VYATTA, INC. | Vyatta System

Connection Management

REFERENCE GUIDE

Connection Tracking
Flow Accounting

Vyatta
Suite 200
1301 Shoreway Road
Belmont, CA 94002
vyatta.com
650 413 7200
1 888 VYATTA 1 (US and Canada)

COPYRIGHT

Copyright © 2005–2012 Vyatta, Inc. All rights reserved.

Vyatta reserves the right to make changes to software, hardware, and documentation without notice. For the most recent version of documentation, visit the Vyatta web site at vyatta.com.

PROPRIETARY NOTICES

Vyatta is a registered trademark of Vyatta, Inc.

VMware, VMware ESX, and VMware server are trademarks of VMware, Inc.

XenServer, and XenCenter are trademarks of Citrix Systems, Inc.

All other trademarks are the property of their respective owners.

RELEASE DATE: March 2012

DOCUMENT REVISION. R6.4 v01

RELEASED WITH: R6.4.0

PART NO. A0-0245-10-0002

Contents

Quick List of Commands

Use this list to help you quickly locate commands.

List of Examples

Use this list to help you locate examples you'd like to look at or try.

Preface

This document describes the various deployment, installation, and upgrade options for Vyatta software.

This preface provides information about using this guide. The following topics are presented:

- Intended Audience
- Organization of This Guide
- Document Conventions
- Vyatta Publications

Intended Audience

This guide is intended for experienced system and network administrators. Depending on the functionality to be used, readers should have specific knowledge in the following areas:

- Networking and data communications
- TCP/IP protocols
- General router configuration
- Routing protocols
- Network administration
- Network security
- IP services

Organization of This Guide

This guide has the following aid to help you find the information you are looking for:

- Quick List of Commands

  Use this list to help you quickly locate commands.

- List of Examples

  Use this list to help you locate examples you'd like to try or look at.

This guide has the following chapters:

Document Conventions

This guide uses the following advisory paragraphs, as follows.

WARNING *Warnings alert you to situations that may pose a threat to personal safety.*

CAUTION *Cautions alert you to situations that might cause harm to your system or damage to equipment, or that may affect service.*

NOTE *Notes provide information you might need to avoid problems or configuration errors.*

This document uses the following typographic conventions.

`Monospace`	Examples, command-line output, and representations of configuration nodes.
`bold Monospace`	Your input: something you type at a command line.
bold	Commands, keywords, and file names, when mentioned inline. Objects in the user interface, such as tabs, buttons, screens, and panes.
italics	An argument or variable where you supply a value.
<key>	A key on your keyboard, such as <Enter>. Combinations of keys are joined by plus signs (“+”), as in <Ctrl>+c.
[key1 | key2]	Enumerated options for completing a syntax. An example is [enable | disable].
num1–numN	A inclusive range of numbers. An example is 1–65535, which means 1 through 65535, inclusive.
arg1..*argN*	A range of enumerated values. An example is eth0..eth3, which means eth0, eth1, eth2, or eth3.
arg[*arg*...] *arg*[,*arg*...]	A value that can optionally represent a list of elements (a space-separated list and a comma-separated list, respectively).

Vyatta Publications

Full product documentation is provided in the Vyatta technical library. To see what documentation is available for your release, see the *Guide to Vyatta Documentation*. This guide is posted with every release of Vyatta software and provides a great starting point for finding the information you need.

Additional information is available on www.vyatta.com and www.vyatta.org.

Chapter 1: Connection Tracking

This chapter explains connection tracking in the Vyatta system.

This chapter presents the following topics:

- Connection Tracking Overview
- Connection Tracking Commands

Connection Tracking Overview

This section presents the following topics:

- Logging
- Connection Tracking Table Components
- Tuning Connection Tracking
- Setting Time-Outs for Connections

The Vyatta system can be configured to track connections using the connection tracking subsystem. Connection tracking becomes operational once either stateful firewall, NAT, WAN load balancing, web caching, or web filtering are configured.

Once configured, entries in the connection tracking table can be displayed using the show conntrack table command. Connection tracking entries can be removed from the connection tracking table using the delete conntrack table command. All entries in the connection tracking table can be removed using the reset conntrack command. Note that delete conntrack table and reset conntrack remove entries from the connection tracking table, but they do not clear the connection.

Logging

Connection events can be logged to the system log. The events to log for specific protocols are configured using the **system conntrack log** commands.

For each protocol type, connection tracking can log when a connection is created, when it is updated, and when it is terminated. For TCP, a connection is considered to be created once the 3-way TCP handshake occurs. For other IP protocols (for example, UDP and ICMP), the connection is considered to be created from a tracking perspective once the first packet of the flow is received. For all protocols, a connection is considered to be terminated when the connection table entry is removed for whatever reason, when there is a normal connection termination, when an RST flag is seen, or when an ICMP or UDP connection timeout occurs.

A separate process is created for each protocol or event configured. For example, a process is created if you configure the system to log new TCP connections. A separate process is created if you configure the system to log TCP connection terminations. Each configuration change restarts the process.

A 2 MB buffer (that is, a netlink socket buffer) is allocated for each process. If traffic is heavy enough to cause a buffer overflow, the system automatically increases the buffer size by 2 MB and restarts the process. This automatic reconfiguration continues to until the buffer reaches a maximum of 8 MB.

NOTE *There is a short time when the process is restarting where no events for that protocol/event type are captured.*

Connection Tracking Table Components

The connection tracking system consists of four components:

- The Connection Tracking Table
- The Connection Tracking Hash Table
- The Connection Tracking Expect Table
- The Connection Tracking Expect Hash Table

The Connection Tracking Table

The connection tracking table contains one entry for each connection being tracked by the system. Each entry is approximately 300 bytes and is dynamically allocated as required. The table has a maximum of 16,384 entries if the firewall is not enabled, and 32,768 entries if the firewall is enabled. This value can be changed using the system conntrack table-size <size> command in order to tune the system.

The Connection Tracking Hash Table

The connection tracking hash table makes searching the connection tracking table faster. The hash table uses "buckets" to record entries in the connection tracking table. By default, there are 4096 buckets in the table and each is 8 bytes.

Memory for the connection tracking hash table is statically allocated. The size of the connection tracking hash table can be tuned using the system conntrack hash-size <size> command. The larger the hash table size, the more static memory is used but the faster the lookup time. The smaller the hash table size, the lower the static memory usage but the slower the lookup time. Typically, the connection tracking hash table is kept at one-eight the number of entries in the connection tracking table.

The Connection Tracking Expect Table

The connection tracking expect table contains one entry for each expected connection related to an existing connection. These are generally used by "connection tracking helpers" (sometimes called "application-level gateways") for protocols such as FTP, SIP, and H.323.

Each entry is approximately 300 bytes and is dynamically allocated as required, up to a maximum of 2048 entries if the firewall is not enabled, and 4096 entries if the firewall is enabled. This value can be tuned using the system conntrack expect-table-size <size> command.

The Connection Tracking Expect Hash Table

The connection tracking expect hash table is used to make searching the connection tracking expect table faster. There are 1024 eight-byte buckets in the table. Memory for the connection tracking expect hash table is statically allocated. The size of the connection tracking expect hash table is not currently configurable.

Tuning Connection Tracking

For most installations, the default values of these tables will serve well. For high-capacity systems where the number of simultaneous connections is potentially greater than the connection tracking table can hold, the table sizes can be increased. When considering increasing table sizes, keep the following in mind:

- Each entry in the connection tracking table and the connection tracking expect table is approximately 300 bytes. This memory is dynamically allocated as required. At the same time, each bucket in the connection tracking hash table is eight bytes. This memory is statically allocated. For reasonable lookup speed, keep approximately one bucket in the connection tracking hash table for every eight entries in the connection tracking table.
- For better look-up performance, increase the size of the connection tracking hash table with respect to the connection tracking table. It does not make sense to bring the ratio for the size of these two tables closer than 1:1 (for example, if the connection tracking table is set to 65,536 then the maximum hash table size should not be greater than 65,536 as well).
- Calculate the maximum amount of memory that could be used by the connection tracking tables and make sure that you have enough free physical memory to accomodate them at their maximum size.

Setting Time-Outs for Connections

The Vyatta system supports setting timeouts for connections according to the connection type. You can set timeout values for generic connections, for ICMP connections, for high-stream or generic UDP connections, or for TCP connections in a number of different states. Define timeout values for connection types by using the system conntrack timeout icmp, system conntrack timeout tcp, system conntrack timeout udp, or system conntrack timeout other command.

You can also define custom timeout values to apply to a specific subset of connections, based on a packet and flow selector. To do this, you create a rule defining the packet and flow selector, using the system conntrack timeout custom command.

The selector for custom timeouts is a 5-tuple consisting of source address and port, destination address and port, and protocol. The options available for protocols within a custom timeout rule (for example, TCP states) are the same as those available for general connection type timeouts. Note that for packets matching a custom timeout rule, the custom timeout overrides any timeout set for the general connection type.

