VYATTA, INC. | Vyatta System

Firewall

REFERENCE GUIDE

IPv4 Firewall
IPv6 Firewall
Zone-Based Firewall

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

COPYRIGHT

PROPRIETARY NOTICES

RELEASE DATE: October 2012

DOCUMENT REVISION. 6.5R1 v01

RELEASED WITH: 6.5R1

PART NO. A0-0231-10-0014

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

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:

Chapter	Description	Page
Chapter 1: Firewall Overview	This chapter provides an overview of firewall protection features on the Vyatta system.	1
Chapter 2: Configuration Examples	This chapter provides configuration examples and examples of how to display firewall information.	10
Chapter 3: Global Firewall Commands	This chapter describes Vyatta system firewall commands. These commands apply to both IPv4 and IPv6 firewalls.	56
Chapter 4: IPv4 Firewall Commands	This chapter describes commands for defining IPv4 firewall packet filters on the Vyatta system.	68
Chapter 5: IPv6 Firewall Commands	This chapter describes commands for defining IPv6 firewall packet filters on the Vyatta system.	120

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: Firewall Overview

This chapter provides an overview of firewall protection features on the Vyatta system.

This section presents the following topics:

Vyatta System Firewall Functionality

Firewall functionality analyzes and filters IP packets between network interfaces. The most common application of this is to protect traffic between an internal network and the Internet. It allows you to filter packets based on their characteristics and perform actions on packets that match the rule. Vyatta system firewall functionality provides the following:

- Packet filtering for traffic traversing the router, using the **in** and **out** keywords on an interface
- Packet filtering for traffic destined for the router itself, using the **local** keyword
- Definable criteria for packet-matching rules, including source IP address, destination IP address, source port, destination port, IP protocol, and ICMP type
- General detection on IP options such as source routing and broadcast packets
- Ability to set the firewall globally for stateful or stateless operation

The Vyatta firewall features both IPv4 and IPv6 stateful packet inspection to intercept and inspect network activity and allow or deny the attempt. Vyatta's advanced firewall capabilities include stateful failover, zone-based firewalling, time-based firewalling, and more.

Defining Firewall Instances

This section presents the following topics:

Firewalls filter packets on interfaces. There are two steps for using the firewall feature:

1. You define a firewall instance and save it under a name. A firewall instance is also called a firewall rule set, where a rule set is just a series of firewall rules. You define the firewall instance and configure the rules in its rule set in the **firewall** configuration node.
2. After defining the instance and specifying the rules in the rule set, you apply the instance to an interface or a zone. You do this by configuring the **interface** configuration node for the interface or zone.

Once the instance is applied to the interface or zone, the rules in the instance begin filtering packets on that location.

Firewall Rules

Firewall rules specify the match conditions for traffic and the action to be taken if the match conditions are satisfied. Traffic can be matched on a number of characteristics, including source IP address, destination IP address, source port, destination port, IP protocol, and ICMP type.

Rules are executed in sequence, according to the rule number. If the traffic matches the characteristics specified by the rule, the rule's action is executed; if not, the system "falls through" to the next rule.

The action can be one of the following:

- **Accept**. Traffic is allowed and forwarded.
- **Drop**. Traffic is silently discarded.
- **Reject**. Traffic is discarded with an ICMP "Port Unreachable" message.
- **Inspect**. Traffic is processed by the intrusion protection system (IPS).

Implicit Drop

All firewall rule sets on the Vyatta system have, by default, an implicit final action of **drop all**; that is, traffic not matching any rule in the rule set is silently discarded. This default action can be changed using the firewall name <name> default-action <action> command.

Exclusion Rules

Note that you should take care in using more than one "exclusion" rule (that is, a rule using the negation operation ("!") to exclude a rule from treatment). Rules are evaluated sequentially, and a sequence of exclusion rules could result in unexpected behavior.

Stateful Firewall and Connection Tracking

The Vyatta system CLI interacts with Netfilter's Connection Tracking System, which is a module providing connection tracking for various system functions, such as firewall, NAT, and WAN load balancing. On the firewall, connection tracking allows for stateful packet inspection.

Stateless firewalls filter packets in isolation, based on static source and destination information. In contrast, stateful firewalls track the state of network connections and traffic flows and allow or restrict traffic based on whether its connection state is known and authorized. For example, when an initiation flow is allowed in one

direction, the responder flow is automatically and implicitly allowed in the return direction. While typically slower under heavy load than stateless firewalls, stateful firewalls are better at blocking unauthorized communications.

By default, the Vyatta firewall is stateless. If you want the firewall to operate stateless in general, you can still configure state rules within a specific rule set. Alternatively, you can configure the firewall globally to operate statefully. In this case, you configure state policies for each of established, related, and invalid traffic, using the firewall state-policy command.

Global state policies configured apply to all IPv4 and IPv6 traffic destined for the router, originating from the router, or traversing the router. Also, once global state policies are configured, they override any state rules configured within the rule set.

Applying Firewall Instances to Interfaces

Once a firewall instance is defined it can be applied to an interface, where the instance acts as a packet filter. The firewall instance filters packets in one of the following ways, depending on what you specify when you apply the firewall instance:

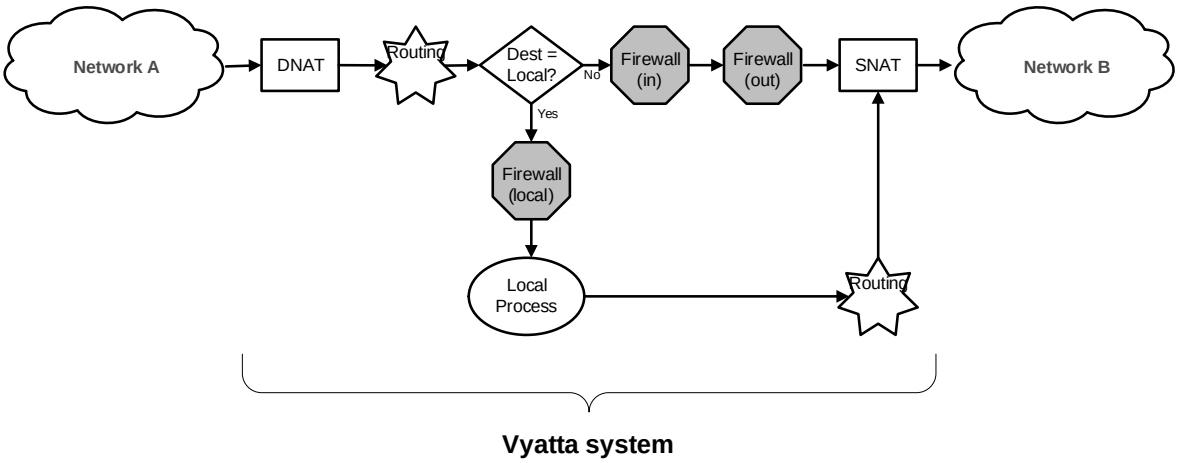
- **in.** If you apply the instance as **in**, the firewall will filter packets entering the interface and traversing the Vyatta system. You can apply one **in** packet filter.
- **out.** If you apply the instance as **out**, the firewall will filter packets leaving the interface. These can be packets traversing the Vyatta system or packets originated on the system. You can apply one **out** packet filter.
- **local.** If you apply the instance as **local**, the firewall will filter packets destined for the Vyatta system. One firewall instance can be applied as a **local** packet filter.

A total of three firewall instances can be applied to an interface: one instance as an **in** filter, one instance as an **out** filter, and one instance as a **local** filter.

Interaction Between Firewall, NAT, and Routing

One of the most important things to understand when working with firewall is the processing order of the various services that might be configured within the Vyatta system. If processing order is not considered, the results achieved may not be as intended. Figure 1-1shows how traffic flows through the firewall, NAT, and routing services within the Vyatta system.

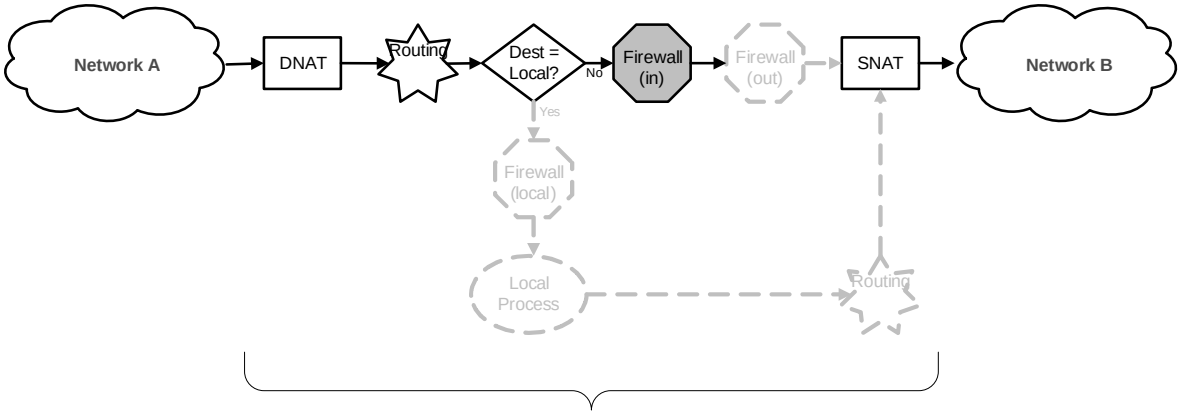
Figure 1-1 Traffic flow through firewall, NAT, and routing components



Scenario 1: Firewall instances applied to inbound traffic

The following diagram shows the traffic flow relationships between firewall, NAT, and routing, within the Vyatta system for traffic flowing through the system and firewall instances applied to **in**-bound traffic on an interface.

Figure 1-2 Inbound traffic flows through the Vyatta system

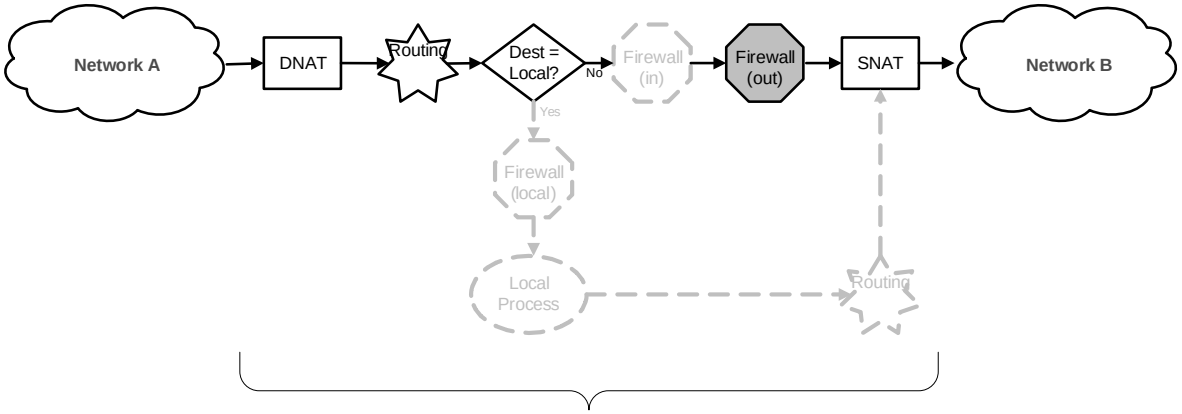


Notice that firewall instances are evaluated after DNAT and routing decisions, but prior to SNAT.

Scenario 2: Firewall instances applied to outbound traffic

The following diagram shows the traffic flow relationships between firewall, NAT, and routing, within the Vyatta system for traffic flowing through the system and firewall instances applied to **out**-bound traffic on an interface.

Figure 1-3 Outbound traffic flows through the Vyatta system

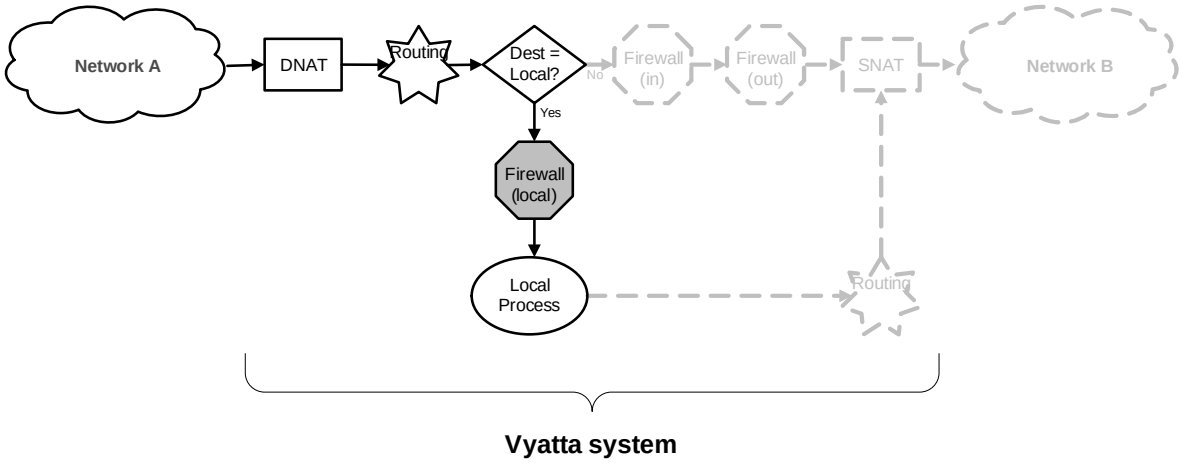


Notice that firewall instances are evaluated after DNAT and routing decisions, but prior to SNAT.

