceeds 100% of the configured threshold, and has gone up by the configured rate change percentage.
- Three minutes after the last message, if the connection rate or attack rate is still between 80% and 100% of

the configured threshold, and has gone up by the configured rate change percentage.

The **server max-conn-trap** <seconds> command specifies the number of seconds that elapse between traps, where <seconds> can be from 1 to 300. The default is 50.

EXAMPLE:

```
ServerIron(config)# ip tcp conn-rate 10000 attack-rate 10000
ServerIron(config)# ip tcp conn-rate-change 50 attack-rate 100
ServerIron(config)# server max-conn-trap 30
```

show server conn-rate

Use **show server conn-rate** to display the global TCP connection rate (per second) and TCP SYN attack rate (per second). This command reports global connection rate information for the ServerIron as well as for each real server.

```
ServerIron# show server conn-rate
Avail. Sessions      =      524286  Total Sessions      =      524288
Total C->S Conn      =           0  Total S->C Conn      =           0
Total Reassign       =           0  Unsuccessful Conn    =           0
last conn rate       =           0  max conn rate        =           0
last TCP attack rate =           0  max TCP attack rate  =           0
SYN def RST          =           0  SYN flood            =           0
Server State - 1:enabled, 2:failed, 3:test, 4:suspect, 5:grace_dn, 6:active

Real Server      State  CurrConn  TotConn  LastRate  CurrRate  MaxRate
rs1               3       0         0         0          0         0
```

EXAMPLE:

Maximum Connections

Use **max-conn** to set the number of maximum connections on a global real server level (all ports) or a single port:

```
!
server real rs1 10.10.1.30
  All ports → max-conn 1200
  port http
  One port → port http max-conn 1000
  port http url "HEAD /"
!
```

Access Control Lists

Introduction

To protect the ServerIron against random SYN attacks, Access Control Lists (ACLs) are available to restrict access to the ServerIron. The ServerIron assumes it will not see a SYN-attack from one of the trusted IPs allowed by the ACL.

It is also possible to use ACLs to protect hosts behind the ServerIron against Smurf attacks. For this, you should disallow ICMP broadcasts and UDP broadcasts by listing out all the internal network addresses in an ACL and applying it to the external interface. Though cumbersome, this approach is the only option for the Switch (S) code base. For the Router (R) code base, a future release might automatically block the addresses.

security acl-id

The **security** global command accepts **acl-id** <acl-num> as a parameter.

Syntax: [no] security acl-id <id>

EXAMPLE:

```
ServerIron(config)# security acl-id 4
```

Once **security acl-id** <acl-num> is configured, only packets matching the configured ACL will be subject to the L4 security rules configured on the system. (Specifically, TRL and manual hold down will take effect only for packets matching this configured ACL). If you want specific traffic to bypass the L4 security features, then do not include those IP addresses in the access list.

NOTE: The **security acl-id** takes precedence over all TRL configuration.

permit, deny

A matched **permit** statement is forwarded to the **security acl-id** for further examination (for example, a rogue IP address). A matched **deny** statement is forwarded onto normal L4 SLB processing.

EXAMPLE:

```
!
access-list 1 permit 11.2.2.0 0.0.0.255
access-list 2 deny 10.1.1.0 0.0.0.255
!
```

access-list

ACLs enable you to permit or deny packets based on source and destination IP address, IP protocol information, or TCP or UDP protocol information.

Use the **access-list** command to configure standard or extended ACLs:

Standard ACL—Permits or denies packets based on source IP address. You can configure up to 99 standard ACLs. You can configure up to 1024 individual ACL entries. There is no limit to the number of ACL entries an ACL can contain except for the system-wide limitation of 1024 total ACL entries.