Connection Tracking Commands

Configuration Commands	
system conntrack expect-table-size <size>	Sets the maximum size of the connection tracking expect table.
system conntrack hash-size <size>	Sets the size of the hash table associated with the connection tracking table.
system conntrack log icmp	Specifies ICMP connection events to be logged.
system conntrack log other	Specifies connection events to be logged for protocols other than TCP, UDP, or ICMP.
system conntrack log tcp	Specifies TCP connection events to be logged.
system conntrack log udp	Specifies UDP connection events to be logged.
system conntrack modules sip	Sets options associated with tracking traffic related to SIP connections.
system conntrack table-size <size>	Sets the maximum size of the connection tracking table.
system conntrack tcp loose <state>	Specifies whether previously established connections are to be tracked for stateful traffic filtering.
system conntrack timeout custom	Defines a timeout value for sets of connections connections selected according to source, destination, and protocol.
system conntrack timeout icmp	Defines a timeout value for ICMP connections.
system conntrack timeout other	Defines a timeout value for connections that use protocols other than ICMP, TCP, or UDP.
system conntrack timeout tcp	Defines a timeout value for TCP connections.
system conntrack timeout udp	Defines a timeout value for UDP connections.
Operational Commands	
delete conntrack table	Deletes connection tracking table entries.
reset conntrack	Completely flushes the connection tracking table.
show conntrack table	Displays connection tracking table entries.

delete conntrack table

Deletes connection tracking table entries.

Syntax

delete conntrack table {**ipv4** | **ipv6**} [**source** *src-addr* [**destination** *dst-addr*]] [**quiet**]

Command Mode

Operational mode.

Parameters

ipv4	Delete IPv4 conntrack table entries. Either **ipv4** or **ipv6** must be specified.
ipv6	Delete IPv6 conntrack table entries. Either **ipv4** or **ipv6** must be specified.
src-addr	Delete conntrack entries whose source address matches this address. If **ipv4** is specified, the format is an IPv4 address, or **0.0.0.0** or the keyword **any** to represent any address. A port can be specified after the address using ":" followed by the port number. For example, "192.168.1.48:22" represents port 22 on IPv4 address 192.168.1.48. If **ipv6** is specified, the format is an IPv6 address, or **0::0** or the keyword **any** to represent any address. A port can be specified after the address using ":" followed by the port number. For example, "[2001:db8:2::2]:22" represents port 22 on IPv6 address 2001:db8:2::2. Note that square brackets are required around the IPv6 address (or the keyword **any**) if a port is specified.

dst-addr	Delete conntrack entries whose destination address matches this address. If **ipv4** is specified, the format is an IPv4 address, or **0.0.0.0** or the keyword **any** to represent any address. A port can be specified after the address using “:” followed by the port number. For example, “192.168.1.48:22” represents port 22 on IPv4 address 192.168.1.48. If **ipv6** is specified, the format is an IPv6 address, or **0::0** or the keyword **any** to represent any address. A port can be specified after the address using “:” followed by the port number. For example, “[2001:db8:2::2]:22” represents port 22 on IPv6 address 2001:db8:2::2. Note that square brackets are required around the IPv6 address if a port is specified.
quiet	Do not print log messages to the console or to the system log. Instead, create a single log entry that displays the parameters used in the **delete conntrack table** command. It is typically used when removing a large number of conntrack entries at once as it provents a potential flood of log messages.

Default

All IPv4 or IPv6 conntrack table entries are deleted. If a port number is specified, entries that use UDP or TCP protocols can be deleted. If no port is specified, then all protocol types can be deleted.

Usage Guidelines

Use this command to delete connection entries from the connection tracking table. Deleting a connection tracking entry does not prevent a new connection between the same source and destination from being created. If system conntrack tcp loose <state> is set to **enable** (as it is by default), any susbsequent data passed between the source and the destination will create a new entry in the connection tracking table. If it is set to **disable**, then susbsequent data passed between the source and destination will be in the INVALID state until a proper TCP three-way handshake establishes a new connection. A firewall rule that drops traffic in the INVALID state can stop this traffic. If you wish to permanently prevent connections between a given source and destination, you must create an explict firewall rule to do this.

NOTE *All conntrack table deletions are logged.*

Examples

Example 1-1 shows the output of the **delete conntrack table ipv4** command. In this case the command deletes all conntrack table entries where the source address is 192.168.1.21.

Example 1-1 “delete conntrack table ipv4” sample output

```
vyatta@vyatta:~$ delete conntrack table ipv4 source 192.168.1.21
Deleting the following conntrack table entries:

CONN ID    Source                  Destination          Protocol
3427168752 192.168.1.21:52250      192.168.1.81:22      tcp [6]
```

reset conntrack

Completely flushes the connection tracking table.

Syntax

reset conntrack

Command Mode

Operational mode.

Parameters

None.

Default

None.

Usage Guidelines

Use this command to flush all connections currently being tracked in the connection tracking table.

show conntrack table

Displays connection tracking table entries.

Syntax

show conntrack table {**ipv4** | **ipv6**} [**source** *src-addr* [**destination** *dst-addr*]]

Command Mode

Operational mode.

Parameters

ipv4	Display IPv4 conntrack table entries. Either **ipv4** or **ipv6** must be specified.
ipv6	Display IPv6 conntrack table entries. Either **ipv4** or **ipv6** must be specified.
src-addr	Conntrack entries whose source address matches this address are to be displayed. If **ipv4** is specified, the format is an IPv4 address, or **0.0.0.0** or the keyword **any** to represent any address. A port can be specified after the address using ":" followed by the port number. For example, "192.168.1.48:22" represents port 22 on IPv4 address 192.168.1.48. If **ipv6** is specified, the format is an IPv6 address, or **0::0** or the keyword **any** to represent any address. A port can be specified after the address using ":" followed by the port number. For example, "[2001:db8:2::2]:22" represents port 22 on IPv6 address 2001:db8:2::2. Note that square brackets are required around the IPv6 address (or the keyword **any**) if a port is specified.

dst-addr Conntrack entries whose destination address matches this address are to be displayed.

If **ipv4** is specified, the format is an IPv4 address, or **0.0.0.0** or the keyword **any** to represent any address. A port can be specified after the address using “:” followed by the port number. For example, “192.168.1.48:22” represents port 22 on IPv4 address 192.168.1.48.

If **ipv6** is specified, the format is an IPv6 address, or **0::0** or the keyword **any** to represent any address. A port can be specified after the address using “:” followed by the port number. For example, “[2001:db8:2::2]:22” represents port 22 on IPv6 address 2001:db8:2::2. Note that square brackets are required around the IPv6 address if a port is specified.

Default

All IPv4 or IPv6 conntrack table entries are displayed. If a port number is specified, entries that use UDP or TCP protocols can be shown. If no port is specified, then all protocol types can be shown.

Usage Guidelines

Use this command to display connections currently being tracked in the connection tracking table. Before connection tracking table entries can be displayed, one of the following system components must be configured: Firewall (stateful), NAT, Web Filtering, Web Caching, or WAN Load Balancing.

Examples

Example 1-2 shows the output of the **show conntrack table ipv4** command. In this case the command displays all connections where the destination port is 22. The source and destination addresses can be anything.

Example 1-2 “show conntrack table ipv4” sample output

```
vyatta@vyatta:~$ show conntrack table ipv4 source 0.0.0.0 destination
0.0.0.0:22
TCP state codes: SS - SYN SENT, SR - SYN RECEIVED, ES - ESTABLISHED,
                 FW - FIN WAIT, CW- CLOSE WAIT, LA - LAST ACK,
                 TW - TIME WAIT, CLOSE - CL, LISTEN - LI

CONN ID    Source              Destination         Protocol    TIMEOUT
3818626200 192.168.74.1:1140   192.168.74.128:22   tcp [6] ES  429809
3818625704 192.168.74.1:1145   192.168.74.200:22   tcp [6] ES  431878
```

```
    3818624216 10.3.0.182:1151     10.3.0.15:22          tcp [6] TW  90
```

Example 1-3 shows the output of the **show conntrack table ipv6** command. In this case the command displays all connections where the destination port is 22. The source and destination addresses can be anything.

Example 1-3 “show conntrack table ipv6” sample output

```
vyatta@vyatta:~$ show conntrack table ipv6 source 0:0:0:0:0:0:0:0 destination
[0:0:0:0:0:0:0:0]:22
CONN ID    Source                            Destination                       Protocol
3818626200 [10FB:0:0:0:C:ABC:1F0C:44DA]:1140 [10FB:0:0:0:C:ABC:1F0C:45AD]:22  tcp [6]
3818672537 [10FB:0:0:0:C:ABC:1F0C:55CB]:2020 [2001:cdba:0:0:0:0:3257:9652]:22 tcp [6]
```

system conntrack expect-table-size <size>

Sets the maximum size of the connection tracking expect table.

Syntax

set system conntrack expect-table-size *size*

delete system conntrack expect-table-size

show system conntrack expect-table-size

Command Mode

Configuration mode.

Configuration Statement

```
system {
   conntrack {
      expect-table-size size
   }
}
```

Parameters

size	The maximum number of entries allowed in the Netfilter connection tracking expect table. For memory usage estimating purposes, each entry, including overhead, uses approximately 300 bytes of kernel memory. The range is 1 to 50000000.

Default

When the firewall is not enabled, the connection tracking expect table is set to track a maximum of 2048 entries; when the firewall is enabled, the connection tracking expect table is set to track a maximum of 4096 entries. Since, each connection tracking expect table entry is about 300 bytes in size, the maximum amount of kernel memory used for connection tracking expect table entries could reach approximately 600 Kbytes [(2048 * 300)/(1024 * 1024)] when firewall is not enabled. Similarly, the maximum amount of kernel memory used for connection tracking expect table entries could reach a maximum of 1.2 Mbytes [(4096 * 300)/(1024 * 1024)] when the firewall is enabled.