Scenario 3: Firewall instances applied to locally bound traffic

The following diagram shows the traffic flow relationships between firewall, NAT, and routing, within the Vyatta system for traffic flowing to the Vyatta system itself (firewall instances applied to **local** traffic on an interface).

Figure 1-4 Traffic flows destined for the Vyatta system

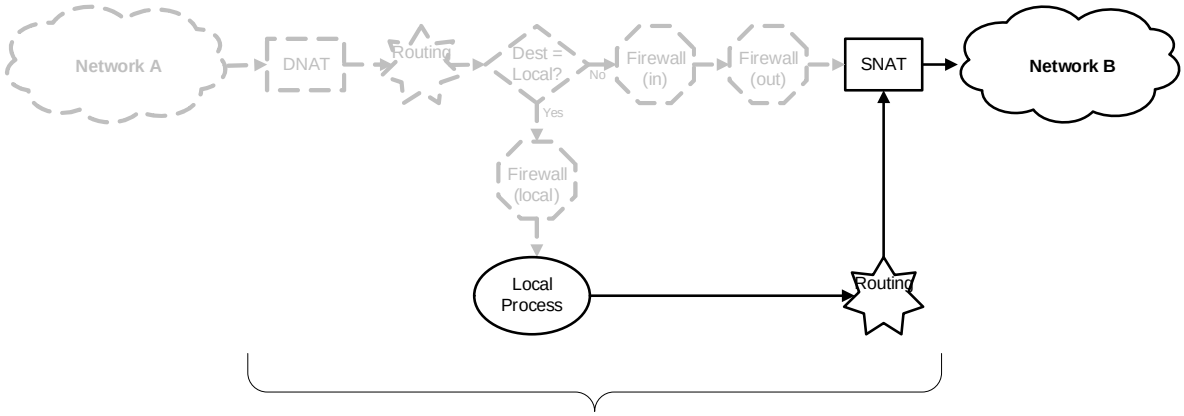


Notice that the firewall instance is evaluated after DNAT and routing. In this scenario, SNAT is not performed.

Scenario 4: Firewall instances applied to locally originated traffic

The following diagram shows the traffic flow relationships between firewall, NAT, and routing, within the Vyatta system for traffic flowing from the Vyatta system itself.

Figure 1-5 Traffic flows originating from the Vyatta system itself



Notice that no firewall instances are evaluated in this case. In this scenario, DNAT is not performed.

Zone-Based Firewall

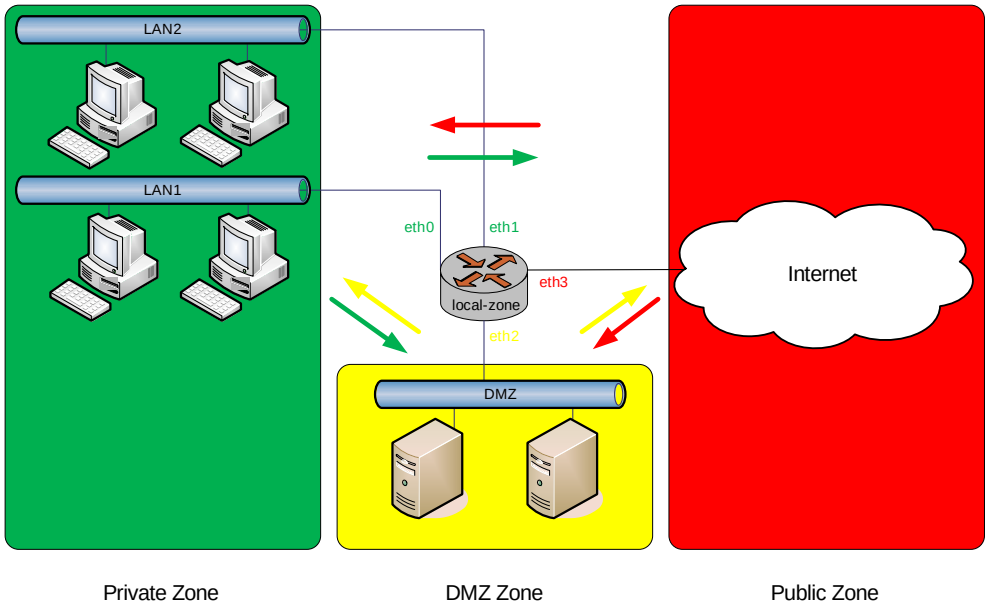
Ordinary firewall rule sets are applied on a per-interface basis to act as a packet filter for the interface. In zone-based firewall, interfaces are grouped into security "zones," where each interface in the zone has the same security level.

Packet-filtering policies are applied to traffic flowing between zones. Traffic flowing between interfaces lying in the same zone is not filtered and flows freely, as the interfaces share the same security level.

Figure 1-6 shows an example of a zone-based firewall implementation. In this example:

- There are three transit zones (that is, points where traffic transits the router): the Private Zone, the DMZ Zone, and the Public Zone.
- The eth3 interface lies in the Public Zone; eth0 and eth1 lie in the Private Zone; and eth2 lies in the DMZ zone.
- The arrows from one zone to another represent traffic filtering policies applied to traffic flowing between zones.
- Traffic flowing between LAN 1 and LAN 2 remains within a single security zone. Thus, traffic from LAN1 to LAN2, and vice-versa, flows unfiltered.

Figure 1-6 Zone-based firewall overview



In addition to the three transit zones in Figure 1-6, there is a fourth zone: the "Local Zone." The Local Zone is the router itself. By default, all traffic coming into the router and originating from the router is allowed.

You can, however, configure traffic filtering policies that allow traffic to the Local Zone from specific zones, and likewise from the Local Zone to only specific zones. As soon as you apply a filtering policy explicitly allowing traffic destined to Local Zone from another zone, traffic from all other zones to the Local Zone is dropped unless explicitly allowed by a filtering policy. Similarly, as soon as you apply a filtering policy to allow traffic originating from the Local Zone to another zone, traffic to all other zones is dropped unless explicitly allowed by a filtering policy.

Note the following additional points about zone-based firewalls:

- An interface can be associated with only one zone.
- An interface belonging to a zone cannot have a per-interface firewall rule set applied and vice versa.
- Traffic between interfaces not belonging to any zone flows unfiltered and per-interface firewall rule sets can be applied to those interfaces.
- By default, all traffic to a zone is dropped unless explicitly allowed by a filtering policy for a **from_zone**.
- Filtering policies are unidirectional: they are defined as a "zone pair" defining the zone from which traffic is sourced (the **from_zone**) and the zone to which traffic is destined (the **to_zone**). In Figure 1-6, these unidirectional policies can be seen as follows:
  - From Private to DMZ

- From Public to DMZ
- From Private to Public
- From DMZ to Public
- From Public to Private
- From DMZ to Private

IPv6 Firewall

The protection offered by a firewall is even more important to sites using IPv6 because IPv6 does not offer NAT functionality. Therefore, a firewall is the only way to protect an IPv6 network.

Note that IPv4 firewall rules and IPv6 firewall rules are completely independent. IPv4 packets are not inspected by rules in IPv6 rule sets, and IPv6 rules are not inspected by rules in IPv4 rule sets. and IPv6 packets are not inspected by rules in the other IP version's table; IPv6 packets are ONLY inspected by the rules in the IPv6 filter table, while IPv4 packets are ONLY inspected by the rules in the IPv4 filter table.

In general, IPv6 support for firewall parallels that for IPv4 firewall. Some IPv4-specific parameters do not apply to IPv6 firewalls, and vice versa, for example:

- The ICMP protocol has an IPv6-specific version: “ICMP for IPv6.” The IPv6 firewall has the **icmpv6** keyword available for the **protocol** filtering option, but the **icmp** keyword is not supported.
- The **fragment** parameter is not supported for IPv6 firewall, since fragmentation does not apply to IPv6.

Chapter 2: Configuration Examples

This chapter provides configuration examples and examples of how to display firewall information.

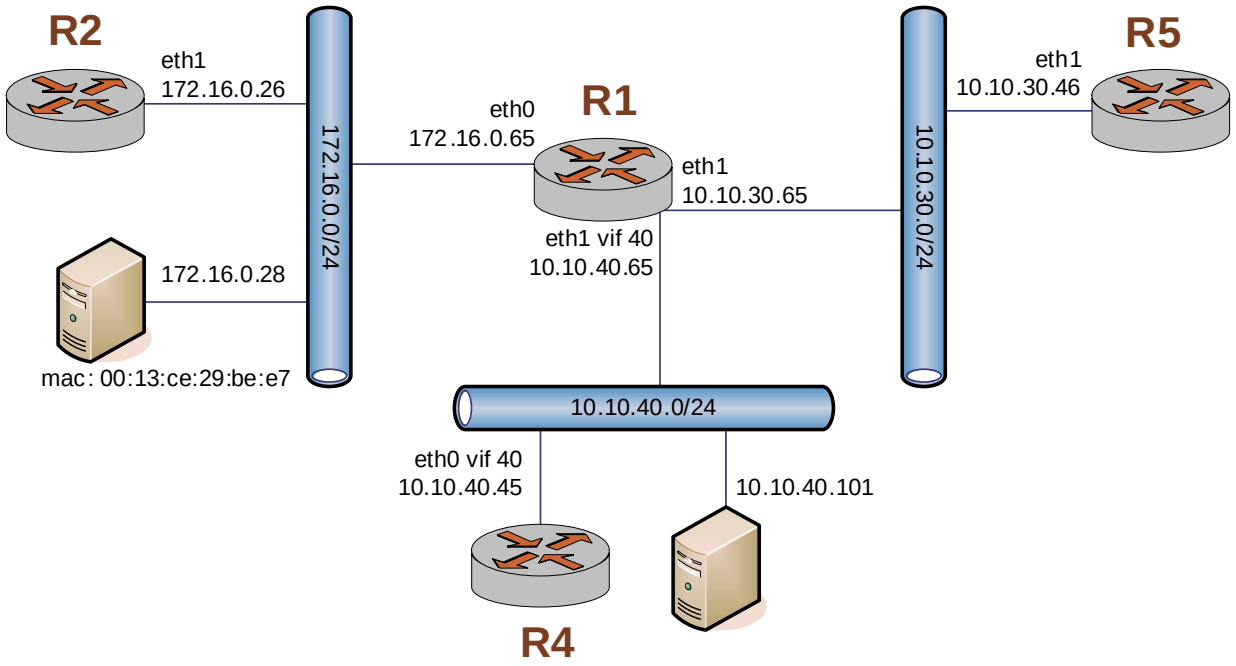
This chapter presents the following topics:

Packet-Filtering

This section presents the following topics:

This section describes a sample configuration for firewall. When you have finished, the firewall will be configured on router R1 as shown in Figure 2-1.

Figure 2-1 Firewall



This section includes the following examples:

- Example 2-1 Filtering on source IP
- Example 2-2 Filtering on source and destination IP
- Example 2-3 Filtering on source IP and destination protocol
- Example 2-4 Defining a network-to-network filter
- Example 2-5 Filtering on source MAC address
- Example 2-6 Excluding an address
- Example 2-7 Activate during specified time periods
- Example 2-8 Limit the rate of specific incoming packets
- Example 2-9 Accept packets with specific TCP flags set.
- Example 2-10 Accept ICMP packets with specific type names.
- Example 2-11 Reject traffic based on groups of addresses, networks, and ports.
- Example 2-12 Drop connection attempts from the same source over a specified threshold in a given period.
- Example 2-13 Creating a per–rule set state rule

Filtering on Source IP

Example 2-1 defines a firewall instance containing one rule, which filters on source IP address only. This rule will deny packets coming from router R2. It then applies the firewall instance to packets inbound on interface eth0.

To create an instance that filters on source IP, perform the following steps in configuration mode.