Extended ACL—Permits or denies packets based on the following information:

- IP protocol
- Source IP address or host name
- Destination IP address or host name
- Source TCP or UDP port (if the IP protocol is TCP or UDP)
- Destination TCP or UDP port (if the IP protocol is TCP or UDP)

Standard ACL Syntax:

```
[no] access-list <num> deny | permit <source-ip> | <hostname> <wildcard> [log]
[no] access-list <num> deny | permit <source-ip>/<mask-bits> | <hostname> [log]
[no] access-list <num> deny | permit host <source-ip> | <hostname> [log]
[no] access-list <num> deny | permit any [log]
[no] ip access-group <num> in | out
```

Extended ACL Syntax:

```
[no] access-list <num> deny | permit <ip-protocol> <source-ip> | <hostname> <wildcard> [<operator>]
<source-tcp/udp-port>] <destination-ip> | <hostname> <wildcard>
[<operator> <destination-tcp/udp-port>] [log]

[no] access-list <num> deny | permit host <ip-protocol> any any [log]
[no] ip access-group <num> in | out
```

Parameters:

<num>—The access list number. For standard ACLs, the value can be from 1 – 99. For extended ACLs, the value can be from 100 – 199.

deny | permit—Indicates whether packets that match a policy in the access list are denied (dropped) or permitted (forwarded).

<source-ip>—Specifies the source IP address. Alternatively, you can specify the <hostname>. To specify the host name instead of the IP address, the host name must be configured using the Brocade device's DNS resolver. To configure the DNS resolver name, use the **ip dns server-address** command at the global CONFIG level of the CLI. If you want the policy to match on all source addresses, enter **any**.

<wildcard>—Specifies the mask value to compare against the host address specified by the <source-ip> parameter. The <wildcard> is a four-part value in dotted-decimal notation (IP address format) consisting of ones and zeros. Zeros in the mask mean the packet's source address must match the <source-ip>. Ones mean any value matches. For example, the <source-ip> and <wildcard> values 209.157.22.26 0.0.0.255 mean that all hosts in the Class C sub-net 209.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in CIDR format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "209.157.22.26 0.0.0.255" as "209.157.22.26/24".

NOTE: When you save ACL policies to the startup-config file, the software changes your <source-ip> values if appropriate to contain zeros where the packet value must match. For example, if you specify 209.157.22.26/24 or 209.157.22.26 255.255.255.0, then save the startup-config file, the values appear as 209.157.22.0/24 (if you have enabled display of sub-net lengths) or 209.157.22.0 255.255.255.0 in the startup-config file.

If you enable the software to display IP sub-net masks in CIDR format, the mask is saved in the file in "/<mask-bits>" format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. By default, the ServerIron displays network mask information in class-based notation. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format. If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with sub-net mask in the display produced by the **show access-list** and **show ip access-list** commands.

host <source-ip> | <hostname>—Specifies a host IP address or name. When you use this parameter, you do not need to specify the mask. A mask of all zeros (0.0.0.0) is implied.

<destination-ip> | <hostname> — specifies the destination IP host for the policy. If you want the policy to match on all destination addresses, enter **any**.

any—Configures the policy to match on all host addresses.

log —Configures the device to generate Syslog entries and SNMP traps for packets that are permitted or denied by the access policy.

in | out—Specifies whether the ACL applies to incoming traffic or outgoing traffic on the port to which you apply the ACL.

<ip-protocol>—Indicates the type of IP packet you are filtering. Use the ? command to see a complete list.

<operator>—Specifies a comparison operator for the TCP or UDP port number. This extended ACL parameter applies only when you specify **tcp** or **udp** as the IP protocol. For example, if you are configuring an entry for HTTP, specify **tcp eq http**. You can enter one of the following operators:

- **eq**—Applies to the TCP or UDP port name or number you enter after **eq**.
- **gt**—Applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt**.
- **lt**—Applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt**.
- **neq**—Applies to all TCP or UDP port numbers except the port number or port name you enter after **neq**.
- **range** —Applies to all TCP or UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the range parameter. The range includes the port names or numbers

you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53**. The first port number in the range must be lower than the last number in the range.

- **established** —Applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK or RST bits on (set to "1") in the Control Bits field of the TCP packet header. This policy applies only to established TCP sessions, not to new sessions. See Section 3.1, "Header Format", in RFC 793 for information about this field.

NOTE: This operator applies only to destination TCP ports, not source TCP ports.