Usage Guidelines

Use this command to specify the maximum size of the Netfilter connection tracking expect table. The connection tracking expect table is a table of connection tracking expectations. These are the mechanism by which connections related to existing connections are “expected”. They are generally used by "connection tracking helpers" (or “application level gateways”) for protocols such as FTP, SIP, and H.323.

If you intend to increase this value, then attention should be payed to the amount of memory available with the system and the approximate amount of memory that might get used by increasing this value.

Note that since memory for connection tracking expect table entries is dynamically allocated, memory usage will increase as the number of expected connections tracked by the system increases. Also, if the maximum number of entries is reached in the connection tracking table then the kernel may begin to drop existing connection tracking expect table entries to accommodate new entries or if it is unable to remove entries from the table then incoming packets may begin to be dropped.

Use the **set** form of this command to modify the maximum size of the connection tracking expect table.

Use the **delete** form of this command to restore the default connection tracking expect table size.

Use the **show** form of this command to view connection tracking expect table size configuration.

system conntrack hash-size <size>

Sets the size of the hash table associated with the connection tracking table.

Syntax

set system conntrack hash-size *size*

delete system conntrack hash-size

show system conntrack hash-size

Command Mode

Configuration mode.

Configuration Statement

```
system {
   conntrack {
      hash-size size
   }
}
```

Parameters

size	The number of buckets in the Netfilter connection tracking hash table. For memory usage estimating purposes, each entry, uses 8 bytes of kernel memory. The range is 1 to 50000000.

Default

The connection tracking hash table contains 4,096 buckets (32 Kbytes).

Usage Guidelines

Use this command to specify the size of the Netfilter connection tracking hash table. The connection tracking table hash table is the data structure used to provide quick seaching of the connection tracking table. The hash table is typically 1/8th the size of the connection tracking table. If the connection tracking table size is increased then the hash table should be increased as well in the same ratio. Making the hash table larger than that uses more memory but also increases the speed of accessing a connection entry. Making it smaller decreases the memory usage but slows down lookup time. Memory for connection tracking hash table entries is allocated statically.

Use the **set** form of this command to modify the size of the connection tracking hash table.

Use the **delete** form of this command to restore the default connection tracking hash table size.

Use the **show** form of this command to view connection tracking hash table size configuration.

system conntrack log icmp

Specifies ICMP connection events to be logged.

Syntax

set system conntrack log icmp {destroy | new | update}

delete system conntrack log icmp [destroy | new | update]

show system conntrack log icmp

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        log {
            icmp {
                destroy
                new
                update
            }
        }
    }
}
```

Parameters

destroy	Log when a ICMP connection is cleared. One of **destroy**, **new**, or **update**, must be specified.
new	Log when a ICMP connection is created. One of **destroy**, **new**, or **update**, must be specified.
update	Log updates to ICMP connections. One of **destroy**, **new**, or **update**, must be specified.

Default

None.

Usage Guidelines

Use this command to specify ICMP connection events to be logged.

Use the **set** form of this command to specify the ICMP connection events to be logged.

Use the **delete** form of this command to remove ICMP connection events from being logged.

Use the **show** form of this command to display the ICMP connection events to be logged.

Message Format

Log messages for ICMP connection events have the following message format:

<timestamp> <host-name> <Vyatta-log-tag>**:** [<event-type>] <protocol-name> <protocol-number> <timeout> **src=**<source-IP> **dst=**<destination-IP> **type=**<icmp-type> **code=**<icmp-code> **id=**<icmp-id> [<flow-status>] **src=**<source-IP-in-return-direction> **dst=**<destination-IP-in-return-direction> **type=**<icmp-type> **code=**<icmp-code> **id=**<icmp-id> **id=**<conntrack-connection-id>

NOTE *The <timeout> is not present for "DESTROY" events.*

Example 1-4 shows sample conntrack log messages for the ICMP protocol.

Example 1-4 Sample conntrack log messages for the ICMP protocol

```
Oct 20 17:53:25 Test5 log-conntrack: [NEW] icmp 1 30 src=192.168.249.10
dst=173.194.33.48 type=8 code=0 id=21851 [UNREPLIED] src=173.194.33.48
dst=10.3.0.183 type=0 code=0 id=21851 id=3973841888

Oct 20 17:53:25 Test5 log-conntrack: [UPDATE] icmp 1 30 src=192.168.249.10
dst=173.194.33.48 type=8 code=0 id=21851 src=173.194.33.48 dst=10.3.0.183
type=0 code=0 id=21851 id=3973841888

Oct 20 17:53:56 Test5 log-conntrack: [DESTROY] icmp 1 src=192.168.249.10
dst=173.194.33.48 type=8 code=0 id=21851 src=173.194.33.48 dst=10.3.0.183
type=0 code=0 id=21851 id=3973841888
```

system conntrack log other

Specifies connection events to be logged for protocols other than TCP, UDP, or ICMP.

Syntax

set system conntrack log other {destroy | new | update}

delete system conntrack log other [destroy | new | update]

show system conntrack log other

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        log {
            other {
                destroy
                new
                update
            }
        }
    }
}
```

Parameters

destroy	Log when a connection is cleared for a protocol other than TCP, UDP, or ICMP. One of **destroy**, **new**, or **update**, must be specified.
new	Log when a connection is created for a protocol other than TCP, UDP, or ICMP. One of **destroy**, **new**, or **update**, must be specified.
update	Log updates to connections for protocols other than TCP, UDP, and ICMP. One of **destroy**, **new**, or **update**, must be specified.

Default

None.

Usage Guidelines

Use this command to specify connection events to be logged for protocols other than TCP, UDP, and ICMP.

Use the **set** form of this command to specify the connection events to be logged for protocols other than TCP, UDP, and ICMP.

Use the **delete** form of this command to remove connection events from being logged for protocols other than TCP, UDP, and ICMP.

Use the **show** form of this command to display the connection events to be logged for protocols other than TCP, UDP, and ICMP.

Message Format

Log messages for connection events for protocols other than TCP, UDP, and ICMP, have the following message format:

<timestamp> <host-name> <Vyatta-log-tag>**:** [<event-type>] <protocol-name> <protocol-number> <timeout> **src=**<source-IP> **dst=**<destination-IP> [<flow-status>] **src=**<source-IP-in-return-direction> **dst=**<destination-IP-in-return-direction> [<flow-status>] **id=**<conntrack-connection-id>

NOTE *The <timeout> is not present for "DESTROY" events.*

For the GRE protocol, source and destination keys (**srckey**, and **dstkey**) are provided for packets in the original direction, as well as packets in the reply direction.

Example 1-5 shows sample conntrack log messages for protocols other than ICMP, TCP, or UDP.

Example 1-5 Sample conntrack log messages for other protocols

```
Dec 21 22:25:31 vyatta log-conntrack: [NEW] gre 47 30 src=192.169.100.75
dst=192.168.100.75 srckey=0x0 dstkey=0x0 [UNREPLIED] src=192.168.100.75
dst=192.169.100.75 srckey=0x0 dstkey=0x0 id=3998350488

Dec 21 22:38:06 vyatta log-conntrack: [UPDATE] gre 47 179
src=192.169.100.1 dst=192.168.100.1 srckey=0x0 dstkey=0x0
src=192.168.100.1 dst=192.169.100.1 srckey=0x0 dstkey=0x0 [ASSURED]
id=3998578376

Dec 21 22:39:50 vyatta log-conntrack: [DESTROY] gre 47 src=192.169.100.17
dst=192.168.100.17 srckey=0x0 dstkey=0x0 src=192.168.100.17
dst=192.169.100.17 srckey=0x0 dstkey=0x0 [ASSURED] id=4080054272
```

system conntrack log tcp

Specifies TCP connection events to be logged.

Syntax

set system conntrack log tcp {destroy | new | update {close_wait | established | fin_wait | last_ack | syn_received | time_wait}}

delete system conntrack log tcp [destroy | new | update [close_wait | established | fin_wait | last_ack | syn_received | time_wait]]

show system conntrack log tcp [destroy | new | update]

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        log {
            tcp {
                destroy
                new
                update {
                    close-wait
                    established
                    fin-wait
                    last-ack
                    syn-received
                    time-wait
                }
            }
        }
    }
}
```

Parameters

destroy	Log when a TCP connection is cleared. One of **destroy**, **new**, or **update**, must be specified.

new	Log when a TCP connection is created. One of **destroy**, **new**, or **update**, must be specified.
update close-wait	Log when a TCP connection enters the CLOSE_WAIT state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.
update established	Log when a TCP connection enters the ESTABLISHED state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.
update fin-wait	Log when a TCP connection enters the FIN_WAIT state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.
update last-ack	Log when a TCP connection enters the LAST_ACK state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.
update syn-received	Log when a TCP connection enters the SYN_RECV state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.
update time-wait	Log when a TCP connection enters the TIME_WAIT state. One of **close-wait**, **established**, **fin-wait**, **last-ack**, **syn-received**, or **time-wait**, must be specified.

Default

Log when a TCP connection enters the ESTABLISHED state.

Usage Guidelines

Use this command to specify TCP connection events to be logged.

Use the **set** form of this command to specify the TCP connection events to be logged.

Use the **delete** form of this command to remove TCP connection events from being logged.

Use the **show** form of this command to display the TCP connection events to be logged.

Message Format

Log messages for TCP connection events have the following message format:

<timestamp> <host-name> <Vyatta-log-tag>**:** [<event-type>] <protocol-name> <protocol-number> <timeout> <tcp-state> **src=**<source-IP> **dst=**<destination-IP> **sport=**<source-port> **dport=**<destination-port> [**<**flow-status**>**] **src=**<source-IP-in-return-direction> **dst=**<destination-IP-in-return-direction> **sport=**<source-port-in-return-direction> **dport=**<destination-port-in-return-direction> [**<**flow-status-in-return-direction**>**] **id=**<conntrack-connection-id>

NOTE *The <timeout> is not present for "DESTROY" events.*

Example 1-6 shows sample conntrack log messages for the ICMP protocol.

Example 1-6 Sample conntrack log messages for the ICMP protocol