Example 2-1 Filtering on source IP

Step	Command
Create the configuration node for FWTEST-1 and its rule Rule 1. This rule rejects traffic matching the specified criteria.	`vyatta@R1# set firewall name FWTEST-1 rule 1 action reject`
This rule applies to traffic that has 176.16.0.26 as the source.	`vyatta@R1# set firewall name FWTEST-1 rule 1 source address 172.16.0.26`
Apply FWTEST-1 to inbound packets on eth0.	`vyatta@R1# set interfaces ethernet eth0 firewall in name FWTEST-1`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name FWTEST-1`
	`rule 1 {`
	`action reject`
	`source {`
	`address 172.16.0.26`
	`}`
	`}`
	`vyatta@R1# show interfaces ethernet eth0`
	`address 172.16.1.1/24`
	`firewall {`
	`in {`
	`name FWTEST-1`
	`}`
	`}`

Filtering on Source and Destination IP

Example 2-2 defines another firewall instance. It contains one rule, which filters on both source and destination IP address. This rule accepts packets leaving R5 through eth1 using 10.10.30.46, and destined for 10.10.40.101. It then applies the firewall instance to packets outbound from vif 1 on interface eth1.

To create an instance that filters on source and destination IP, perform the following steps in configuration mode.

Example 2-2 Filtering on source and destination IP

Step	Command
Create the configuration node for FWTEST-2 and its rule Rule 1. This rule accepts traffic matching the specified criteria.	`vyatta@R1# set firewall name FWTEST-2 rule 1 action accept`
This rule applies to traffic that has 10.10.30.46 as the source.	`vyatta@R1# set firewall name FWTEST-2 rule 1 source address 10.10.30.46`
This rule applies to traffic that has 10.10.40.101 as the destination.	`vyatta@R1# set firewall name FWTEST-2 rule 1 destination address 10.10.40.101`
Apply FWTEST-2 to outbound packets on eth1 vif 40.	`vyatta@R1# set interfaces ethernet eth1 vif 40 firewall out name FWTEST-2`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name FWTEST-2`
	`rule 1 {`
	`  action accept`
	`  destination {`
	`  address 10.10.40.101`
	`  }`
	`  source {`
	`  address 10.10.30.46`
	`  }`
	`}`
	`vyatta@R1# show interfaces ethernet eth1`
	`vif 40 {`
	`  firewall {`
	`  out {`
	`  name FWTEST-2`
	`  }`
	`  }`
	`}`

Filtering on Source IP and Destination Protocol

Example 2-3 defines a firewall rule that filters on source IP address and destination protocol. This rule allows TCP packets originating from address 10.10.30.46 (that is, R5), and destined for the Telnet port of R1. The instance is applied to local packets (that is, packets destined for this router, R1) through eth1.

To create a instance that filters on source IP and destination protocol, perform the following steps in configuration mode.

Example 2-3 Filtering on source IP and destination protocol

Step	Command
Create the configuration node for FWTEST-3 and its rule Rule 1. This rule accepts traffic matching the specified criteria.	`vyatta@R1# set firewall name FWTEST-3 rule 1 action accept`
This rule applies to traffic that has 10.10.30.46 as the source.	`vyatta@R1# set firewall name FWTEST-3 rule 1 source address 10.10.30.46`
This rule applies to TCP traffic.	`vyatta@R1# set firewall name FWTEST-3 rule 1 protocol tcp`
This rule applies to traffic that is destined for the Telnet service.	`vyatta@R1# set firewall name FWTEST-3 rule 1 destination port telnet`
Apply FWTEST-3 to packets bound for this router arriving on eth1.	`vyatta@R1# set interfaces ethernet eth1 firewall local name FWTEST-3`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name FWTEST-3`
	`rule 1 {`
	`action accept`
	`destination {`
	`port telnet`
	`}`
	`protocol tcp`
	`source {`
	`address 10.10.30.46`
	`}.`
	`}`
	`vyatta@R1 show interfaces ethernet eth1`
	`firewall {`
	`local {`
	`name FWTEST-3`
	`}`
	`}`
	`vif 40 {`
	`firewall {`
	`out {`
	`name FWTEST-2`
	`}`
	`}`
	`}`

Defining a Network-to-Network Filter

Example 2-4 creates a network-to-network packet filter, allowing packets originating from 10.10.40.0/24 and destined for 172.16.0.0/24. It then applies the firewall instance to packets inbound through vif 40 on interface eth1.

To create a network-to-network filter, perform the following steps in configuration mode.

Example 2-4 Defining a network-to-network filter

Step	Command
Create the configuration node for FWTEST-4 and its rule Rule 1. This rule accepts traffic matching the specified criteria.	`vyatta@R1# set firewall name FWTEST-4 rule 1 action accept`
This rule applies to traffic coming from the network 10.10.40.0/24.	`vyatta@R1# set firewall name FWTEST-4 rule 1 source address 10.10.40.0/24`
This rule applies to traffic destined for the network 172.16.0.0/24.	`vyatta@R1# set firewall name FWTEST-4 rule 1 destination address 172.16.0.0/24`
Apply FWTEST-4 to packets bound for this router arriving through vif 40 on eth1.	`vyatta@R1# set interfaces ethernet eth1 vif 40 firewall in name FWTEST-4`
Commit the configuration.	`vyatta@R1# commit`

Example 2-4 Defining a network-to-network filter

Show the configuration.

```
vyatta@R1# show firewall name FWTEST-4
 rule 1 {
    action accept
    destination {
        address 172.16.0.0/24
    }
    source {
        address 10.10.40.0/24
    }
 }
vyatta@R1 show interfaces ethernet eth1
 firewall {
    local {
        name FWTEST-3
    }
 }
 vif 40 {
    firewall {
        in {
            name FWTEST-4
        }
        out {
            name FWTEST-2
        }
    }
 }
```

Filtering on Source MAC Address

Example 2-5 defines a firewall instance containing one rule, which filters on source MAC address only. This rule will allow packets coming from a specific computer, identified by its MAC address rather than its IP address. The instance is applied to packets inbound on interface eth0.

To create a instance that filters on source MAC address, perform the following steps in configuration mode:

Example 2-5 Filtering on source MAC address

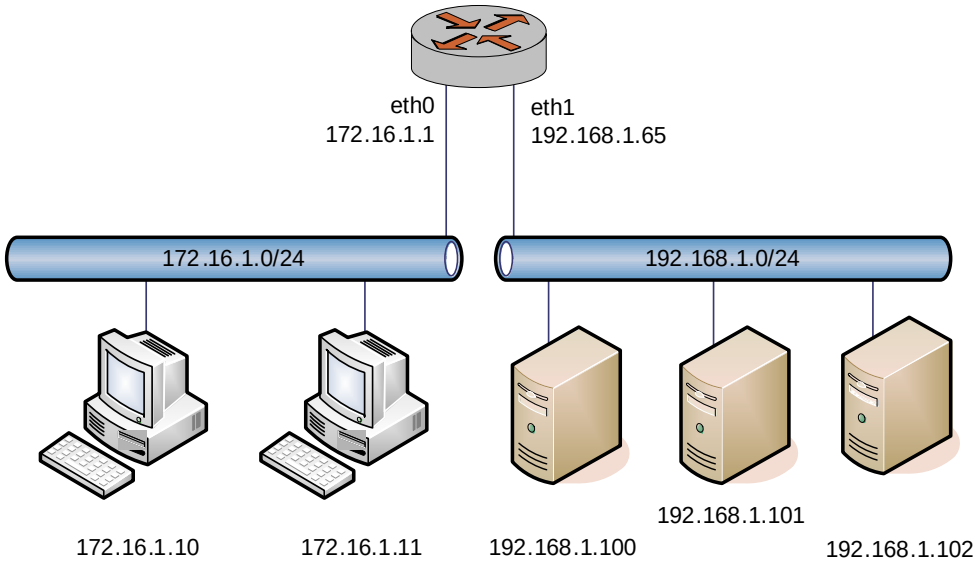
Step	Command
Create the configuration node for FWTEST-5 and its rule Rule 1. This rule accepts traffic matching the specified criteria.	`vyatta@R1# set firewall name FWTEST-5 rule 1 action accept`
This rule applies to traffic that has 00:13:ce:29:be:e7 as the source MAC address.	`vyatta@R1# set firewall name FWTEST-5 rule 1 source mac-address 00:13:ce:29:be:e7`
Apply FWTEST-5 to inbound packets on eth0.	`vyatta@R1# set interfaces ethernet eth0 firewall in name FWTEST-5`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name FWTEST-5`
	`rule 1 {`
	`action accept`
	`source {`
	`mac-address 00:13:ce:29:be:e7`
	`}`
	`}`
	`vyatta@R1 show interfaces ethernet eth0`
	`address 172.16.1.1/24`
	`firewall {`
	`in {`
	`name FWTEST-5`
	`}`
	`}`

Excluding an Address

The firewall rule shown in Example 2-6 allows all traffic from the 172.16.1.0/24 network except to server 192.168.1.100.

Figure 2-2 Excluding an address



To create a instance that excludes an address, perform the following steps in configuration mode.

Example 2-6 Excluding an address

Step	Command
Create the configuration node for FWTEST-5 and its rule 10. Give a description for the rule.	`vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 description "Allow all traffic from LAN except to server 192.168.1.100"`
All traffic that matches the rule will be accepted.	`vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 action accept`
Any traffic from network 172.16.1.0/24 matches the rule.	`vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 source address 172.16.1.0/24`
Traffic destined anywhere EXCEPT 192.168.1.100 matches the rule. That traffic does not match the rule, and invokes the implicit “reject all” rule.	`vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 destination address !192.168.1.100`
Apply the instance NEGATED-EXAMPLE to inbound packets on eth0.	`vyatta@R1# set interfaces ethernet eth0 firewall in name NEGATED-EXAMPLE`
Commit the configuration.	`vyatta@R1# commit`

Example 2-6 Excluding an address

Show the configuration.	vyatta@R1# show firewall
	name NEGATED-EXAMPLE {
	rule 10 {
	action accept
	description "Allow all traffic from LAN except to server 192.168.1.100"
	destination {
	address !192.168.1.100
	}
	source {
	address 172.16.1.0/24
	}
	}
	}
	
	vyatta@R1# show interfaces ethernet eth0
	address 172.16.1.1/24
	firewall {
	in {
	name NEGATED-EXAMPLE
	}
	}
	hw-id 00:0c:29:99:d7:74

Activating during Specific Time Periods

The Vyatta system supportes time-based firewall rules, which limit the operation of a rule to specific periods of time.

The firewall rule shown in Example 2-7 limits the rule configured in Example 2-6 to being active only on weekdays from 9:00 AM until 5:00 PM. To add this limitation to the rule, perform the following steps in configuration mode.

Example 2-7 Activate during specified time periods

Step	Command
Set a start time of 9:00am.	vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 time starttime 09:00:00
Set a stop time of 5:00pm.	vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 time stoptime 17:00:00
Set the days of the week.	vyatta@R1# set firewall name NEGATED-EXAMPLE rule 10 time weekdays Mon,Tue,Wed,Thu,Fri
Commit the configuration.	vyatta@R1# commit

Example 2-7 Activate during specified time periods

Show the configuration.

```
vyatta@R1# show firewall
name NEGATED-EXAMPLE {
    rule 10 {
        action accept
        description "Allow all traffic from LAN except to
server 192.168.1.100"
        destination {
            address !192.168.1.100
        }
        source {
            address 172.16.1.0/24
        }
        time {
            starttime 09:00:00
            stoptime 17:00:00
            weekdays Mon,Tue,Wed,Thu,Fri
        }
    }
}

vyatta@R1# show interfaces ethernet eth0
    address 172.16.1.1/24
    firewall {
        in {
            name NEGATED-EXAMPLE
        }
    }
    hw-id 00:0c:29:99:d7:74
```

Limiting Traffic Rates

The Token Bucket Filter (TBF) queuing mechanism can be activated by a firewall rule to limit the rate of incoming packets. Packets are limited to an administatively set rate, but may have short bursts in excess of this rate. Two rules are required to achieve this; one to accept traffic within the limit, and one to drop traffic in excess of the limit.

For example, to create a rule that accepts a limited rate of two ICMP Echo Request packets (pings) per second, but provides for short bursts without dropping packets, and a rule that will drop packets that do not get matched by the first rule, perform the following steps in configuration mode.