<tcp/udp-port>—Specifies the TCP or UDP port number or well-known name. The device recognizes the many well-known names. Use the ? command to see a complete list. For other ports, you must specify the port number.

in | out—Specifies whether the ACL applies to incoming traffic or outgoing traffic on the port to which you apply the ACL.

EXAMPLE:

To configure a standard ACL and apply it to outgoing traffic on port 1 enter:

```
ServerIron(config)# access-list 1 deny host 209.157.22.26 log
ServerIron(config)# access-list 1 deny 209.157.29.12 log
ServerIron(config)# access-list 1 deny host IPhost1 log
ServerIron(config)# access-list 1 permit any
ServerIron(config)# int eth 1
ServerIron(config-if-1)# ip access-group 1 out
ServerIron(config-if-1)# write mem
```

The commands in this example configure an ACL to deny packets from three source IP addresses from being forwarded on port 1. The last ACL entry in this ACL permits all packets that are not explicitly denied by the first three ACL entries.

To configure an extended ACL that blocks all Telnet traffic received on port 1 from IP host 209.157.22.26:

```
ServerIron(config)# access-list 101 deny tcp host 209.157.22.26 any eq telnet log
ServerIron(config)# access-list 101 permit ip any any
ServerIron(config)# int eth 1
ServerIron(config-if-1)# ip access-group 101 in
ServerIron(config)# write mem
```

ip access-list, ip access-group

Use **ip access-list** to create a named IP ACL and **ip access-group** to apply it to an interface.

The commands for configuring named ACL entries are different from the commands for configuring numbered ACL entries. The command to configure a numbered ACL is **access-list**. The command for configuring a named ACL is **ip access-list**. In addition, when you configure a numbered ACL entry, you specify all the command parameters on the same command. When you configure a named ACL, you specify the ACL type (standard or extended) and the ACL number with one command, which places you in the configuration level for that ACL. Once you enter the configuration level for the ACL, the command syntax is the same as the syntax for numbered ACLs.

Syntax: [no] ip access-list extended | standard <string> | <num>

Syntax: [no] ip access-group <string> in | out

Parameters:

The extended and standard parameters indicate the ACL type.

The <string> parameter is the ACL name. You can specify a string of up to 256 alphanumeric characters. You can use blanks in the ACL name if you enclose the name in quotation marks (for example, "ACL for Net1"). The <num> parameter allows you to specify an ACL number if you prefer. If you specify a number, you can specify from 1 – 99 for standard ACLs or 100 – 199 for extended ACLs.

The options at the ACL configuration level and the syntax for the **ip access-group** command are the same for numbered and named ACLs.

EXAMPLE:

```
ServerIron(config)# ip access-list standard Net1
ServerIron(config-std-nacl)# deny host 209.157.22.26 log
ServerIron(config-std-nacl)# deny 209.157.29.12 log
ServerIron(config-std-nacl)# deny host IPhost1 log
ServerIron(config-std-nacl)# permit any
ServerIron(config-std-nacl)# exit
ServerIron(config)# int eth 1/1
ServerIron(config-if-1)# ip access-group Net1 out
```

The commands in this example configure a standard ACL named “Net1”. The entries in this ACL deny packets from three source IP addresses from being forwarded on port 1. Since the implicit action for an ACL is “deny”, the last ACL entry in this ACL permits all packets that are not explicitly denied by the first three ACL entries.

That the command prompt changes after you enter the ACL type and name. The “std” in the command prompt indicates that you are configuring entries for a standard ACL. For an extended ACL, this part of the command prompt is “ext”. The “nacl” indicates that are configuring a named ACL.