```
Oct 20 17:48:59 Test5 log-conntrack: [NEW] tcp 6 120 SYN_SENT
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80 [UNREPLIED]
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 id=3973842632

Oct 20 17:48:59 Test5 log-conntrack: [UPDATE] tcp 6 60 SYN_RECV
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 id=3973842632

Oct 20 17:48:59 Test5 log-conntrack: [UPDATE] tcp 6 300 ESTABLISHED
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 [ASSURED]
id=3973842632

Oct 20 17:49:04 Test5 log-conntrack: [UPDATE] tcp 6 120 FIN_WAIT
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 [ASSURED]
id=3973842632

Oct 20 17:49:04 Test5 log-conntrack: [UPDATE] tcp 6 30 LAST_ACK
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 [ASSURED]
id=3973842632

Oct 20 17:49:04 Test5 log-conntrack: [UPDATE] tcp 6 120 TIME_WAIT
src=192.168.249.10 dst=74.125.224.151 sport=39082 dport=80
src=74.125.224.151 dst=10.3.0.183 sport=80 dport=39082 [ASSURED]
id=3973842632

Oct 20 17:51:04 Test5 log-conntrack: [DESTROY] tcp 6 src=192.168.249.10
dst=74.125.224.151 sport=39082 dport=80 src=74.125.224.151 dst=10.3.0.183
sport=80 dport=39082 [ASSURED] id=3973842632
```

system conntrack log udp

Specifies UDP connection events to be logged.

Syntax

set system conntrack log udp {destroy | new | update}

delete system conntrack log udp [destroy | new | update]

show system conntrack log udp

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        log {
            udp {
                destroy
                new
                update
            }
        }
    }
}
```

Parameters

destroy	Optional. Log when a UDP connection is cleared. One of **destroy**, **new**, or **update**, must be specified.
new	Optional. Log when a UDP connection is created. One of **destroy**, **new**, or **update**, must be specified.
update	Optional. Log updates to UDP connections. One of **destroy**, **new**, or **update**, must be specified.

Default

None.

Usage Guidelines

Use this command to specify UDP connection events to be logged.

Use the **set** form of this command to specify the UDP connection events to be logged.

Use the **delete** form of this command to remove UDP connection events from being logged.

Use the **show** form of this command to display the UDP connection events to be logged.

Message Format

Log messages for UDP connection events have the following message format:

<timestamp> <host-name> <Vyatta-log-tag>**:** [<event-type>] <protocol-name> <protocol-number> <timeout> **src=**<source-IP> **dst=**<destination-IP> **sport=**<source-port> **dport=**<destination-port> [**<**flow-status**>**] **src=**<source-IP-in-return-direction> **dst=**<destination-IP-in-return-direction> **sport=**<source-port-in-return-direction> **dport=**<destination-port-in-return-direction> **id=**<conntrack-connection-id>

NOTE *The <timeout> is not present for "DESTROY" events.*

Example 1-7 shows sample conntrack log messages for the ICMP protocol.

Example 1-7 Sample conntrack log messages for the ICMP protocol

```
Oct 20 17:56:04 test5 log-conntrack: [NEW] udp 17 30 src=192.168.249.10
dst=192.168.249.150 sport=48325 dport=53 [UNREPLIED] src=192.168.249.150
dst=192.168.249.10 sport=53 dport=48325 id=3973841889

Oct 20 17:56:04 test5 log-conntrack: [UPDATE] udp 17 30 src=192.168.249.10
dst=192.168.249.150 sport=48325 dport=53 src=192.168.249.150
dst=192.168.249.10 sport=53 dport=48325 id=3973841889

Oct 20 17:56:34 test5 log-conntrack: [DESTROY] udp 17 src=192.168.249.10
dst=192.168.249.150 sport=48325 dport=53 src=192.168.249.150
dst=192.168.249.10 sport=53 dport=48325 id=3973841889
```

system conntrack modules sip

Sets options associated with tracking traffic related to SIP connections.

Syntax

set system conntrack modules sip [**disable** | **enable-indirect-media** | **enable-indirect-signalling** | **port** *port*]

delete system conntrack modules sip [**disable** | **enable-indirect-media** | **enable-indirect-signalling** | **port**]

show system conntrack modules sip

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        modules {
            sip {
                disable
                enable-indirect-media
                enable-indirect-signalling
                port port
            }
        }
    }
}
```

Parameters

disable	Disable SIP connection tracking.
enable-indirect-media	Media streams can originate from, or be delivered to, addresses other than those used during the signalling (SIP) phase of the connection. By default, the connection tracking system only expects media streams using the same source/destination address pair as the SIP signalling stream.

enable-indirect-signalling	Incoming calls can come from an address other than the one a phone is registered with (typically the address of the PBX the phone registers with on boot). By default, the connection tracking system will only expect incoming calls to a phone from its registrar.
port *port*	Multinode. The port number that SIP traffic is carried on. Up to eight ports can be specified by creating additional **port** configuration nodes. The default is 5060. ***NOTE*** *If this parameter is set then only the port numbers specified will be tracked. If you wish to track port 5060 in addition to other ports then it must be specified explicitly along with the others you wish to track.*

Default

SIP traffic is carried on port 5060.

Usage Guidelines

Use this command to specify options associated with connection tracking SIP traffic.

Use the **set** form of this command to set options associated with connection tracking SIP traffic.

Use the **delete** form of this command to restore the default configuration.

Use the **show** form of this command to view the configuration.

system conntrack table-size <size>

Sets the maximum size of the connection tracking table.

Syntax

set system conntrack table-size *size*

delete system conntrack table-size

show system conntrack table-size

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        table-size size
    }
}
```

Parameters

size	The maximum number of entries allowed in the Netfilter connection tracking table. For memory usage estimating purposes, each entry, including overhead, uses approximately 300 bytes of kernel memory. The range is 1 to 50000000.

Default

When the firewall is not enabled, the connection tracking table is set to track a maximum of 16,384 entries; when the firewall is enabled, the connection tracking table is set to track a maximum of 32,768 entries. Since, each connection tracking entry is about 300 bytes in size, the maximum amount of kernel memory used for connection tracking entries could reach approximately 4.5 Mbytes [(16384 * 300)/(1024 * 1024)] when firewall is not enabled. Similarly, the maximum amount of kernel memory used for connection tracking entries could reach a maximum of 9 Mbytes [(32768 * 300)/(1024 * 1024)] when the firewall is enabled.

Usage Guidelines

Use this command to specify the maximum size of the Netfilter connection tracking table. The connection tracking table tracks the state of network connections and traffic streams, allowing the system to relate them to provide stateful traffic filtering.

If you intend to increase this value, then attention should be payed to the amount of memory available with the system and the approximate amount of memory that might get used by increasing this value.

Note that since memory for connection tracking entries is dynamically allocated, memory usage will increase as the number of connections tracked by the system increases. Also, if the maximum number of entries is reached in the connection tracking table then the kernel may begin to drop existing connection tracking entries to accommodate new entries or if it is unable to remove connection entries from the table then incoming packets may begin to be dropped.

NOTE *In most environments, if the connection tracking table size is modified, the connection tracking hash table size (**conntrack-hash-size**) should also be modified so that it remains 1/8th the size of the connection tracking table.*

Use the **set** form of this command to modify the maximum size of the connection tracking table.

Use the **delete** form of this command to restore the default connection tracking table size.

Use the **show** form of this command to view connection tracking table size configuration.

system conntrack tcp loose <state>

Specifies whether previously established connections are to be tracked for stateful traffic filtering.