Example 2-8 Limit the rate of specific incoming packets

Step	Command
Set the protocol to match to ICMP.	`vyatta@R1# set firewall name RATE-LIMIT rule 20 protocol icmp`
Set ICMP type of 8 (echo-request).	`vyatta@R1# set firewall name RATE-LIMIT rule 20 icmp type 8`
Set ICMP code of 0 for type 8	`vyatta@R1# set firewall name RATE-LIMIT rule 20 icmp code 0`
Set the desired rate of 2 packets per second.	`vyatta@R1# set firewall name RATE-LIMIT rule 20 limit rate 2/second`
Set the burst size of 5 packets.	`vyatta@R1# set firewall name RATE-LIMIT rule 20 limit burst 5`
Set the action to accept.	`vyatta@R1# set firewall name RATE-LIMIT rule 20 action accept`
Set the description.	`vyatta@R1# set firewall name RATE-LIMIT rule 20 description “Rate-limit incoming icmp echo-request packets to 2/second allowing short bursts of 5 packets”`
Set the protocol to match to ICMP.	`vyatta@R1# set firewall name RATE-LIMIT rule 30 protocol icmp`
Set ICMP type of 8 (echo-request).	`vyatta@R1# set firewall name RATE-LIMIT rule 30 icmp type 8`
Set ICMP code of 0 for type 8	`vyatta@R1# set firewall name RATE-LIMIT rule 30 icmp code 0`
Set the action to drop.	`vyatta@R1# set firewall name RATE-LIMIT rule 30 action drop`
Set the description.	`vyatta@R1# set firewall name RATE-LIMIT rule 30 description “Drop remaining echo requests in excess of the rate in rule 20”`
Commit the configuration.	`vyatta@R1# commit`

Example 2-8 Limit the rate of specific incoming packets

Show the configuration.	`vyatta@R1# show firewall name RATE-LIMIT`
	`rule 20 {`
	`  action accept`
	`  description "Rate-limit incoming icmp echo-request packets to 2/second allowing short bursts of 5 packets"`
	`  icmp {`
	`  code 0`
	`  type 8`
	`  }`
	`  limit {`
	`  burst 5`
	`  rate 2/second`
	`  }`
	`  protocol icmp`
	`}`
	`rule 30 {`
	`  action drop`
	`  description "Drop remaining echo requests in excess of the rate in rule 20"`
	`  icmp {`
	`  code 0`
	`  type 8`
	`  }`
	`  protocol icmp`
	`}`
	`vyatta@R1#`

Matching TCP Flags

The Vyatta system supports filtering on the TCP flags within TCP packets. For example, to create a rule to accept packets with the SYN flag set, and the ACK, FIN, and RST flags unset, perform the following steps in configuration mode.

Example 2-9 Accept packets with specific TCP flags set.

Step	Command
Set the protocol to match to tcp.	`vyatta@R1# set firewall name TCP-FLAGS rule 30 protocol tcp`
Set the TCP flags to match.	`vyatta@R1# set firewall name TCP-FLAGS rule 30 tcp flags SYN,!ACK,!FIN,!RST`
Set the action to accept.	`vyatta@R1# set firewall name TCP-FLAGS rule 30 action accept`
Commit the configuration.	`vyatta@R1# commit`

Example 2-9 Accept packets with specific TCP flags set.

Show the configuration.	`vyatta@R1# show firewall name TCP-FLAGS`
	`rule 30 {`
	`  action accept`
	`  protocol tcp`
	`  tcp {`
	`  flags SYN,!ACK,!FIN,!RST`
	`  }`
	`}`
	`vyatta@R1#`

Matching ICMP Type Names

Packets can be filtered for ICMP type names. For example, to create a rule that allows only ICMP Echo Request packets through, perform the following steps in configuration mode.

Example 2-10 Accept ICMP packets with specific type names.

Step	Command
Set the protocol to match to icmp.	`vyatta@R1# set firewall name ICMP-NAME rule 40 protocol icmp`
Set the ICMP packet type to match.	`vyatta@R1# set firewall name ICMP-NAME rule 40 icmp type-name echo-request`
Set the action to accept.	`vyatta@R1# set firewall name ICMP-NAME rule 40 action accept`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name ICMP-NAME`
	`rule 40 {`
	`  action accept`
	`  protocol icmp`
	`  icmp {`
	`  type-name echo-request`
	`  }`
	`}`
	`vyatta@R1#`

Matching Groups

Groups of addresses, ports, and networks can be defined for similar filtering. For example, to create a rule that rejects traffic to a group of addresses and ports and from a group of networks, perform the following steps in configuration mode.

Example 2-11 Reject traffic based on groups of addresses, networks, and ports.

Step	Command
Add a range of addresses to an address group.	`vyatta@R1# set firewall group address-group SERVERS address 1.1.1.1-1.1.1.5`
Add another address to an address group.	`vyatta@R1# set firewall group address-group SERVERS address 1.1.1.7`
Add a network to a network group.	`vyatta@R1# set firewall group network-group NETWORKS network 10.0.10.0/24`
Add a port to a port group.	`vyatta@R1# set firewall group port-group PORTS port 22`
Add a port name to a port group.	`vyatta@R1# set firewall group port-group PORTS port ftp`
Add a range of ports to a port group.	`vyatta@R1# set firewall group port-group PORTS port 1000-2000`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall group`
	`group {`
	`  address-group SERVERS {`
	`  address 1.1.1.1-1.1.1.5`
	`  address 1.1.1.7`
	`  }`
	`  network-group NETWORKS {`
	`  network 10.0.10.0/24`
	`  }`
	`  port-group PORTS {`
	`  port 22`
	`  port ftp`
	`  port 1000-2000`
	`  }`
	`}`
	`vyatta@R1#`

Specify a reject action within a firewall instance.	`vyatta@R1# set firewall name REJECT-GROUPS rule 10 action reject`
Specify an address group to match as a destination.	`vyatta@R1# set firewall name REJECT-GROUPS rule 10 destination group address-group SERVERS`
Specify an port group to match as a destination.	`vyatta@R1# set firewall name REJECT-GROUPS rule 10 destination group port-group PORTS`
Specify an network group to match as a source.	`vyatta@R1# set firewall name REJECT-GROUPS rule 10 source group network-group NETWORKS`

Example 2-11 Reject traffic based on groups of addresses, networks, and ports.

Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name REJECT-GROUPS`
	`rule 10{`
	`  action reject`
	`  destination {`
	`  group {`
	`  address-group SERVERS`
	`  port-group PORTS`
	`  }`
	`  }`
	`  source {`
	`  group {`
	`  network-group NETWORKS`
	`  }`
	`  }`
	`}`
	`vyatta@R1#`

Matching Recently-Seen Sources

The **recent** command can be used to help prevent "brute force" attacks where an external device opens a continuous flow of connections (for example, to the SSH port) in an attempt to break into the system. In these cases, the external source address may be unknown; however, this command enables matching based on the external host's behavior without initially knowing its IP address.

For example, to create a rule that limits incoming SSH connection attempts from the same host to three within 30 seconds, perform the following steps in configuration mode.

Example 2-12 Drop connection attempts from the same source over a specified threshold in a given period.

Step	Command
Match TCP packets.	`vyatta@R1# set firewall name STOP-BRUTE rule 10 protocol tcp`
Match a destination port of 22 (i.e. ssh).	`vyatta@R1# set firewall name STOP-BRUTE rule 10 destination port 22`
Match connection attempts.	`vyatta@R1# set firewall name STOP-BRUTE rule 10 state new enable`
Match the same source address 3 times in ...	`vyatta@R1# set firewall name STOP-BRUTE rule 10 recent count 3`

Example 2-12 Drop connection attempts from the same source over a specified threshold in a given period.

... 30 seconds.	`vyatta@R1# set firewall name STOP-BRUTE rule 10 recent time 30`
Drop packets that match these criteria.	`vyatta@R1# set firewall name STOP-BRUTE rule 10 action drop`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show firewall name STOP-BRUTE`
	`rule 10{`
	`  action drop`
	`  destination {`
	`  port 22`
	`  }`
	`  protocol tcp`
	`  recent {`
	`  count 3`
	`  time 30`
	`  }`
	`  state {`
	`  new enable`
	`  }`
	`}`
	`vyatta@R1#`

Stateful Behavior

This section presents the following topics:

- Configuring Stateful Behavior Per Rule Set
- Configuring Global State Policies

Stateless firewalls filter packets in isolation, based on static source and destination information. In contrast, stateful firewalls track the state of network connections and traffic flows and allow or restrict traffic based on whether its connection state is known and authorized. For example, when an initiation flow is allowed in one direction, the responder flow is automatically and implicitly allowed in the return direction.

For stateful behavior, you configure the action to take for established, related, and invalid traffic:

- Established traffic is return traffic belonging to a connection that has already had packets travel in both directions; for example, a reply packet, or an outgoing packet on a connection that has been replied to.

- Related traffic is traffic related to an existing connection; for example, an FTP data transfer connection or an ICMP error.
- Invalid traffic is packets that cannot be identified for some reason; for example, ICMP errors that do not correspond to any known connection, or packets not identified because the system has run out of resources.

By default, the Vyatta firewall is stateless. If you want the firewall to operate statefully, you have two choices:

- You can leave the firewall operating statelessly in general, and specify statefule behavior per rule set by configuring state rules within the rule set. This configuration is described in "Configuring Stateful Behavior Per Rule Set" on page 28.
- You can enable stateful behavior global, by configuring global state policies. This configuration is described in "Configuring Global State Policies" on page 29.

Configuring Stateful Behavior Per Rule Set

Even if you want the firewall to operate statelessly in general, you can still configure state rules within a specific rule set. Example 2-13 configures the first two rules in the firewall rule set TEST1.

- Rule 1 accepts established flows and flows related to existing connections, for all protocols.
- Rule 2 drops invalid traffic and logs the drop action.

To configure per–rule set state rules, perform the following steps in configuration mode.

Example 2-13 Creating a per–rule set state rule

Step	Command
Create the configuration node for the TEST1 rule set and give a description for the rule set.	`vyatta@R1# set firewall name TEST1 description "Filter traffic statefully"`

Create a state rule allowing only established and related traffic. This means that only traffic initiated in on the system or traffic related to established connections (such as FTP data connections or ICMP messages associated with a flow) are allowed.	`vyatta@R1# set firewall name TEST1 rule 1 action accept`
	`vyatta@R1# set firewall name TEST1 rule 1 state established enable`
	`vyatta@R1# set firewall name TEST1 rule 1 state related enable`

Example 2-13 Creating a per–rule set state rule

Create a state rule dropping invalid traffic.	`vyatta@R1# set firewall name TEST1 rule 2 action drop`
	`vyatta@R1# set firewall name TEST1 rule 2 state invalid enable`
	`vyatta@R1# set firewall name TEST1 rule 2 log enable`

Commit the configuration.	`vyatta@R1# commit`

Show the firewall configuration.	`vyatta@R1# show firewall name TEST1`
	`description "Filter traffic statefully"`
	`rule 1 {`
	`  action accept`
	`  state {`
	`  established enable`
	`  related enable`
	`  }`
	`}`
	`rule 2 {`
	`  action drop`
	`  log enable`
	`  state {`
	`  invalid enable`
	`  }`
	`}`

Configuring Global State Policies

You can change the behavior to be globally stateful by setting a global state policy, using the firewall state-policy command. Within the state policy, you can configure the behavior for established, related, and invalid traffic. When state policies are defined, state rules for return traffic of that type need not be explicitly mentioned within the rule sets.

The global state policy configured applies to all IPv4 and IPv6 traffic destined for the router, originating from the router, or traversing the router. Note that once the firewall is configured to be globally stateful, this setting overrides any state rules configured within the rule set.

Example 2-14 configures the firewall globally to allow all return traffic. In addition, the firewall will allow any traffic (such as FTP data) related to allowed traffic in the original direction. The firewall will drop invalid traffic. This behavior is the same as that configured in the TEST1 rule set in Example 2-13, except that it will be applied globally instead of being restricted to the one rule set.

To configure this global stateful behavior, perform the following steps in configuration mode.

Example 2-14 Setting a global state policy

Step	Command

Allow return traffic and traffic related to existing connections. Drop invalid traffic and log the drop.	`vyatta@R1# set firewall state-policy established action accept`
	`vyatta@R1# set firewall state-policy related action accept`
	`vyatta@R1# set firewall state-policy invalid action drop`
	`vyatta@R1# set firewall state-policy invalid log enable`

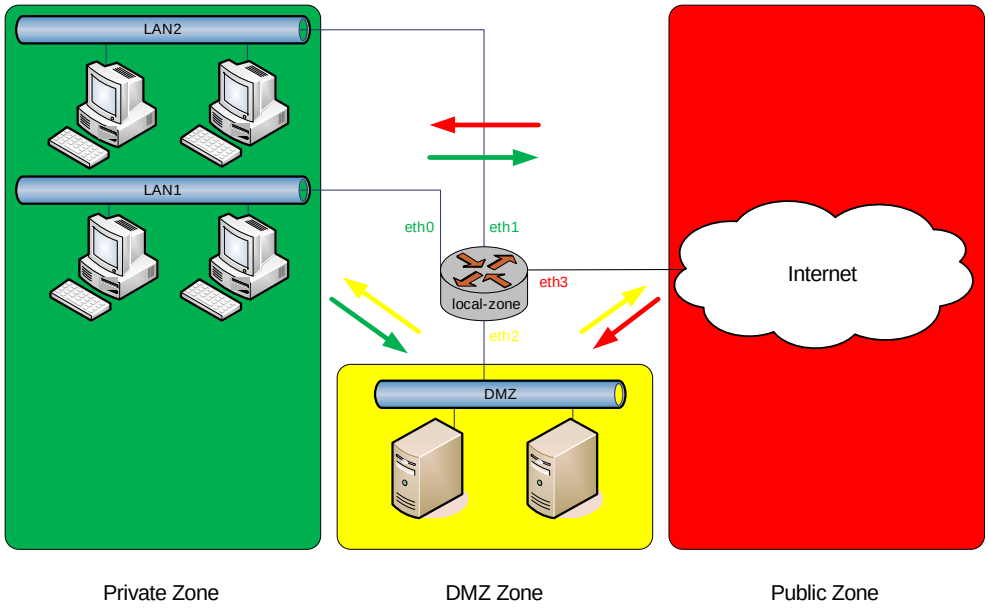
Commit the configuration.	`vyatta@R1# commit`

Show the state policy configuration.	`vyatta@R1# show firewall state-policy`
	`established {`
	`  action accept`
	`}`
	`related {`
	`  action accept`
	`}`
	`invalid {`
	`  action drop`
	`  log enable`
	`}`
	`[edit]`

Zone-Based Firewall

The Vyatta system also supports a zone-based model. Figure 2-3 shows a Zone-based configuration with three user-defined zones. The examples that follow show the configuration for this diagram.