EXAMPLE:

To configure a named extended ACL entry:

```
ServerIron(config)# ip access-list extended "block Telnet"
ServerIron(config-ext-nacl)# deny tcp host 209.157.22.26 any eq telnet log
ServerIron(config-ext-nacl)# permit ip any any
ServerIron(config-ext-nacl)# exit
ServerIron(config)# int eth 1
ServerIron(config-if-1)# ip access-group "block Telnet" in
```

EXAMPLE:

To apply an ACL to a subset of ports within a virtual interface:

```
ServerIron(config)# vlan 10 name IP-subnet-vlan
ServerIron(config-vlan-10)# untag ethernet 1/1 to 2/12
ServerIron(config-vlan-10)# router-interface ve 1
ServerIron(config-vlan-10)# exit
ServerIron(config)# access-list 1 deny host 209.157.22.26 log
ServerIron(config)# access-list 1 deny 209.157.29.12 log
ServerIron(config)# access-list 1 deny host IPhost1 log
ServerIron(config)# access-list 1 permit any
ServerIron(config)# interface ve 1
ServerIron(config-vif-1)# ip access-group 1 in eth 1/1 eth 1/3 eth 2/1 to 2/4
```

The commands in the previous example configure port-based VLAN 10, add ports 1/1 – 2/12 to the VLAN, and add virtual routing interface 1 to the VLAN. The commands following the VLAN configuration commands configure ACL 1. Finally, the last two commands apply ACL 1 to a subset of the ports associated with virtual interface 1.

ip strict-acl-mode

Use **[no] ip strict-acl-mode** to enable strict ACL TCP mode. With this mode enabled, the ServerIron ADX compares all TCP packets against the configured ACLs before forwarding them.

By default, when you use ACLs to filter TCP traffic, the device does not compare all TCP packets against the ACLs. Instead, the device compares TCP control packets against the ACLs, but not data packets. Control packets include packet types such as SYN (Synchronization) packets, FIN (Finish) packets, and RST (Reset) packets.

In normal TCP operation, TCP data packets are present only if a TCP control session for the packets also is established. For example, data packets for a session never occur if the TCP SYN for that session is dropped. Therefore, by filtering the control packets, the Brocade device also implicitly filters the data packets associated

with the control packets. This mode of filtering optimizes forwarding performance for TCP traffic by forwarding data packets without examining them. Since the data packets are present in normal TCP traffic only if a corresponding TCP control session is established, comparing the packets for the control session to the ACLs is sufficient for filtering the entire session including the data.

However, it is possible to generate TCP data packets without corresponding control packets, in test or research situations for example. In this case, the default ACL mode does not filter the data packets, since there is no corresponding control session to filter. To filter this type of TCP traffic, use the strict ACL TCP mode. This mode compares all TCP packets to the configured ACLs, regardless of whether the packets are control packets or data packets.

Regardless of whether the strict mode is enabled or disabled, the device always compares TCP control packets against the configured ACLs.

NOTE: If the device's configuration currently has ACLs associated with interfaces, remove the ACLs from the interfaces before changing the ACL mode.

EXAMPLE:

```
ServerIron(config)# ip strict-acl-mode
```

clear statistics dos-attack

Use clear statistics dos-attack to reset counters for ICMP and TCP SYN packet burst thresholds, as displayed by show statistics dos-attack.

EXAMPLE:

```
ServerIron# clear statistics dos-attack
ServerIron# show statistics dos-attack
```

Maximum Concurrent Connection Limit Per Client

This feature restricts each client to a specified number of connections, based on the client's subnet, to prevent any one client from using all available connections.

Limiting the Number of Concurrent Connections Per Client

This feature restricts each client to a specified number of concurrent connections, based on the client's subnet, to prevent any one client from using all available connections.

You associate a configured client subnet with a maximum permissible connection value. The association is stored in the ServerIron by means of a Dynamic Prefix (DP) trie. The key stored in the DP trie is the associated maximum connection value. The choice of the DP trie for storing the client subnet allows to define different prefix lengths and subnet masks for each client subnet. Since the DP trie lookup returns the longest prefix match, it is not required that all configured client subnets should have the same subnet mask.

Configuring the max connection limit per client consists of the following tasks:

- Configure the maximum connections allowed per client address or prefix
- Applying configured number of maximum connections to a specific VIP

Configure the maximum number of connections

1. Begin by creating a policy set or group by entering commands such as the following:

```
ServerIron(config)#client-connection-limit max-conn1
```

Syntax: [no] client-connection-limit <name>

Enter a name for the policy set or group for <name>.