Syntax

set system conntrack tcp loose {**enable** | **disable**}

delete system conntrack tcp loose

show system conntrack tcp loose

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        tcp {
            loose [enable|disable]
        }
    }
}
```

Parameters

enable	The system allows the processing of traffic sent on previously established connections.
disable	The system does not allow the processing of traffic sent on previously established connections.

Default

The system allows the processing of traffic sent on previously established connections.

Usage Guidelines

Use this command to specify whether loose TCP tracking is to be applied; that is, whether traffic sent on previously established connections should be allowed in stateful traffic filtering.

In stateful traffic filtering, the system retains the state of data flows authorized from the trusted network. When loose TCP connection tracking is enabled, the system permits traffic on flows that were established previously; when disabled, the system rejects these flows.

Use the **set** form of this command to specify whether traffic on previously established connections are allowed or rejected.

Use the **delete** form of this command to restore the default behavior.

Use the **show** form of this command to view loose TCP tracking configuration.

system conntrack timeout custom

Defines a timeout value for sets of connections connections selected according to source, destination, and protocol.

Syntax

set system conntrack timeout custom rule *rule-num* {**destination** {**address** *ip-addr* **|** **port** *port-num*} **|** **source** {**address** *ip-addr* **|** **port** *port-num*} **|** **protocol** {**icmp** *timeout* **|** **other** *timeout* **|** **tcp** {**close** *timeout* **|** **close-wait** *timeout* **|** **established** *timeout* **|** **fin-wait** *timeout* **|** **last-ack** *timeout* **|** **syn-received** *timeout* **|** **syn-sent** *timeout* **|** **time-wait** *timeout*} **|** **udp** {**other** *timeout* **|** **stream** *timeout*}}

delete system conntrack timeout rule *rule-num* [**destination** [**address** **|** **port**] **|** **source** [**address** **|** **port**] **|** **protocol** [**icmp** **|** **other** **|** **tcp** [**close** **|** **close-wait** **|** **established** **|** **fin-wait** **|** **last-ack** **|** **syn-received** **|** **syn-sent** **|** **time-wait**] **|** **udp** [**other** **|** **stream**]]

show system conntrack timeout custom rule *rule-num* [**destination** [**address** **|** **port**] **|** **source** [**address** **|** **port**] **|** **protocol** [**icmp** **|** **other** **|** **tcp** [**close** **|** **close-wait** **|** **established** **|** **fin-wait** **|** **last-ack** **|** **syn-received** **|** **syn-sent** **|** **time-wait**] **|** **udp** [**other** **|** **stream**]]

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        timeout {
            custom
                rule rule-num
                    destination {
                        address ip-addr
                        port port-num
                    }
                    source {
                        address ip-addr
                        port port
                    }
                    protocol
                        icmp timeout
                        other timeout
                        tcp {
                            close timeout
                            close-wait timeout
                            established timeout
```

```
                         fin-wait timeout
                         last-ack timeout
                         syn-received timeout
                         syn-sent timeout
                         time-wait timeout
                     }
                     udp {
                         stream timeout
                         other timeout
                     }
       }
    }
}
```

Parameters

rule rule-num	An integer uniquely identifying the rule. Rules are executed in numeric order, and the first rule matched is executed.
address ip-addr	The source or destination IP address. The format is an IPv4 address in dotted decimal format or a hyphen-separated range of contiguous addresses.
port port-num	The source or destination port number or a hyphen-separated range of port numbers. The range is 0 to 65535..
icmp *timeout*	The amount of time, in seconds, to wait for the specified ICMP reply packet before considering the ICMP connection as terminated. The range is 1 to 21474836. The default is 30.
other *timeout*	The amount of time, in seconds, to wait for a response in a protocol other than ICMP, TCP, or UDP before considering the connection terminated. The range is 1 to 21474836. The default is 600 (10 minutes).
close *timeout*	The amount of time, in seconds, to wait in the CLOSE state before timing out. The range is 1 to 21474836. The default is 10.
close-wait *timeout*	The amount of time, in seconds, to wait in the CLOSE-WAIT state before timing out. The range is 1 to 21474836. The default is 60.

established *timeout*	The amount of time, in seconds, to wait in the ESTABLISHED state before timing out. The range is 1 to 21474836. The default is 432000 (5 days).
fin-wait *timeout*	The amount of time, in seconds, to wait in the FIN-WAIT state before timing out. The range is 1 to 21474836. The default is 120.
last-ack *timeout*	The amount of time, in seconds, to wait in the LAST-ACK state before timing out. The range is 1 to 21474836. The default is 30.
syn-received *timeout*	The amount of time, in seconds, to wait in the SYN-RECEIVED state before timing out. The range is 1 to 21474836. The default is 60.
syn-sent *timeout*	The amount of time, in seconds, to wait in the SYN-SENT state before timing out. The range is 1 to 21474836. The default is 120.
time-wait *timeout*	The amount of time, in seconds, to wait in the TIME-WAIT state before timing out. The range is 1 to 21474836. The default is 120.

Default

None

Usage Guidelines

Use this command to define a timeout value to be applied to a specific subset of connections.

The subset of connections affected by the timeout is based on a packet and flow selector. The selector is defined within a rule, using a 5-tuple consisting of source address and port, destination address and port, and protocol. Rules are executed in order, according to the numeric identifier. The timeout value from the first matched rule is applied to the packet or flow.

The protocol options available within a custom timeout rule (for example, TCP states) are the same as those available for general connection type timeouts. Note that for packets matching a custom timeout rule, the custom timeout overrides any timeout set for the general connection type.

Use the **set** form of this command to define a rule for applying a custom timeout for specific subsets of connections.

Use the **delete** form of this command to delete a custom timeout rule. In this case, the default timeout value for the general connection type comes back into effect.

Use the **show** form of this command to view defined custom connection tracking timeout rules.

system conntrack timeout icmp

Defines a timeout value for ICMP connections.

Syntax

set system conntrack timeout icmp *timeout*

delete system conntrack timeout icmp

show system conntrack timeout icmp

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        timeout {
            icmp timeout
        }
    }
}
```

Parameters

timeout	The amount of time, in seconds, to wait for an ICMP reply packet before considering the ICMP connection as terminated. The range is 1 to 21474836. The default is 30.

Default

The connection tracking system will wait for an ICMP reply packet for 30 seconds before considering the connection terminated.

Usage Guidelines

Use this command to specify the amount of time to wait for an ICMP reply packet before considering the “connection” (which in this case is an expected message sequence) terminated. Replies are expected for echo requests, timestamp requests, information requests, and address mask requests.

Use the **set** form of this command to specify the connection tracking timeout for ICMP replies.

Use the **delete** form of this command to remove the connection tracking timeout for ICMP replies and restore the default behavior.

Use the **show** form of this command to view the connection tracking timeout for ICMP replies.

system conntrack timeout other

Defines a timeout value for connections that use protocols other than ICMP, TCP, or UDP.

Syntax

set system conntrack timeout other *timeout*

delete system conntrack timeout other

show system conntrack timeout other

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        timeout {
            other timeout
        }
    }
}
```

Parameters

timeout	The amount of time, in seconds, to wait for a response before considering the connection terminated. The range is 1 to 21474836. The default is 600 (10 minutes).

Default

The connection tracking system waits for a response for 600 seconds before considering the connection terminated.

Usage Guidelines

Use this command to specify the amount of time to wait for a response before considering the “connection” terminated.

Use the **set** form of this command to specify the connection tracking timeout for replies for protocols other than TCP, UDP, or ICMP.

Use the **delete** form of this command to remove the connection tracking timeout for replies for protocols other than TCP, UDP, or ICMP, and restore the default behavior.

Use the **show** form of this command to view the connection tracking timeout for replies for protocols other than TCP, UDP, or ICMP.

system conntrack timeout tcp

Defines a timeout value for TCP connections.

Syntax

set system conntrack timeout tcp {**close** *timeout* | **close-wait** *timeout* | **established** *timeout* | **fin-wait** *timeout* | **last-ack** *timeout* | **syn-received** *timeout* | **syn-sent** *timeout* | **time-wait** *timeout*}

delete system conntrack timeout tcp [**close** | **close-wait** | **established** | **fin-wait** | **last-ack** | **syn-received** | **syn-sent** | **time-wait**]

show system conntrack timeout tcp [**close** | **close-wait** | **established** | **fin-wait** | **last-ack** | **syn-received** | **syn-sent** | **time-wait**]

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        timeout {
            tcp {
                close timeout
                close-wait timeout
                established timeout
                fin-wait timeout
                last-ack timeout
                syn-received timeout
                syn-sent timeout
                time-wait timeout
            }
        }
    }
}
```

Parameters

close *timeout*	The amount of time, in seconds, to wait in the CLOSE state before timing out. The range is 1 to 21474836. The default is 10.

close-wait *timeout*	The amount of time, in seconds, to wait in the CLOSE-WAIT state before timing out. The range is 1 to 21474836. The default is 60.
established *timeout*	The amount of time, in seconds, to wait in the ESTABLISHED state before timing out. The range is 1 to 21474836. The default is 432000 (5 days).
fin-wait *timeout*	The amount of time, in seconds, to wait in the FIN-WAIT state before timing out. The range is 1 to 21474836. The default is 120.
last-ack *timeout*	The amount of time, in seconds, to wait in the LAST-ACK state before timing out. The range is 1 to 21474836. The default is 30.
syn-received *timeout*	The amount of time, in seconds, to wait in the SYN-RECEIVED state before timing out. The range is 1 to 21474836. The default is 60.
syn-sent *timeout*	The amount of time, in seconds, to wait in the SYN-SENT state before timing out. The range is 1 to 21474836. The default is 120.
time-wait *timeout*	The amount of time, in seconds, to wait in the TIME-WAIT state before timing out. The range is 1 to 21474836. The default is 120.

Default

None.

Usage Guidelines

Use this command to specify the amount of time a TCP connection can be in a specific state state before it times out.

Use the **set** form of this command to specify the TCP connection state timeout.

Use the **delete** form of this command to restore the TCP connection state timeout to the default value.

Use the **show** form of this command to view the TCP connection timeout.

system conntrack timeout udp

Defines a timeout value for UDP connections.

Syntax

set system conntrack timeout udp {**stream** *timeout* | **other** *timeout*}

delete system conntrack timeout udp [**stream** | **other**]

show system conntrack timeout udp other

Command Mode

Configuration mode.

Configuration Statement