Figure 2-3 Zone-based firewall configuration



This section presents the following topics:

- Filtering Traffic Between the Transit Zones
- Filtering Traffic To and From the Local Zone
- Considerations for Remote Access VPN
- Using Per-Interface Rule Sets with Zone-Based Firewall

Filtering Traffic Between the Transit Zones

The first step in setting up zone-based traffic filtering is to create zone policies, as shown in Example 2-15. To create the zone policies, perform the following steps in configuration mode.

Example 2-15 Creating the zone policies

Step	Command
Create the configuration node for the DMZ zone and give a description for the zone.	`vyatta@R1# set zone-policy zone dmz description “DMZ ZONE”`
Add the interface contained in the DMZ zone.	`vyatta@R1# set zone-policy zone dmz interface eth2`
Create the configuration node for the Private zone and give a description for the zone.	`vyatta@R1# set zone-policy zone private description “PRIVATE ZONE”`

Example 2-15 Creating the zone policies

Add one of the interfaces contained in the Private zone.	`vyatta@R1# set zone-policy zone private interface eth0`
Add the other interface contained in the Private zone.	`vyatta@R1# set zone-policy zone private interface eth1`
Create the configuration node for the Public zone and give a description for the zone.	`vyatta@R1# set zone-policy zone public description “PUBLIC ZONE”`
Add the interface contained in the Public zone.	`vyatta@R1# set zone-policy zone public interface eth3`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show zone-policy`
	`zone dmz {`
	`  description "DMZ ZONE"`
	`  interface eth2`
	`}`
	`zone private {`
	`  description "PRIVATE ZONE"`
	`  interface eth0`
	`  interface eth1`
	`}`
	`zone public {`
	`  description "PUBLIC ZONE"`
	`  interface eth3`
	`}`

At this point, while traffic can flow freely within a zone, no traffic flows between zones. All traffic flowing from one zone to another will be dropped. For example, because eth0 and eth1 lie in the same zone (Private), traffic between these interfaces flows freely. However, traffic from eth1 to eth2 (which lies in the DMZ zone) is dropped.

The next step, shown in Example 2-16, is to create firewall rule sets to allow traffic between zones. The first rule set allows all traffic to the Public zone. To configure this rule set, perform the following steps in configuration mode.

Example 2-16 Creating the rule set for traffic to the Public Zone

Step	**Command**
Create the configuration node for the to_public rule set and give a description for the rule set.	`vyatta@R1# set firewall name to_public description "allow all traffic to PUBLIC zone"`

Example 2-16 Creating the rule set for traffic to the Public Zone

Create a rule to accept all traffic sent to the Public Zone.	`vyatta@R1# set firewall name to_public rule 1 action accept`
Commit the configuration.	`vyatta@R1# commit`

Show the firewall configuration.	`vyatta@R1# show firewall name to_public`
	` description "allow all traffic to PUBLIC zone"`
	` rule 1 {`
	`  action accept`
	` }`

The next step, shown in Example 2-17, creates two rule sets: one from the Private zone to the DMZ zone and one from the Public zone to the DMZ zone.

- The rule set from Public to DMZ accepts all traffic for HTTP, HTTPS, FTP, SSH, and Telnet, as well as all ICMP traffic.
- The rule set from Private to DMZ accepts HTTP, HTTPS and ICMP traffic only.

To configure these rule sets, perform the following steps in configuration mode.

Example 2-17 Creating the rule set for traffic to the DMZ zone

Step	Command
Create the configuration node for the private_to_dmz rule set and give a description for the rule set.	`vyatta@R1# set firewall name private_to_dmz description "filter traffic from PRIVATE zone to DMZ zone"`

Create a rule to allow traffic sent from the Private zone to HTTP, HTTPS, FTP, SSH, and Telnet ports in the DMZ zone.	`vyatta@R1# set firewall name private_to_dmz rule 1 action accept`
	`vyatta@R1# set firewall name private_to_dmz rule 1 destination port http,https,ftp,ssh,telnet`
	`vyatta@R1# set firewall name private_to_dmz rule 1 protocol tcp`

Create a rule to allow all ICMP traffic sent from the Private zone to the DMZ zone.	`vyatta@R1# set firewall name private_to_dmz rule 2 action accept`
	`vyatta@R1# set firewall name private_to_dmz rule 2 icmp type-name any`
	`vyatta@R1# set firewall name private_to_dmz rule 2 protocol icmp`

Commit the configuration.	`vyatta@R1# commit`

Example 2-17 Creating the rule set for traffic to the DMZ zone

Show the firewall configuration.	vyatta@R1# show firewall name private_to_dmz
	description "filter traffic from PRIVATE zone to DMZ zone"
	rule 1 {
	action accept
	destination {
	port http,https,ftp,ssh,telnet
	}
	protocol tcp
	}
	rule 2 {
	action accept
	icmp {
	type-name any
	}
	protocol icmp
	}

Create the configuration node for the public_to_dmz rule set and give a description for the rule set.	vyatta@R1# set firewall name public_to_dmz description "filter traffic from PUBLIC zone to DMZ zone"
Create a rule to allow traffic sent from the Public zone only to HTTP and HTTPS ports in the DMZ zone.	vyatta@R1# set firewall name public_to_dmz rule 1 action accept vyatta@R1# set firewall name public_to_dmz rule 1 destination port http,https vyatta@R1# set firewall name public_to_dmz rule 1 protocol tcp
Create a rule to allow all ICMP traffic sent from the Public zone to the DMZ zone.	vyatta@R1# set firewall name public_to_dmz rule 2 action accept vyatta@R1# set firewall name public_to_dmz rule 2 icmp type-name any vyatta@R1# set firewall name public_to_dmz rule 2 protocol icmp
Commit the configuration.	vyatta@R1# commit

Example 2-17 Creating the rule set for traffic to the DMZ zone

Show the firewall configuration.	vyatta@R1# show firewall name public_to_dmz
	description "filter traffic from PUBLIC zone to DMZ zone"
	rule 1 {
	action accept
	destination {
	port http,https
	}
	protocol tcp
	}
	rule 2 {
	action accept
	icmp {
	type-name any
	}
	protocol icmp
	}

The next step, shown in Example 2-18, creates a rule set for traffic to the Private zone.

Note that this rule set includes state rules specifically allowing traffic from existing and related connections. This rule is required because, in this scenario

- The rule set from Public to DMZ accepts all traffic for HTTP, HTTPS, FTP, SSH, and Telnet, as well as all ICMP traffic.
- The rule set from Private to DMZ accepts HTTP, HTTPS and ICMP traffic only.

To configure this rule set, perform the following steps in configuration mode.

Example 2-18 Creating the rule set for traffic to the Private zone

Step	Command
Create the configuration node for the to_private rule set and give a description for the rule set.	vyatta@R1# set firewall name to_private description "filter traffic to PRIVATE zone"
Create a rule to allow only established and related traffic to the Private zone. This means that only traffic initiated in the Private zone or traffic related to established connections (such as FTP data connections or ICMP messages associated with a flow) are allowed.	vyatta@R1# set firewall name to_private rule 1 action accept vyatta@R1# set firewall name to_private rule 1 state established enable vyatta@R1# set firewall name to_private rule 1 state related enable vyatta@R1# set firewall name to_private rule 1 protocol all

Example 2-18 Creating the rule set for traffic to the Private zone

Commit the configuration.	`vyatta@R1# commit`

Show the firewall configuration.	`vyatta@R1# show firewall name to_private`
	` description "filter traffic to PRIVATE zone"`
	` rule 1 {`
	`  action accept`
	`  protocol all`
	`  state {`
	`  established enable`
	`  related enable`
	`  }`
	` }`

Example 2-19 applies the rule set to the DMZ zone.

Example 2-19 Applying a rule sets to the DMZ zone.

Step	**Command**
Apply the private_to_dmz rule set to traffic from the Private Zone to the DMZ Zone.	`vyatta@R1# set zone-policy zone dmz from private firewall name private_to_dmz`
Apply the public_to_dmz rule set to traffic from the Public Zone to the DMZZone.	`vyatta@R1# set zone-policy zone dmz from public firewall name public_to_dmz`
Commit the configuration.	`vyatta@R1# commit`

Show the DMZ Zone policy configuration.	`vyatta@R1# show zone-policy zone dmz`
	` description "DMZ ZONE"`
	` from private {`
	`  firewall {`
	`  name private_to_dmz`
	`  }`
	` }`
	` from public {`
	`  firewall {`
	`  name public_to_dmz`
	`  }`
	` }`
	` interface eth2`

Example 2-20 applies the rule set to the Private zone.

Example 2-20 Applying a rule set to the Private zone.

Step	Command
Apply the to_private rule set to traffic from the DMZ Zone to the Private Zone.	`vyatta@R1# set zone-policy zone private from dmz firewall name to_private`
Apply the to_private rule set to traffic from the Public Zone to the Private Zone.	`vyatta@R1# set zone-policy zone private from public firewall name to_private`
Commit the configuration.	`vyatta@R1# commit`

Show the Private Zone policy configuration.	`vyatta@R1# show zone-policy zone private`
	`description "PRIVATE ZONE"`
	`from dmz {`
	`  firewall {`
	`  name to_private`
	`  }`
	`}`
	`from public {`
	`  firewall {`
	`  name to_private`
	`  }`
	`}`
	`interface eth0`
	`interface eth1`

Example 2-21 applies the rule set to the Public zone.

Example 2-21 Applying a rule set to the Public zone.

Step	Command
Apply the to_public rule set to traffic from the DMZ Zone to the Public Zone.	`vyatta@R1# set zone-policy zone public from dmz firewall name to_public`
Apply the to_pubic rule set to traffic from the Private Zone to the Public Zone.	`vyatta@R1# set zone-policy zone public from private firewall name to_public`
Commit the configuration.	`vyatta@R1# commit`

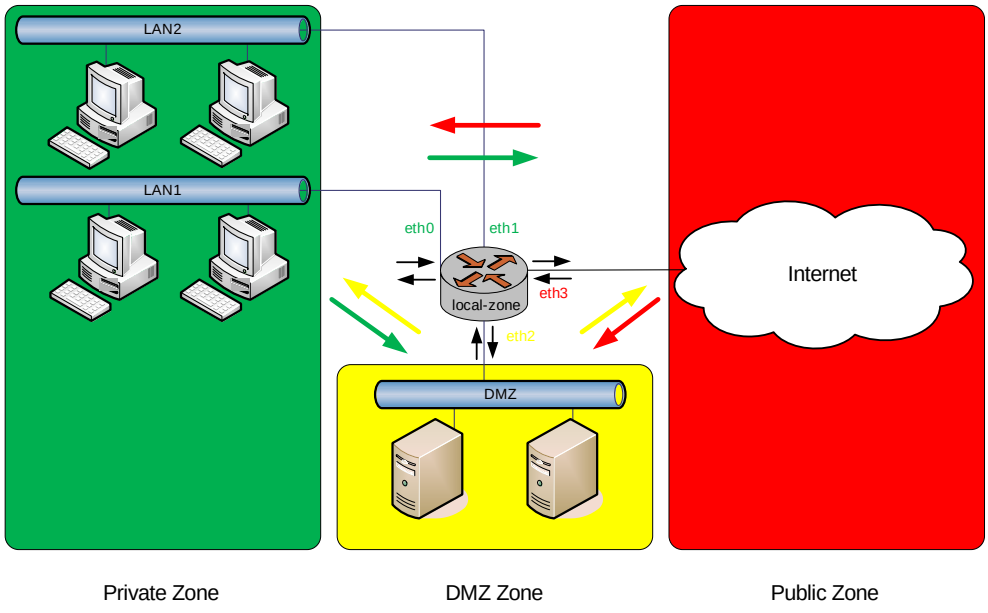
Example 2-21 Applying a rule set to the Public zone.