Use the **no** form of the command to delete the policy group.

After creating a name, the CLI changes to the config-client-max-conn level.

2. Next, create the policy for maximum number of connections using one of the following methods:

Create a policy for the maximum number of connections for specific clients

To set a maximum number of connections for a clients in a subnet, enter the a command such as the following:

```
ServerIron(config)# client-connection-limit max-conn1
ServerIron(config-client-max-conn)# max-conn 100.1.1.0 255.255.255.0 10
```

In the example above, clients with IP addresses in the 100.1.1.0 subnet will be allowed only 10 connections.

Syntax: [no] max-conn [<client-ip-address> <client-subnet-mask> <max-connections>

Enter the clients' IP address and subnet mask for <client-ip-address> <client-subnet-mask>

Enter a number from 0 to any value for <max-connections>. There is not default for this parameter.

Specifying a maximum number of connections for clients not specified in a policy

You can specify a default maximum number of connections for all clients that are not specified in any max connection group by entering a command such as the following:

```
ServerIron(config)# client-connection-limit max-conn1
ServerIron(config-client-max-conn)# max-conn default 10
```

In this example, all clients not specified in any max connection group will have a maximum of 10 connections.

Syntax: [no] max-conn [<client-ip-address> <client-subnet-mask> default <max-connections>

Enter a default maximum number of connections for <max-connections>

Excluding clients from maximum connection policy

If you want certain clients to be excluded from any maximum connection policies, enter a command such as the following:

```
ServerIron(config)# client-connection-limit max-conn1
ServerIron(config-client-tr1)# max-conn 100.1.4.0 255.255.255.0 exclude
```

In this example, clients in the 100.1.4.0 subnet will be excluded for any maximum connection rules.

Syntax: [no] max-conn [<client-ip-address> <client-subnet-mask> exclude

Displaying the maximum number of connections for clients that are currently connected

To show the maximum number connection policy for a client that is currently connected, enter command such as the following on the barrel processor (BP) console:

```
ServerIron1# show conn pass1 0
Max Count: 2500 Total Count: 55

IP address Mask config hit denied
0.0.0.0 0.0.0.0 10 0 0
120.20.1.0 255.255.255.192 12 0 0
120.20.1.16 255.255.255.240 15 0 0
120.20.1.21 255.255.255.255 exclude 0 0
120.20.1.23 255.255.255.255 exclude 0 0
120.20.1.24 255.255.255.255 15 20 5
Current connections:
VIP 20.20.1.6: 15
120.20.1.25 255.255.255.255 exclude 0 0
120.20.1.27 255.255.255.255 exclude 20 0
Current connections:
VIP 20.20.1.6: 20
120.20.1.29 255.255.255.255 exclude 0 0
120.20.1.30 255.255.255.255 15 20 5
```

```
Current connections:
VIP 20.20.1.6: 15
120.20.1.33 255.255.255.255 exclude 20 0
ServerIron1#
```

Syntax: show connection-limit <name> <offset>

Enter the name of the max connection policy for <name>.

Enter the starting entry for <offset>

Binding the Policy to a VIP

After creating a maximum connection policy, bind it to a VIP by entering commands such as the following:

```
ServerIron(config)#server virtual-name-or-ip virt-2
ServerIron(config-vs-virt-2)#client-max-conn-limit max-conn1
```

Syntax: [no] client-max-conn-limit <name>

Enter the name of the max connection policy for <name>.

NOTE: When the policy is bound to a VIP, the policy limits the number of connections that a client can have on any real server on the network.

Firewall Load Balancing Enhancements

This section contains the following sections:

- Enabling Firewall Strict Forwarding
- Enabling Firewall VRRPE Priority
- Enabling Track Firewall Group
- Enabling Firewall Session Sync Delay

Enabling Firewall Strict Forwarding

To enable load balancing only when traffic is going to a firewall, use the following command:

```
ServerIron(config)# server fw-strict-fwd
```

Syntax: server fw-strict-fwd

Use the **server fw-strict-fwd** command in the global configuration mode. Without this command, when the ServerIron receives traffic that matches the firewall flow session and the traffic is not received from a firewall, then the ServerIron assumes that it needs to be load balanced to a firewall.