```
system {
    conntrack {
        timeout {
            udp {
                stream timeout
                other timeout
            }
        }
    }
}
```

Parameters

stream *timeout*	The amount of time, in seconds, for a UDP stream that has reached the assured state to wait for a packet before timing out. The range is 1 to 21474836. The default is 180. You should ensure that this values is larger than any set using the **other** option.
other *timeout*	The amount of time, in seconds, the UDP connection waits in the ESTABLISHED state before timing out. The range is 1 to 21474836. The default is 30.

Default

None.

Usage Guidelines

Use this command to specify the amount of time a UDP connection can....

Once a UDP connection is initiated and enters the ESTABLISHED state, it stays in this state until a few packets are transmitted back and forth. At that point, the connection becomes assured and is considered a stream.

- Use the **other** option to set timeout for the initial packet exchange.
- Use the **stream** option to set the timeout for an assured UDP stream.

Use the **set** form of this command to specify UDP connection timeout.

Use the **delete** form of this command to restore the default value for UDP timeout.

Use the **show** form of this command to view UDP timeout configuration.

Chapter 2: Flow Accounting

This chapter explains how to configure flow accounting using the Vyatta system.

This chapter presents the following topics:

Flow Accounting Configuration

This section presents the following topics:

- Flow Accounting Overview
- Configuring an Interface for Flow Accounting
- Displaying Flow Accounting Information
- Exporting Flow Accounting information

Flow Accounting Overview

Flow accounting provides the ability to locally display information about network traffic, as well as the ability to export this information to Netflow- or sFlow-compatible collection servers.

A network flow is defined as a unidirectional sequence of packets all of which have a common source IP address, destination IP address, source port (for UDP or TCP, 0 for other protocols), destination port (for UDP or TCP, type and code for ICMP, 0 for other protocols), IP protocol, ingress interface, and Type of Service.

Each separate TCP session with identical network flow information is counted as a new flow in the statistics. A TCP flow is considered complete if its session completes or the flow times out. There are a number of available timeout values that can be configured, as required.

For connectionless protocols like ICMP and UDP, a flow is considered complete after no packets for that flow appear for a configurable timeout period.

Flow accounting is defined on a per-interface basis. All packets recieved by the interface can be counted, resulting in very precise statistics. However, viewing all packets consumes significant computing resources. An alternative is to sample every *n* packets (the sampling rate) and to estimate data traffic based on these samples. This consumes fewer system resources than viewing all packets, especially for large data volumes, while still providing reasonable accuracy.

Configuring an Interface for Flow Accounting

In order for flow accounting information to be collected and displayed for an interface, the interface must first be configured for flow accounting. The following example shows how to configure eth0 for flow accounting in configuration mode.

Example 2-1 Configuring an interface for flow accounting

Step	Command
Configure flow accounting on eth0.	`vyatta@vyatta# set system flow-accounting interface eth0`

Example 2-1 Configuring an interface for flow accounting

Step	Command
Commit the configuration.	`vyatta@vyatta# commit`

Displaying Flow Accounting Information

Once flow accounting is configured on selected interfaces it provides the ability to display network traffic information for all configured interfaces, by interface, by interface and host, by interface and port, as well as by traffic volume on an interface. The following operational mode example shows flow accounting for eth0.

Example 2-2 Showing flow accounting information for eth0

```
vyatta@vyatta:~$ show flow-accounting interface eth0
flow-accounting for [eth0]
Src Addr        Dst Addr        Sport Dport Proto    Packets      Bytes    Flows
192.168.1.156   192.168.1.80    3024  22      tcp         98       6520        0
192.168.1.8     255.255.255.255 22936 2220    udp          2        696        1
192.168.1.8     255.255.255.255 22936 3245    udp          2        696        1
192.168.1.8     255.255.255.255 22936 2214    udp          2        696        1
192.168.1.8     255.255.255.255 22936 3242    udp          2        696        1
192.168.1.156   192.168.1.255   138   138     udp          2        480        1
192.168.1.8     192.168.1.255   138   138     udp          1        240        1
192.168.1.10    192.168.1.255   2214  22936   udp          4        240        1
192.168.1.156   192.168.1.255   3245  22936   udp          4        240        1
192.168.1.10    192.168.1.255   2220  22936   udp          4        240        1
192.168.1.156   192.168.1.255   3242  22936   udp          4        240        1
192.168.1.8     192.168.1.255   137   137     udp          1         78        1

Total entries: 12
Total flows  : 11
Total pkts   : 126
Total bytes  : 11,062
vyatta@vyatta:~$
```

The following example shows flow accounting for host 192.168.1.156 on eth0.

Example 2-3 Showing flow accounting information for 192.168.1.156 on eth0

```
vyatta@vyatta:~$ show flow-accounting interface eth0 host 192.168.1.156
Src Addr        Dst Addr        Sport Dport Proto    Packets      Bytes    Flows
192.168.1.156   192.168.1.80    3024  22      tcp        107       7036        0
192.168.1.156   192.168.1.255   138   138     udp          2        480        1
192.168.1.156   192.168.1.255   3245  22936   udp          4        240        1
```

```
192.168.1.156   192.168.1.255   3242  22936   udp          4        240       1

Total entries: 4
Total flows  : 3
Total pkts   : 117
Total bytes  : 7,996
vyatta@vyatta:~$
```

Exporting Flow Accounting information

In addition to displaying flow accounting information locally, this information can be exported to a collection server. The following example shows how to configure the system to export flow accounting information in Netflow format to a collection server with IP address 192.168.1.20 on the default port.

Example 2-4 Exporting data in Netflow format to 192.168.1.20

Step	Command
Configure the export of data in Netflow format to 192.168.1.20.	`vyatta@vyatta#` **`set system flow-accounting netflow server 192.168.1.20`**
Commit the configuration.	`vyatta@vyatta#` **`commit`**

Flow Accounting Commands

This section presents the following commands.

Configuration Commands	
system flow-accounting interface <interface>	Specifies the interface on which to record inbound flow statistics.
system flow-accounting netflow engine-id <id>	Specifies the system ID to appear in Netflow data.
system flow-accounting netflow sampling-rate <rate>	Specifies the rate at which packets are sampled for statistics.
system flow-accounting netflow server <ipv4>	Specifies a Netflow collector to which to export Netflow data.
system flow-accounting netflow timeout expiry-interval <interval>	Specifies the interval at which Netflow data will be sent to a Netflow collector.
system flow-accounting netflow timeout flow-generic <timeout>	Specifies the flow timeout for generic IP traffic.
system flow-accounting netflow timeout icmp <timeout>	Specifies the flow timeout for ICMP traffic.
system flow-accounting netflow timeout max-active-life <life>	Specifies the maximum time for which any flow can have data collected.
system flow-accounting netflow timeout tcp-fin <timeout>	Specifies the TCP flow timeout after receiving a TCP FIN packet.
system flow-accounting netflow timeout tcp-generic <timeout>	Specifies the generic TCP flow timeout.
system flow-accounting netflow timeout tcp-rst <timeout>	Specifies the TCP flow timeout after receiving a TCP RST packet.
system flow-accounting netflow timeout udp <timeout>	Specifies the flow timeout for UDP traffic.
system flow-accounting netflow version <version>	Specifies the Netflow format that data will be exported in.
system flow-accounting sflow agent-address <addr>	Allows you to specify the IP address of the sFlow agent.
system flow-accounting sflow sampling-rate <rate>	Specifies the rate at which sFlow statistics are recorded.
system flow-accounting sflow server <ipv4>	Specifies an sflow collector to export sFlow data to.

Configuration Commands	
system flow-accounting syslog-facility <facility>	Specifies the kinds of flow accounting messages to be logged.

Operational Commands	
clear flow-accounting counters	Clears all flow accounting counters.
restart flow-accounting	Restarts the flow accounting process.
show flow-accounting	Displays flow statistics for all interfaces on which flow accounting is enabled.
show flow-accounting interface <interface>	Displays flow statistics for a specific interface configured for flow accounting.

clear flow-accounting counters

Clears all flow accounting counters.

Syntax

clear flow-accounting counters

Command Mode

Operational mode.

Parameters

None.

Default

None.

Usage Guidelines

Use this command to clear flow accounting counters on all configured interfaces.

restart flow-accounting

Restarts the flow accounting process.

Syntax

restart flow-accounting process

Command Mode

Operational mode.

Parameters

None.

Default

None.

Usage Guidelines

Use this command to restart the flow accounting process.

show flow-accounting

Displays flow statistics for all interfaces on which flow accounting is enabled.

Syntax

show flow-accounting

Command Mode

Operational mode.

Parameters

None.

Default

None.

Usage Guidelines

Use this command to display flow statistics for all interfaces configured for flow accounting. Statistics are displayed for each interface configured for flow accounting.

show flow-accounting interface <interface>

Displays flow statistics for a specific interface configured for flow accounting.

Syntax

show flow-accounting interface *interface* [**host** *host*] [**port** *port*] [**top** *top*]

Command Mode

Operational mode.

Parameters

interface	The interface from which to obtain flow statistics (for example, eth0). This interface must first be configured for flow accounting.
host	The IP address of a specific host whose flow statistics are to be displayed.
port	The port number of a specific port whose flow statistics are to be displayed.
top	The number of flows with the heaviest traffic to be displayed. They are displayed in decending order based on the number of bytes received on the interface.

Default

None.

Usage Guidelines

Use this command to display flow statistics for the specified interface. The interface must first be configured for flow accounting.

system flow-accounting interface <interface>

Specifies the interface on which to record inbound flow statistics.

Syntax

set system flow-accounting interface *interface*

delete system flow-accounting interface *interface*

show system flow-accounting *interface*

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      interface interface
   }
}
```

Parameters

interface	Multi-node. The interface on which to record inbound flow statistics (for example, eth0). You can enable multiple interfaces for flow accounting by creating multiple **interface** configuration nodes.