Show the public zone policy configuration.	vyatta@R1# show zone-policy zone public
	description "PUBLIC ZONE"
	from dmz {
	firewall {
	name to_public
	}
	}
	from private {
	firewall {
	name to_public
	}
	}
	interface eth3

Filtering Traffic To and From the Local Zone

The Local Zone is a special zone which refers to the Vyatta system itself. By default, all traffic destined for the system and originating from the system is allowed. In Figure 2-4 we see arrows depicting traffic flows to and from the transit zones (Private, DMZ, and Public) as well as to and from the Local Zone.

Figure 2-4 Default traffic to/from the Local Zone



To create a configuration that restricts access to the Vyatta system to hosts located within the Private Zone, perform the following steps in configuration mode.

Example 2-22 Restricting Vyatta system access to hosts located in the Private Zone.

Step	Command
Create the configuration node for the private_to_vyatta rule set and give a description for the rule set.	`vyatta@R1# set firewall name private_to_vyatta description “filter traffic from PRIVATE zone to local-zone”`
Allow all traffic.	`vyatta@R1# set firewall name private_to_vyatta rule 1 action accept`
Commit the configuration.	`vyatta@R1# commit`

Show the private_to_vyatta firewall configuration.	`vyatta@R1# show firewall name private_to_vyatta`
	` description "filter traffic from PRIVATE zone to local-zone"`
	` rule 1{`
	`  action accept`
	` }`

Apply the private_to_vyatta rule set to traffic from the Private Zone to the Local Zone.	`vyatta@R1# set zone-policy zone vyatta from private firewall name private_to_vyatta`
Set the Local Zone.	`vyatta@R1# set zone-policy zone vyatta local-zone`
Commit the configuration.	`vyatta@R1# commit`

Show the Local Zone policy configuration.	`vyatta@R1# show zone-policy zone vyatta`
	` from private {`
	`  firewall {`
	`  name private_to_vyatta`
	`  }`
	` }`
	` local-zone`

At this point, only traffic from the Private Zone destined for the Vyatta system is allowed. Traffic from all other zones is dropped. However, all traffic originating from the Vyatta system is still allowed to all zones.

NOTE *Care should be taken when defining the local-zone. If you are configuring the system via a remote connection (e.g. via ssh) and restrict access from the zone you are in your session will be dropped. You must make sure that traffic from the zone you are in to to the Vyatta system is allowed.*

Be aware that there are services (e.g. DNS forwarding and Web Proxy) that terminate connections to them within the Vyatta system and then initiate connections to another host. In the case of DNS forwarding, packets destined to the router for lookup of a non-cached DNS entry result in the DNS forwarder initiating a connection to the external name-server to retrieve the DNS entry and then pass it back to the originating client. In the example configuration above where packets to the router are allowed only from the PRIVATE zone, DNS lookups coming back to

the router from an external name-server in the PUBLIC zone would be dropped. Thus, to allow packets destined for the router from the PUBLIC zone, we define a rule set and apply it in the local-zone as follows:

Example 2-23 Filtering traffic from the Public Zone to the Vyatta system.

Step	Command
Create the configuration node for the public_to_vyatta rule set and give a description for the rule set.	`vyatta@R1# set firewall name public_to_vyatta description “filter traffic from PUBLIC zone to local-zone”`

Allow the specified traffic.	`vyatta@R1# set firewall name public_to_vyatta rule 1 action accept`
	`vyatta@R1# set firewall name public_to_vyatta rule 1 protocol all`
	`vyatta@R1# set firewall name public_to_vyatta rule 1 state established enable`
	`vyatta@R1# set firewall name public_to_vyatta rule 1 state related enable`

Commit the configuration.	`vyatta@R1# commit`

Show the public_to_vyatta firewall configuration.	`vyatta@R1# show firewall name public_to_vyatta`
	`description "filter traffic from PUBLIC zone to local-zone"`
	`rule 1{`
	`action accept`
	`protocol all`
	`state {`
	`established enable`
	`related enable`
	`}`
	`}`

Apply the public_to_vyatta rule set to traffic from the Public Zone to the Local Zone.	`vyatta@R1# set zone-policy zone vyatta from public firewall name public_to_vyatta`
Commit the configuration.	`vyatta@R1# commit`

Example 2-23 Filtering traffic from the Public Zone to the Vyatta system.

Show the new Local Zone policy configuration.	`vyatta@R1# show zone-policy zone vyatta`
	`from private {`
	`  firewall {`
	`  name private_to_vyatta`
	`  }`
	`}`
	`from public {`
	`  firewall {`
	`  name public_to_vyatta`
	`  }`
	`}`
	`local-zone`

By default all traffic originating from the Local Zone is permitted. If you wish to restrict this you must define the local-zone as a "from zone" within the definition of a transit zone. Once the local-zone is used as a "from zone" all traffic from the Vyatta system to all other zones is blocked unless explicitly allowed through the use of a rule set that allows traffic into a specific zone.

For example, to allow traffic from the Vyatta system only to the Private Zone we would do the following:

Example 2-24 Allow traffic from the Vyatta system to the Private Zone.

Step	Command
Create the configuration node for the from_vyatta rule set and give a description for the rule set.	`vyatta@R1# set firewall name from_vyatta description "allow all traffic from local-zone"`
Allow the specified traffic.	`vyatta@R1# set firewall name from_vyatta rule 1 action accept` `vyatta@R1# set firewall name from_vyatta rule 1 protocol all`
Commit the configuration.	`vyatta@R1# commit`

Show the from_vyatta firewall configuration.	`vyatta@R1# show firewall name from_vyatta`
	`description "allow all traffic from local-zone"`
	`rule 1{`
	`  action accept`
	`  protocol all`
	`}`

Apply the from_vyatta rule set to traffic from the Local Zone to the Private Zone.	`vyatta@R1# set zone-policy zone private from vyatta firewall name from_vyatta`
Commit the configuration.	`vyatta@R1# commit`

Example 2-24 Allow traffic from the Vyatta system to the Private Zone.

Show the new Private Zone policy configuration.

```
vyatta@R1# show zone-policy zone private
 description "PRIVATE ZONE"
 from dmz {
    firewall {
        name to_private
    }
 }
 from public {
    firewall {
        name to_private
    }
 }
 from vyatta {
    firewall {
        name from_vyatta
    }
 }
 interface eth0
 interface eth1
```

Remember that the services that require traffic to originate from the Vyatta system require appropriate filtering to those zones from the local-zone. For example, for DNS forwarding to work traffic would have to be permitted from the Vyatta system to the Public Zone.

Considerations for Remote Access VPN

We extend our example by adding a separate zone to handle Remote Access VPN users. We treat these users like users in the Private zone (though it is not necessary to do so). To this end, a separate “vpn” zone is created and policies are applied just like for Private zone users. One difference is that all Remote Access VPN users that access the Vyatta system present as separate L2TP or PPTP interfaces so the **interface** is defined as “l2tp+” or “pptp+”, meaning any L2TP or PPTP interface. In this example we also assume that no interaction is required between the VPN zone and the Private zone. The following configuration shows each of the zones now that the VPN zone is aded.

Example 2-25 Zone policy with VPN zone added.

Step	Command

Show the VPN Zone policy configuration. The “interface l2tp+” means any L2TP connection. The “interface pptp+” means any PPTP connection.	vyatta@R1# show zone-policy zone vpn
	default-action drop
	description "REMOTE ACCESS VPN ZONE"
	from dmz {
	firewall {
	name to_private
	}
	}
	from public {
	firewall {
	name to_private
	}
	}
	from vyatta {
	firewall {
	name from_vyatta
	}
	}
	interface l2tp+
	interface pptp+

Show the DMZ Zone policy configuration (the “from vpn” section has been added).	vyatta@R1# show zone-policy zone dmz
	description "DMZ ZONE"
	from private {
	firewall {
	name private_to_dmz
	}
	}
	from public {
	firewall {
	name public_to_dmz
	}
	}
	from vpn {
	firewall {
	name private_to_dmz
	}
	}
	interface eth2

Example 2-25 Zone policy with VPN zone added.

Show the Private Zone policy configuration (no changes to the Private zone as there is no traffic between Private and VPN zones).

```
vyatta@R1# show zone-policy zone private
 description "PRIVATE ZONE"
 from dmz {
   firewall {
       name to_private
   }
 }
 from public {
   firewall {
       name to_private
   }
 }
 from vyatta {
   firewall {
       name from_vyatta
   }
 }
 interface eth0
 interface eth1
```

Show the Public zone policy configuration (the “from vpn” section has been added).

```
vyatta@R1# show zone-policy zone public
 description "PUBLIC ZONE"
 from dmz {
   firewall {
       name to_public
   }
 }
 from private {
   firewall {
       name to_public
   }
 }
 from vpn {
   firewall {
       name to_public
   }
 }
 interface eth3
```

Example 2-25 Zone policy with VPN zone added.

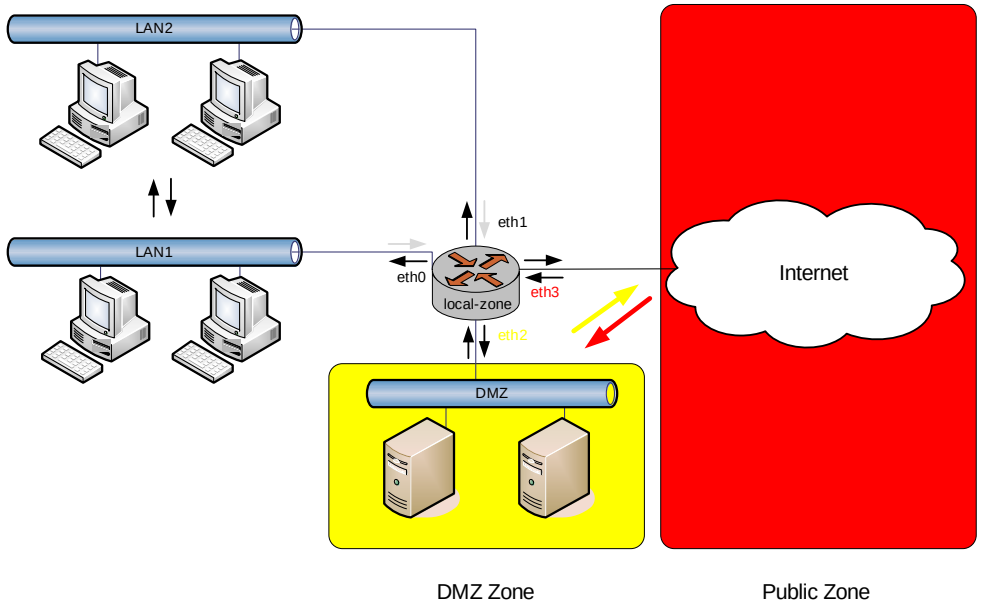
Show the Local Zone policy configuration (the “from vpn” section has been added).

```
vyatta@R1# show zone-policy zone vyatta
 from private {
    firewall {
        name private_to_vyatta
    }
 }
 from public {
    firewall {
        name public_to_vyatta
    }
 }
 from vpn {
    firewall {
        name private_to_vyatta
    }
 }
 local-zone
```

Using Per-Interface Rule Sets with Zone-Based Firewall

On the creation of a zone (transit or local), traffic to that zone is only allowed from another zone by using firewall rule sets to filter traffic from that zone. Thus, interfaces that are not included as part of any zone will not be able to send traffic to any zone. However, traffic between interfaces that are not part of any zone flows freely and can be filtered using per interface firewall rule sets. Consider the example below:

Figure 2-5 Default traffic to/from the Local Zone



There are three zones defined in this topology - DMZ, Public, local-zone. A sample zone-policy configuration for this topology may look something like this:

Example 2-26 Zone policy for topology with three zones (DMZ, Public, and local-zone).

Step	Command

Show the zone policy configuration.	vyatta@R1# show zone-policy
	zone dmz {
	default-action drop
	description "DMZ ZONE"
	from public {
	firewall {
	name public_to_dmz
	}
	}
	interface eth2
	}
	zone public {
	default-action drop
	description "PUBLIC ZONE"
	from dmz {
	firewall {
	name to_public
	}
	}
	interface eth3
	}
	zone vyatta {
	default-action drop
	from dmz {
	firewall {
	name dmz_to_vyatta
	}
	}
	from public {
	firewall {
	name public_to_vyatta
	}
	}
	local-zone
	}

eth0 and eth1 are not part of any zone. Thus, traffic to any of the three zones from these interfaces will be dropped. Traffic flowing between LAN1 and LAN2 will flow freely and unfiltered. Also, traffic going out eth0 and eth1 from any of the zones (DMZ, Public, and local-zone) will flow unfiltered. Now, suppose that we want to reject all traffic from any of the zones going out eth0 and eth1 and also, want to allow just ICMP packets between LAN1 and LAN2. We would configure the system as follows:

Example 2-27 Reject traffic from zones and allow only ICMP between LANs.