This command checks to ensure that traffic is going to a firewall and only then does the ServerIron load balance it to a firewall.

Enabling Firewall VRRPE Priority

To configure VRRPE state to track the firewall group state, use the following command:

```
ServerIron(config)# server fw-g 2
ServerIron(config-tc-2)#fw-vrrpe-priority
ServerIron(config-tc-2)#
```

Syntax: fw-vrrpe-priority <priority>

Use the **fw-vrrpe-priority** command in the fw-group configuration mode. <priority> is the VRRPE priority associated with current firewall group state. Valid values are 1 to 255.

NOTE: This command can be used with the **track-fw-group** command below to force VRRPE state to track the firewall group state for a specific vrid.

Enabling Track Firewall Group

To enable track-fw-group to track the firewall group state, use the following commands:

```
ServerIron(config)#int ve 1
ServerIron(config-vif-1)# ip vrrp-e vrid 1
ServerIron(config-vif-1-vrid-1)# track-fw-group
```

Syntax: track-fw-group <group-num>

Use the **track-fw-group** command under the VRRPE config level. <group-num> is the firewall group that needs to be tracked for this VRRPE. This command is used along with the fw-vrrpe-priority command to force VRRPE state to track the FW group state. This command works similar to the **track-port** command. When the firewall group state is STANDBY, then the VRRPE current priority is decremented by the fw-vrrpe-priority specified under that firewall group. It is recommended that you track only the firewall group state and no other port, because tracking firewall group state automatically tracks the router ports, firewall paths, and more.

Enabling Firewall Session Sync Delay

To enable server fw-sess-sync-delay, use the following command:

```
ServerIron(config)#server fw-sess-sync-delay 10
```

Syntax: server fw-sess-sync-delay <secs>

Use the **server fw-sess-sync-delay** command added at the global config level. <secs> is the number of seconds to delay the fast session sync after one of the ServerIrons is reloaded in HA FWLB. Valid values range from 1 to 100. This command can be useful in configurations where many real servers or firewalls are configured.

Syn-Cookie Threshold Trap

To configure the syn cookie attack rate threshold, use the following command:

```
ServerIron# server syn-cookie-attack-rate-threshold 10000
```

Syntax: syn-cookie-attack-rate-threshold <threshold>

<threshold> is a decimal number ranging from 1 to 10000000.

If the current syn cookie attack rate is larger than the syn cookie attack rate threshold, the snTrapSynCookieAttackThreshReached trap is generated.

To configure the snTrapSynCookieAttackThreshReached trap interval, use the following command:

```
ServerIron# server max-conn-trap-interval 10
```

Syntax: server max-conn-trap-interval <decimal number in seconds>

<seconds> is a decimal number in seconds. The default value is 60 seconds.

Service Port Attack Protection in Hardware

A ServerIron can be enabled to deny traffic that is destined to VIP address but to a port that is not defined under a VIP. Such traffic can be dropped in hardware without impacting the MP or BP CPUs.

You can enable this feature globally by entering the following command:

```
ServerIron(config)# server vip-protection
```

Syntax: [no] server vip-protection

Once enabled, the VIP protection applies to all existing and new VIP configurations.

If you want to enable the feature on individual VIPs, enter the following command:

```
ServerIron(config)# server virtual-name-or-ip v1  
ServerIron(config-vs-v1)# vip-protection
```

Syntax: [no] vip-protection

VIP protection adds CAM entries for each defined virtual port associated with each VIP. An additional CAM entry is defined for ICMP traffic destined to each VIP. An entry to drop the traffic is also added in the CAM for each VIP, which makes sure that traffic destined to any destination port other than the virtual ports is dropped by hardware.

NOTES:

- VIP protection does not support complex protocols such as FTP, TFTP, MMS, RTSP, SIP, that establish data connections based on the information exchanged on control channel.
- VIP protection cannot be enabled on a VIP that is part of a dynamic NAT address pool.
- VIP protection cannot be used along with features that require binding of virtual default port to real server default port.