Default

None.

Usage Guidelines

Use this command to configure an interface to record inbound flow statistics.

Use the **set** form of this command to configure an interface to record inbound flow statistics.

Use the **delete** form of this command to stop an interface from recording inbound flow statistics.

Use the **show** form of this command to show the interfaces configured to record inbound flow statistics.

system flow-accounting netflow engine-id <id>

Specifies the system ID to appear in Netflow data.

Syntax

set system flow-accounting netflow engine-id *id*

delete system flow-accounting netflow engine-id

show system flow-accounting netflow engine-id

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      netflow {
         engine-id id
      }
   }
}
```

Parameters

id	The system ID that will appear in Netflow data indentifying the router that the data came from. The range is 0 to 255.

Default

None.

Usage Guidelines

Use this command to configure the system ID to appear in Netflow data.

Use the **set** form of this command to configure the system ID to appear in Netflow data.

Use the **delete** form of this command to remove the system ID configuration.

Use the **show** form of this command to view the configuration.

system flow-accounting netflow sampling-rate <rate>

Specifies the rate at which packets are sampled for statistics.

Syntax

set system flow-accounting netflow sampling-rate *rate*

delete system flow-accounting netflow sampling-rate

show system flow-accounting netflow sampling-rate

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            sampling-rate rate
        }
    }
}
```

Parameters

rate	The rate at which packets are sampled (that is, if 1 in *n* packets are sampled, *n* is the rate).

Default

Every packet is sampled (that is, the sampling rate is 1).

Usage Guidelines

Use this command to configure the Netflow sampling rate for flow accounting. The system samples one in every *n* packets, where *n* is the value configured for the **sampling-rate** option.

The advantage of sampling every *n* packets, where *n* > 1, allows you to decrease the amount of processing resources required for flow accounting. The disadvantage of not sampling every packet is that the statistics produced are estimates of actual data flows.

Use the **set** form of this command to specfy the sampling rate.

Use the **delete** form of this command to sample all packets.

Use the **show** form of this command to display sampling rate configuration.

system flow-accounting netflow server <ipv4>

Specifies a Netflow collector to which to export Netflow data.

Syntax

set system flow-accounting netflow server *ipv4* [**port** *port*]

delete system flow-accounting netflow server *ipv4* [**port**]

show system flow-accounting netflow server *ipv4* [**port**]

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      netflow {
         server ipv4 {
            port port
         }
      }
   }
}
```

Parameters

ipv4	Multi-node. The IP address of a Netflow collector to which to export the Netflow data. You can export Netflow data to more than one collector by issuing this command multiple times.
port	The port on the Netflow collector to which to export the Netflow. The default value is 2055.

Default

None.

Usage Guidelines

Use this command to specify a Netflow collector for exporting flow accounting data.

Use the **set** form of this command to specify a Netflow collector.

Use the **delete** form of this command to remove a Netflow collector configuration.

Use the **show** form of this command to display Netflow collector configuration.

system flow-accounting netflow timeout expiry-interval <interval>

Specifies the interval at which Netflow data will be sent to a Netflow collector.

Syntax

set system flow-accounting netflow timeout expiry-interval *interval*

delete system flow-accounting netflow timeout expiry-interval

show system flow-accounting netflow timeout expiry-interval

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            timeout {
                expiry-interval interval
            }
        }
    }
}
```

Parameters

interval	The interval at which Netflow data will be sent to a Netflow collector.

Default

Netflow data will be sent every 60 seconds.

Usage Guidelines

Use this command to configure the interval at which the system will send Netflow data to a Neflow collector. The Netflow collector must first be configured using the system flow-accounting netflow server <ipv4> command.

Use the **set** form of this command to configure the interval at which the system will send Netflow data to a Neflow collector.

Use the **delete** form of this command to return the system to the default value interval.

Use the **show** form of this command to view the configuration.

system flow-accounting netflow timeout flow-generic <timeout>

Specifies the flow timeout for generic IP traffic.

Syntax

set system flow-accounting netflow timeout flow-generic *timeout*

delete system flow-accounting netflow timeout flow-generic

show system flow-accounting netflow timeout flow-generic

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      netflow {
         timeout {
            flow-generic timeout
         }
      }
   }
}
```

Parameters

timeout	The flow timeout, in seconds, for generic IP traffic. This includes all IP traffic except TCP, UDP, and ICMP. The range is 1 to 4294967296. The default value is 3600 (1 hour).

Default

Generic IP traffic flows time out after 3600 seconds.

Usage Guidelines

Use this command to configure the flow timeout for generic IP traffic. Generic IP traffic consists of all IP traffic except TCP, UDP, and ICMP. (Generic IP traffic would include, for example, GRE, AH, ESP, and so on.)

This parameter defines the amount of time the system continues to wait for data from a generic IP flow before considering the flow complete.

Use the **set** form of this command to set the flow timeout for generic IP traffic.

Use the **delete** form of this command to return the flow timeout for generic IP traffic to the default value.

Use the **show** form of this command to view generic IP traffic flow timeout configuration.

system flow-accounting netflow timeout icmp <timeout>

Specifies the flow timeout for ICMP traffic.

Syntax

set system flow-accounting netflow timeout icmp *timeout*

delete system flow-accounting netflow timeout icmp

show system flow-accounting netflow timeout icmp

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            timeout {
                icmp timeout
            }
        }
    }
}
```

Parameters

timeout	The flow timeout, in seconds, for ICMP traffic. The range is 1 to 4294967296. The default value is 300 (5 minutes).

Default

ICMP traffic flows timeout after 300 seconds.

Usage Guidelines

Use this command to configure the flow timeout for ICMP traffic. This parameter defines the amount of time the system continues to wait for data from an ICMP flow before considering the flow complete.

Use the **set** form of this command to set the flow timeout for ICMP traffic.

Use the **delete** form of this command to to return the flow timeout for ICMP traffic to the default value.

Use the **show** form of this command to view ICMP traffic flow timeout configuration.

system flow-accounting netflow timeout max-active-life <life>

Specifies the maximum time for which any flow can have data collected.

Syntax

set system flow-accounting netflow timeout max-active-life *life*

delete system flow-accounting netflow timeout max-active-life

show system flow-accounting netflow timeout max-active-life

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            timeout {
                max-active-life life
            }
        }
    }
}
```

Parameters

life	The global flow timeout, in seconds. The range is 1 to 4294967296. The default value is 604800 (7 days).

Default

All flows time out after 604,800 seconds.

Usage Guidelines

Use this command to configure the global flow timeout.

This parameter defines the amount of time the system continues to wait for data from any flow before considering the flow complete. Even if the flow is still active when it reaches this timeout value, it will be considered complete from a flow accounting perspective.

Use the **set** form of this command to set the global flow timeout.

Use the **delete** form of this command to to return the global flow timeout to the default value.

Use the **show** form of this command to view global flow timeout configuration.

system flow-accounting netflow timeout tcp-fin <timeout>

Specifies the TCP flow timeout after receiving a TCP FIN packet.

Syntax

set system flow-accounting netflow timeout tcp-fin *timeout*

delete system flow-accounting netflow timeout tcp-fin

show system flow-accounting netflow timeout tcp-fin

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            timeout {
                tcp-fin timeout
            }
        }
    }
}
```

Parameters

timeout	The flow timeout, in seconds, after receiving a TCP FIN packet. The range is 1 to 4294967296. The default value is 300 (5 minutes).

Default

A TCP flow times out 300 seconds after receiving a TCP FIN packet without receiving the corresponding FIN ACK, ACK sequence.

Usage Guidelines

Use this command to configure the TCP flow timeout after receiving a TCP FIN packet. This parameter defines the amount of time the system continues to wait for data from a TCP flow after receiving a TCP FIN packet without having received the corresponding FIN ACK, ACK sequence. When this timeout expires, the flow is considered complete.

Use the **set** form of this command to set the TCP FIN flow timeout.

Use the **delete** form of this command to return the TCP FIN flow timeout to the default value.

Use the **show** form of this command to view TCP FIN timeout configuration.

system flow-accounting netflow timeout tcp-generic <timeout>

Specifies the generic TCP flow timeout.

Syntax

set system flow-accounting netflow timeout tcp-generic *timeout*

delete system flow-accounting netflow timeout tcp-generic

show system flow-accounting netflow timeout tcp-generic

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            timeout {
                tcp-generic timeout
            }
        }
    }
}
```

Parameters

timeout	The generic TCP flow timeout, in seconds. The range is 1 to 4294967296. The default value is 3600 (1 hour).

Default

A TCP flow will timeout 3600 seconds after seeing no data or TCP FIN, FIN ACK, ACK sequence.

Usage Guidelines

Use this command to configure the TCP flow timeout after seeing no data or TCP FIN, FIN ACK, ACK sequence. This parameter defines the amount of time the system will continue to wait for data from a TCP flow without seeing any data, or a TCP FIN, and the corresponding FIN ACK, ACK sequence, before considering the flow complete.

Use the **set** form of this command to set the generic TCP flow timeout.

Use the **delete** form of this command to return the generic TCP flow timeout to the default value.

Use the **show** form of this command to view the configuration.

system flow-accounting netflow timeout tcp-rst <timeout>

Specifies the TCP flow timeout after receiving a TCP RST packet.

Syntax

set system flow-accounting netflow timeout tcp-rst *timeout*

delete system flow-accounting netflow timeout tcp-rst

show system flow-accounting netflow timeout tcp-rst

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      netflow {
         timeout {
            tcp-rst timeout
         }
      }
   }
}
```

Parameters

timeout	The flow timeout after receiving a TCP RST packet. The range is 1 to 4294967296. The default value is 120 (2 minutes).