Step	Command

Show the allow_ping_only firewall configuration.
NOTE: “not_allowed_nets” is a network group containing subnets of the DMZ and Public zones.
	vyatta@R1# show firewall name allow_ping_only
	description "allow nothing from zones. allow icmp packets between LANs"
	rule 1 {
	action reject
	protocol all
	source {
	group {
	network-group not_allowed_nets
	}
	}
	}
	rule 2 {
	action accept
	icmp {
	type-name any
	}
	protocol icmp
	}

Show the firewall configuration of eth0 and eth1.	vyatta@R1# show interfaces ethernet eth0 firewall
	out {
	name allow_ping_only
	}
	vyatta@R1# show interfaces ethernet eth1 firewall
	out {
	name allow_ping_only
	}

This does not filter traffic originating from the Vyatta system going out interfaces eth0 and eth1. There are no commands to filter traffic originating from the system on a per interface basis. If the zone-policy configuration in this example had the local-zone (zone vyatta) being used as a from zone under DMZ and/or Public then traffic originating from the system would only go out those zones and nothing else.

Using Firewall with VRRP Interfaces

This section presents the following topics:

A VRRP interface is a logical abstraction that allows the system to implement RFC 3768–compliant MAC address behavior. VRRP can be configured with or without VRRP interfaces. To achieve the expected results when filtering traffic, it is important to understand how traffic flows on systems using VRRP.

- If no VRRP interface is designed, traffic flows in and out through physical interface or vif.
- If a VRRP interface is designed, traffic flows in through the VRRP interface and out through the physical interface or vif.

This traffic flow affects how you design and attach firewall rule sets.

Applying a Rule Set to a VRRP Interface

When a host sends a packet to the router, the packet ingresses through the VRRP interface. But when the router sends traffic to the host, traffic egresses through the parent interface or vif.

The firewall rule sets for the VRRP interface and the physical interface are independent. Specifically, packet filtering rules applied to incoming traffic on the parent interface are NOT applied to traffic arriving on the VRRP interface. When designing firewall rule sets for incoming traffic, make sure you apply an appropriate rule set for your VRRP interface; otherwise, all incoming traffic will be unfiltered.

Example 2-1 defined a simple firewall rule set, FWTEST-1, which filters on source IP address. Example 2-28 applies the same rule set to inbound traffic on the VRRP interface. In this example, eth2 is already configured. Specifically:

- It is a member of VRRP group 15.
- It has rule set FWTEST-1 applied for inbound traffic.

To apply the rule set to the VRRP interface, perform the following steps in configuration mode.

Example 2-28 Applying a firewall rule set to a VRRP interface

Step	Command

View initial configuration for eth2. Note the firewall rule set attached to the physical interface.	`vyatta@R1# show interfaces ethernet eth2`
	`address 172.16.1.20/24`
	`duplex auto`
	`firewall {`
	`in {`
	`name FWTEST-1`
	`}`
	`}`
	`hw-id 00:0c:29:c6:a2:59`
	`smp_affinity auto`
	`speed auto`
	`vrrp {`
	`vrrp-group 15 {`
	`advertise-interval 1`
	`interface {`
	`}`
	`preempt true`
	`sync-group test`
	`virtual-address 172.16.1.25`
	`}`
	`}`

Attach the same FW-TEST1 rule set for inbound traffic on the VRRP interface.	`set interfaces ethernet eth2 vrrp vrrp-group 15 firewall in name FWTEST-1`
Commit the configuration.	`vyatta@R1# commit`

Example 2-28 Applying a firewall rule set to a VRRP interface

Show the configuration.

```
vyatta@R1# show interfaces ethernet eth2
 address 172.16.1.20/24
 duplex auto
 firewall {
    in {
        name FWTEST-1
    }
}
 hw-id 00:0c:29:c6:a2:59
 smp_affinity auto
 speed auto
 vrrp {
     vrrp-group 15 {
        advertise-interval 1
        interface {
            firewall {
                in {
                    name FWTEST-1
                }
            }
        }
        preempt true
        sync-group test
        virtual-address 172.16.1.25
    }
}
```

Using VRRP with a Zone-Based Firewall

When a physical interface or vif has a VRRP interface defined, all incoming traffic arrives through the VRRP interface. Zone-based firewalls drop all traffic in and out unless explicitly allowed. Therefore, if you are using VRRP interfaces with zone-based firewall, you must make sure you include the VRRP interfaces in your zone. Otherwise, no incoming traffic will be allowed into the zone; it will all be dropped.

In Example 2-20, the Private zone was defined to include interfaces eth0 and eth1. Example 2-29 adds VRRP interfaces for both eth0 and eth1. In this example:

- Interface eth0 is a member of VRRP group 99.
- Interface eth1 is a member of VRRP group 101.

When you add configuration to a VRRP interface, you do not specify the interface identifier. The system internally constructs the identifier from the name of the parent interface together with the VRRP group ID.

Example 2-29 Adding VRRP interfaces to the Private zone

Step	Command
Add one of the interfaces contained in the Private zone.	`vyatta@R1# set zone-policy zone private interface eth0 vrrp vrrp-group 99 interface`
Add the other interface contained in the Private zone.	`vyatta@R1# set zone-policy zone private interface eth1 vrrp vrrp-group 101 interface`
Commit the configuration.	`vyatta@R1# commit`

Show the configuration.	`vyatta@R1# show zone-policy zone private`
	`description "PRIVATE ZONE"`
	`from dmz {`
	`  firewall {`
	`  name to_private`
	`  }`
	`}`
	`from public {`
	`  firewall {`
	`  name to_private`
	`  }`
	`}`
	`from vyatta {`
	`  firewall {`
	`  name from_vyatta`
	`  }`
	`}`
	`interface eth0`
	`interface eth0v99`
	`interface eth1`
	`interface eth1v101`

Viewing Firewall Information

This section presents the following topics:

- Showing Firewall Instance Information
- Showing Firewall Configuration on Interfaces
- Showing Firewall Configuration

This section includes the following examples:

- Example 2-30 Showing firewall instances
- Example 2-31 Showing firewall configuration on an interface

- Example 2-32 Displaying the “firewall” configuration node

Showing Firewall Instance Information

You can see how firewall instances are set up by using the **show firewall** command in operational mode and specifying the name of the instance. If no instance is specified then all defined instances are displayed.

Example 2-30 shows the information you configured for firewall instance FWTEST-1 and FWTEST-3.

Example 2-30 Showing firewall instances

```
vyatta@R1:~$ show firewall FWTEST-1

Active on (eth0, IN)

State Codes: E - Established, I - Invalid, N - New, R - Related

rule  action  source              destination         proto  state
----  ------  ------              -----------         -----  -----
1     REJECT  172.16.0.26         0.0.0.0/0           all    any
1025  DROP    0.0.0.0/0           0.0.0.0/0           all    any


vyatta@R1:~$ show firewall FWTEST-3

Active on (eth1, LOCAL)

State Codes: E - Established, I - Invalid, N - New, R - Related

rule  action  source              destination         proto  state
----  ------  ------              -----------         -----  -----
1     ACCEPT  10.10.30.46         0.0.0.0/0           tcp    any
                                  dst ports: telnet
1025  DROP    0.0.0.0/0           0.0.0.0/0           all    any


vyatta@R1:~$
```

Showing Firewall Configuration on Interfaces

Example 2-31 shows how firewall instance FWTEST-1 is applied to interface eth0.

Example 2-31 Showing firewall configuration on an interface

```
vyatta@R1# show interfaces ethernet eth0 firewall
    in {
        name FWTEST-1
    }
vyatta@R1#
```

Showing Firewall Configuration

You can always view the information in configuration nodes by using the **show** command in configuration mode. In this case you can view firewall configuration by using the **show firewall** command in configuration mode, as shown in Example 2-32.

Example 2-32 Displaying the “firewall” configuration node

```
vyatta@R1# show firewall
    name FWTEST-1 {
        rule 1 {
            action reject
            source {
                address 172.16.0.26
            }
        }
    }
    name FWTEST-2 {
        rule 1 {
            action accept
            destination {
                address 10.10.40.101
            }
            source {
                address 10.10.30.46
            }
        }
    }
    name FWTEST-3 {
        rule 1 {
            action accept
            destination {
                port telnet
            }
            protocol tcp
            source {
```

```
            address 10.10.30.46
        }
    }
}
name FWTEST-4 {
    rule 1 {
        action accept
        destination {
            address 172.16.0.0/24
        }
        source {
            address 10.10.40.0/24
        }
    }
}
name FWTEST-5 {
    rule 1 {
        action accept
        source {
            mac-addr 00:13:ce:29:be:e7
        }
    }
}
vyatta@R1#
```

Chapter 3: Global Firewall Commands

This chapter describes Vyatta system firewall commands. These commands apply to both IPv4 and IPv6 firewalls.

Global Firewall Commands

This chapter contains the following commands.

Configuration Commands	
Global Configuration Commands	
firewall	Enables or disables firewall on the Vyatta system.
firewall state-policy	Allows you to configure global statefulness policy for traffic that is established, related to an existing connection, or invalid.
Operational Commands	
Global Operational Commands	
clear firewall	Clears statistics associated with a firewall rule set.
show firewall	Displays firewall group information.
show firewall group	Displays firewall group information.

clear firewall

Clears statistics associated with a firewall rule set.

Syntax

clear firewall {**name** *name* | **ipv6-name** *name*} [**rule** *rule-num*] **counters**

Command Mode

Operational mode.

Parameters

name *name*	Clears counters for the specified IPv4 firewall rule set.
ipv6-name *name*	Clears counters for the specified IPv6 rule set.
rule *rule-num*	Clears counters for a specific rule within a rule set.

Default

When no rule is specified, statistics are cleared for all rules in the rule set.

Usage Guidelines

Use this command to clear the statistics associated with a firewall rule set or, optionally, a rule within a firewall rule set.

firewall

Enables or disables firewall on the Vyatta system.

Syntax

set firewall

delete firewall

show firewall

Command Mode

Configuration mode.

Configuration Statement

```
firewall {}
```

Parameters

None.

Default

None.

Usage Guidelines

Use this command to define firewall configuration settings and rule sets, using other **firewall** commands.

Once the firewall rule sets have been defined, they must be applied to interfaces as packet filters using firewall-related **interface** commands. Until a firewall rule set has been applied to an interface, it has no effect on traffic destined for or traversing the system.

Note that after the final user-defined rule in a rule set is executed, an implicit rule of **reject all** takes effect.

Use the **set** form of this command to create firewall configuration.

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

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

firewall state-policy

Allows you to configure global statefulness policy for traffic that is established, related to an existing connection, or invalid.