Default

A TCP flow will timeout 120 seconds after seeing a TCP RST packet without seeing any other packets (i.e. data, TCP FIN, FIN ACK, or ACK).

Usage Guidelines

Use this command to configure the TCP flow timeout after seeing a TCP RST packet but no data,TCP FIN, FIN ACK, or ACK. This parameter defines the amount of time the system will continue to wait for data from a TCP flow after seeing a TSCP RST but without seeing any data, TCP FIN, FIN ACK, or ACK packets, before considering the flow complete.

Use the **set** form of this command to set the TCP RST flow timeout.

Use the **delete** form of this command to return the TCP RST flow timeout to the default value.

Use the **show** form of this command to view the configuration.

system flow-accounting netflow timeout udp <timeout>

Specifies the flow timeout for UDP traffic.

Syntax

set system flow-accounting netflow timeout udp *timeout*

delete system flow-accounting netflow timeout udp

show system flow-accounting netflow timeout udp

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      netflow {
         timeout {
            udp timeout
         }
      }
   }
}
```

Parameters

timeout	The flow timeout for UDP traffic.The range is 1 to 4294967296. The default value is 300 (5 minutes).

Default

UDP traffic flows timeout after 300 seconds.

Usage Guidelines

Use this command to configure the flow timeout for UDP traffic. This parameter defines the amount of time the system will continue to wait for data from an UDP flow before considering the flow complete.

Use the **set** form of this command to set the flow timeout for UDP traffic.

Use the **delete** form of this command to to return the flow timeout for UDP traffic to the default value.

Use the **show** form of this command to view the configuration.

system flow-accounting netflow version <version>

Specifies the Netflow format that data will be exported in.

Syntax

set system flow-accounting netflow version *version*

delete system flow-accounting netflow version

show system flow-accounting netflow version

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        netflow {
            version version
        }
    }
}
```

Parameters

version	The Netflow version the exported data is formatted in. Supported values are 1, 5, and 9. The default value is 5.

Default

Netflow version 5 format is used.

Usage Guidelines

Use this command to set the formatting of the exported data to match a Netflow version.

Use the **set** form of this command to specify the Netflow version.

Use the **delete** form of this command to remove the configured version number and use the default value.

Use the **show** form of this command to display Netflow version configuration.

system flow-accounting sflow agent-address <addr>

Allows you to specify the IP address of the sFlow agent.

Syntax

set system flow-accounting sflow agent-address *addr*

delete system flow-accounting sflow agent-address

show system flow-accounting sflow agent-address

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        sflow {
            agent-address addr
        }
    }
}
```

Parameters

addr	The IP address of the sFlow agent to be included in sFlow packets sent to the collector. Supported values are **auto** (in which case, the system selects one of its own IP address) or an IPv4 address. The default value is **auto**.

Default

The system selects an IP address to send as the source for sFlow data.

Usage Guidelines

Use this command to configure an IP address to be sent to the sFlow collector indicating the source of the sFlow data—i.e., the local Vyatta system.

Use the **set** form of this command to set the agent address.

Use the **delete** form of this command to remove the agent address and use the default.

Use the **show** form of this command to view the configuration.

system flow-accounting sflow sampling-rate <rate>

Specifies the rate at which sFlow statistics are recorded.

Syntax

set system flow-accounting sflow sampling-rate *rate*

delete system flow-accounting sflow sampling-rate

show system flow-accounting sflow sampling-rate

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        sflow {
            sampling-rate rate
        }
    }
}
```

Parameters

rate	The rate at which packets are sampled (that is, if 1 in *n* packets are sampled, *n* is the rate).

Default

Every packet is sampled (that is, the sampling rate is 1).

Usage Guidelines

Use this command to configure the sFlow sampling rate for flow accounting. The system samples one in every *n* packets, where *n* is the value configured for the **sampling-rate** option.

The advantage of sampling every *n* packets, where *n* > 1, allows you to decrease the amount of processing resources required for flow accounting. The disadvantage of not sampling every packet is that the statistics produced are estimates of actual data flows.

Use the **set** form of this command to specfy the sampling rate.

Use the **delete** form of this command to sample all packets.

Use the **show** form of this command to display sampling rate configuration.

system flow-accounting sflow server <ipv4>

Specifies an sflow collector to export sFlow data to.

Syntax

set system flow-accounting sflow server *ipv4* [**port** *port*]

delete system flow-accounting sflow server *ipv4* [**port**]

show system flow-accounting sflow server *ipv4* [**port**]

Command Mode

Configuration mode.

Configuration Statement

```
system {
    flow-accounting {
        sflow {
            server ipv4 {
                port port
            }
        }
    }
}
```

Parameters

ipv4	Multi-node. The IP address of an sFlow collector to export the sFlow data to. You can export sFlow data to more than one sFlow collector by issuing this command multiple times.
port	The port on the sFlow collector to export the sFlow data to. The default value is 6343.

Default

None.

Usage Guidelines

Use this command to specify an sFlow collector to which to export sFlow data.

Use the **set** form of this command to specify an sFlow collector.

Use the **delete** form of this command to remove an sFlow collector configuration.

Use the **show** form of this command to display sFlow collector configuration.

system flow-accounting syslog-facility <facility>

Specifies the kinds of flow accounting messages to be logged.

Syntax

set system flow-accounting syslog-facility *facility*

delete system flow-accounting syslog-facility

show system flow-accounting syslog-facility

Command Mode

Configuration mode.

Configuration Statement

```
system {
   flow-accounting {
      syslog-facility facility
   }
}
```

Parameters

facility	The kinds of messages to be logged using **syslog**. Please see the Usage Guidelines in the **system syslog** command for supported facilities. The default value is **daemon**.

Default

System daemon messages are logged.

Usage Guidelines

Use this command to configure the kinds of flow accounting messages that will be logged.

Use the **set** form of this command to specify the kinds of flow accounting messages that will be logged.

Use the **delete** form of this command to remove the configuration.

Use the **show** form of this command to display configuration flow accounting logging configuration.

Glossary of Acronyms

ACL	access control list
ADSL	Asymmetric Digital Subscriber Line
AMI	Amazon Machine Image
API	Application Programming Interface
AS	autonomous system
ARP	Address Resolution Protocol
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BIOS	Basic Input Output System
BPDU	Bridge Protocol Data Unit
CA	certificate authority
CCMP	AES in counter mode with CBC-MAC
CHAP	Challenge Handshake Authentication Protocol
CLI	command-line interface
DDNS	dynamic DNS
DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6

DLCI	data-link connection identifier
DMI	desktop management interface
DMZ	demilitarized zone
DN	distinguished name
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
eBGP	external BGP
EBS	Amazon Elastic Block Storage
EC2	Amazon Elastic Compute Cloud
EGP	Exterior Gateway Protocol
ECMP	equal-cost multipath
ESP	Encapsulating Security Payload
FIB	Forwarding Information Base
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
HDLC	High-Level Data Link Control
I/O	Input/Ouput
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IGP	Interior Gateway Protocol
IPS	Intrusion Protection System
IKE	Internet Key Exchange
IP	Internet Protocol
IPOA	IP over ATM

IPsec	IP security
IPv4	IP Version 4
IPv6	IP Version 6
ISP	Internet Service Provider
KVM	Kernel-Based Virtual Machine
L2TP	Layer 2 Tunneling Protocol
LACP	Link Aggregation Control Protocol
LAN	local area network
LDAP	Lightweight Directory Access Protocol
LLDP	Link Layer Discovery Protocol
MAC	medium access control
MIB	Management Information Base
MLPPP	multilink PPP
MRRU	maximum received reconstructed unit
MTU	maximum transmission unit
NAT	Network Address Translation
ND	Neighbor Discovery
NIC	network interface card
NTP	Network Time Protocol
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
P2P	peer-to-peer
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PAT	Port Address Translation

PCI	peripheral component interconnect
PKI	Public Key Infrastructure
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM
PPPoE	PPP over Ethernet
PPTP	Point-to-Point Tunneling Protocol
PVC	permanent virtual circuit
QoS	quality of service
RADIUS	Remote Authentication Dial-In User Service
RHEL	Red Hat Enterprise Linux
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPng	RIP next generation
Rx	receive
S3	Amazon Simple Storage Service
SLAAC	Stateless Address Auto-Configuration
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SONET	Synchronous Optical Network
SSH	Secure Shell
SSID	Service Set Identifier
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol
ToS	Type of Service

Tx	transmit
UDP	User Datagram Protocol
vif	virtual interface
VLAN	virtual LAN
VPC	Amazon virtual private cloud
VPN	Virtual Private Network
VRRP	Virtual Router Redundancy Protocol
WAN	wide area network
WAP	wireless access point
WPA	Wired Protected Access