Syntax

set firewall state-policy {established | related | invalid } action *action* **[log enable]**

delete firewall state-policy [log]

show firewall state-policy

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    state-policy {
        established {
            action {accept|drop|reject}
            log enable
        }
        related {
            action {accept|drop|reject}
            log enable
        }
        invalid {
            action {accept|drop|reject}
            log enable
        }
    }
}
```

Parameters

established	Sets the global statefulness policy for established traffic. Established traffic is traffic belonging to a connection that has already had packets travel in both directions; for example, a reply packet, or an outgoing packet on a connection that has been replied to.

related	Sets the global statefulness policy for related traffic. Related traffic is traffic related to an existing connection; for example, an FTP data transfer connection or an ICMP error.
invalid	Sets the global statefulness policy for invalid traffic. Invalid traffic is packets that cannot be identified for some reason. Examples are ICMP errors that do not correspond to any known connection, or a TCP packet that is not part of an existing TCP connection—for example, a TCP ACK RST or FIN packet that does not correspond to a valid connection in the connection tracking table.
action	The action to be taken when a packet satisfies the criteria specified in the policy. Supported values are as follows: **accept**: Accepts and forwards matched packets. **drop**: Silently drops matched packets. **reject**: Drops matched packets with an ICMP "Port unreachable" message.
log enable	Logs the packets processed by this policy. To disable logging, delete the **log enable** configuration node.

Default

If this statement is not configured, the firewall is stateless. In this case, specific rules governing statefulness can be configured within the rule set.

Usage Guidelines

Use this command to configure a global state policy for traffic associated with established connections, traffic related to established connections, and invalid traffic.

Setting this configuration node makes the firewall globally stateful. You then define policies for established traffic, related traffic, and invalid traffic.

When configured to be stateful, the firewall tracks the state of network connections and traffic flows and allows or restricts traffic based on whether its connection state is known and authorized. For example, when an initiation flow is allowed in one direction, the stateful firewall automatically allows responder flows in the return direction.

The state policy configured applies to all IPv4 and IPv6 traffic destined for the router, originating from the router, or traversing the router. Once the firewall is configured to be globally stateful, this setting overrides any state rules configured within rule sets.

Use the **set** form of this command to configure global state policies for the firewall.

Use the **delete** form of this command to remove state policy configuration and return the firewall to stateless operation. Use the **delete** command also to delete the **log enable** option and disable logging for the specified policy.

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

show firewall

Displays statistics for firewall rule sets.

Syntax

show firewall [**name** *name* | **ipv6-name** *name*] [**rule** *rule-num* | **statistics**]]

Command Mode

Operational mode.

Parameters

name *name*	Displays information for the specified IPv4 firewall rule set.
ipv6-name *name*	Displays information for the specified IPv6 rule set.
rule *rule-num*	Displays information for a specific rule within a rule set.
statistics	Optional. Displays summary statistics information about the specified IPv6 firewall instance.

Default

When used with no option, shows information for all configured firewall rule sets.

Usage Guidelines

Use this command to display statistics information about configured firewall rule sets.

Examples

Example 3-1 shows firewall configuration. The output includes configured global state information plus information about configured firewall rule sets.

Example 3-1 Viewing firewall configuration information

```
vyatta@R1:~$# show firewall

------------------------
Firewall Global Settings
```

```
------------------------

Firewall state-policy for all IPv4 and IPv6 traffic

state           action   log
-----           ------   ---
invalid         drop     enabled
established     accept   disabled
related         accept   disabled

----------------------------
Firewall Rulesets Information
----------------------------

IPv4 Firewall "allow_all":

 Active on (eth1,IN)

 Active on traffic to -
  zone [private] from zones [dmz, public]

(State Codes: E - Established, I - Invalid, N - New, R - Related)

rule  action  source              destination         proto  state
----  ------  ------              -----------         -----  -----
1     ACCEPT  0.0.0.0/0           0.0.0.0/0           all    any
1025  DROP    0.0.0.0/0           0.0.0.0/0           all    any
```

Example 3-2 shows detail for all firewall rule s on R1. In this example, only one rule set (TEST) has been defined.

Example 3-2 Displaying detailed firewall rule information

```
vyatta@R1:~$ show firewall detail
------------------------------------------------------------
IPv4 Firewall "TEST": Active on (eth0,IN)

rule  action   proto  packets  bytes
----  ------   -----  -------  -----
10    accept   all    0        0
  condition - saddr 192.168.0.0/24

20    drop     icmp   0        0
  condition - saddr 192.168.74.0/24

30    accept   tcp    0        0
```

```
  condition - state NEW,ESTABLISHED

1025  drop     all    0        0

vyatta@R1:~$
```

Example 3-3 shows statistics for all firewall rules on R1.

Example 3-3 “show firewall statistics ”: Displaying rule statistics

```
vyatta@R1:~$ show firewall statistics
------------------------------------------------------------
IPv4 Firewall "TEST": Active on (eth0,IN)

rule  packets   bytes     action  source              destination
----  -------   -----     ------  ------              -----------
10    0         0         ACCEPT  192.168.0.0/24      0.0.0.0/0
20    0         0         DROP    192.168.74.0/24     0.0.0.0/0
30    0         0         ACCEPT  0.0.0.0/0           0.0.0.0/0
1025  0         0         DROP    0.0.0.0/0           0.0.0.0/0

vyatta@R1:~$
```

show firewall group

Displays firewall group information.

Syntax

show firewall group [*group-name*]

Command Mode

Operational mode.

Parameters

group-name	The name of a specific firewall group.

Default

All groups are displayed.

Usage Guidelines

Use this command to display firewall group information. Supported group types include address groups, network groups, and port groups.

Examples

Example 3-4 shows all firewall groups on R1.

Example 3-4 "show firewall group": Displaying information on all defined firewall groups.

```
vyatta@R1:~$ show firewall group
Name       : SERVERS
Type       : address
Description: My set of blocked servers
References : FW1-25-destination
Members    :
             1.1.1.1
             1.1.1.2
             1.1.1.3
             1.1.1.5
             1.1.1.7
             3.3.3.3
```

```
Name        : BAD-NETS
Type        : network
Description: my bad nets
References : none
Members     :
              2.2.0.0/16
              8.8.8.0/24
              9.0.0.0/24
vyatta@R1:~$
```

Chapter 4: IPv4 Firewall Commands

This chapter describes commands for defining IPv4 firewall packet filters on the Vyatta system.

This chapter presents the following topics:

IPv4 Firewall Commands

This chapter contains the following commands.

Configuration Commands	
Interface Commands	
interfaces <interface> firewall <direction> name <fw-name>	Applies an IPv4 firewall instance to the defined interface.
General Detection	
firewall all-ping <state>	Enables or disables response to all IPv4 ICMP Echo Request (ping) messages.
firewall broadcast-ping <state>	Enables or disables response to broadcast IPv4 ICMP Echo Request and Timestamp Request messages.
firewall ip-src-route <state>	Specifies whether to process packets with the Loose Source Route or Strict Source Route IP options.
firewall log-martians <state>	Specifies whether to log packets with invalid addresses.
firewall receive-redirects <state>	Specifies whether to process IPv4 ICMP redirect messages.
firewall send-redirects <state>	Specifies whether to allow sending of IPv4 ICMP redirect messages.
firewall source-validation <state>	Specifies a policy for source validation by reversed path, as defined in RFC 3704.
firewall syn-cookies <state>	Specifies policy for using TCP SYN cookies with IPv4.
Firewall Groups	
firewall group	Defines a group of objects for referencing in firewall rules.
firewall group address-group <group-name>	Defines a group of IP addresses for referencing in firewall rules.
firewall group network-group <group-name>	Defines a group of networks for referencing in firewall rules.

firewall group port-group <group-name>	Defines a a group of ports for referencing in firewall rules.
Rules and Rule Sets	
firewall name <name>	Defines an IPv4 firewall rule set.
firewall name <name> rule <rule-num>	Defines the actions, processing behavior, and match conditions for an IPv4 firewall rule set.
firewall name <name> rule <rule-num> description <desc>	Specifies a brief description for an IPv4 firewall rule.
firewall name <name> rule <rule-num> disable	Disables a firewall rule.
firewall name <name> rule <rule-num> limit	Specifies traffic rate limiting parameters for a firewall rule.
firewall name <name> rule <rule-num> log <state>	Enables or disables logging of firewall rule actions.
firewall name <name> rule <rule-num> time	Specifies the times at which this rule is applied.
Operational Commands	
See operational commands in "Chapter 3: Global Firewall Commands."	

firewall all-ping <state>

Enables or disables response to all IPv4 ICMP Echo Request (ping) messages.

Syntax

set firewall all-ping {**enable** | **disable**}

delete firewall all-ping

show firewall all-ping

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   all-ping state
}
```

Parameters

enable	The system responds to IPv4 ICMP Echo Request messages.
disable	The system does not respond to IPv4 ICMP Echo Request messages.

Default

The system responds to IPv4 ICMP Echo Request messages.

Usage Guidelines

Use this command to specify whether the system responds to IPv4 ICMP Echo Request messages (pings). This includes all ping messages: unicast, broadcast, or multicast.

Pings are a network tool used to establish the reachability of a device from the local system. Pings are often disallowed as a potential means of Denial of Service (DoS) attacks.

Use the **set** form of this command to enable or disable responses to pings.

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

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

firewall broadcast-ping <state>

Enables or disables response to broadcast IPv4 ICMP Echo Request and Timestamp Request messages.

Syntax

set firewall broadcast-ping {**enable** | **disable**}

delete firewall broadcast-ping

show firewall broadcast-ping

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    broadcast-ping state
}
```

Parameters

enable	The system responds to broadcast IPv4 ICMP Echo and Timestamp Request messages.
disable	The system does not respond to broadcast IPv4 ICMP Echo and Timestamp Request messages.

Default

IPv4 ICMP Echo and Timestamp Request messages are not processed.

Usage Guidelines

Use this command to specify whether the system processes broadcast IPv4 ICMP Echo Request and broadcast IPv4 ICMP Timestamp Request messages.

Pings are a network tool used to establish the reachability of a device from the local system. Pings, and particularly broadcast pings are often disallowed because of the potential of a Denial of Service (DoS) attack. Timestamp requests are used by to

query another device for the current date and time. Broadcast timestamp requests are also often disallowed, both because of the potential for a DoS attack and because the query allows an attacker to learn the date set on the queried machine.

Use the **set** form of this command to specify whether the system responds to broadcast ICMP IPv4 ICMP Echo and Timestamp Request messages.

Use the **delete** form of this command to restore the default behavior for responding to these messages.

Use the **show** form of this command to view configured behavior for responding to these messages.

firewall group

Defines a group of objects for referencing in firewall rules.

Syntax

set firewall group

delete firewall group

show firewall group

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   group {}
}
```

Parameters

None.

Default

None.

Usage Guidelines

Use this command to define a group of objects that can be firewall group configuration. A firewall group is a mechanism for grouping various network objects and matching any of the elements in the group rather than having to specify them individually. You can create groups of addresses, networks, or ports.

The **firewall group** configuration node is a multi-node: you can define multiple groups by creating multiple **firewall group** configuration nodes.

Use the **set** form of this command to create the firewall group configuration.

Use the **delete** form of this command to remove a firewall group.

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

firewall group address-group <group-name>

Defines a group of IP addresses for referencing in firewall rules.

Syntax

set firewall group address-group *group-name* {**address** *address* | **description** *desc*}

delete firewall group address-group *group-name* {**address** *address* | **description**}

show firewall group address-group *group-name* {**address** *address* | **description**}

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    group {
        address-group group-name {
            address address
            description desc
        }
    }
}
```

Parameters

group-name	Mandatory. The name of the firewall address group.
address *address*	Mandatory. Adds the specified IPv4 address or range of IPv4 addresses to the specified address group. IPv4 address ranges are specified by separating two contiguous IPv4 addresses with a hyphen; for example, 10.0.0.1-10.0.0.50.
description *desc*	Allows you to specify a brief description for the address group.

Default

None.

Usage Guidelines

Use this command to specify an address group. An address group is a collection of host IP addresses and address ranges which, once defined, can be collectively referenced within a firewall command.

An address group is considered matched if the packet address matches any address or address range within the group.

Use the **set** form of this command to specify the address group.

Use the **delete** form of this command to remove the address group or its members.

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

firewall group network-group <group-name>

Defines a group of networks for referencing in firewall rules.

Syntax

set firewall group network-group *group-name* {**network** *ipv4net* | **description** *desc*}

delete firewall group network-group *group-name* {**network** *ipv4net* | **description** *desc*}

show firewall group network-group *group-name* {**network** *ipv4net* | **description**}

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    group {
        network-group group-name {
            description desc
            network ipv4net
        }
    }
}
```

Parameters

group-name	Mandatory. The name of the firewall network group.
network *ipv4net*	Mandatory. Adds an IPv4 network to the specified network group. The format is *ip-address/prefix*.
description *desc*	Allows you to specify a brief description for the network group.

Default

None.

Usage Guidelines

Use this command to define a network group. A network group is a collection of network addresses which, once defined, can be collectively referenced within a firewall command.

A network group is considered matched if the packet address matches any network address or address range within the group.

Use the **set** form of this command to define a network group.

Use the **delete** form of this command to remove the network group or its members.

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

firewall group port-group <group-name>

Defines a a group of ports for referencing in firewall rules.

Syntax

set firewall group port-group *group-name* {**port** *port* | **description** *desc*}

delete firewall group port-group *group-name* {**port** *port* | **description**}

show firewall group port-group *group-name* {**port** *port* | **description**}

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    group {
        port-group group-name {
            description desc
            port port
        }
    }
}
```

Parameters

group-name	Mandatory. The name of the firewall port group.
port *port*	Mandatory. Adds a port number to the specified port group. Supported formats include a port name (any name in **/etc/services**), a port number, or a hyphen-separated range of port numbers; for example, 1001-1050.
description *desc*	Allows you to specify a brief description for the port group.

Default

None.

Usage Guidelines

Use this command to specify a port group. A port group is a collection of port names, port numbers, and port number ranges which, once defined, can be collectively referenced within a firewall command.

A port group is considered matched if the packet port matches any port name or number within the group.

Use the **set** form of this command to specify a port group.

Use the **delete** form of this command to remove the port group or its members.

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

firewall ip-src-route <state>

Specifies whether to process packets with the Loose Source Route or Strict Source Route IP options.

Syntax

set firewall ip-src-route {**enable** | **disable**}

delete firewall ip-src-route

show firewall ip-src-route

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ip-src-route state
}
```

Parameters

enable	Processes packets with source routing IP options set.
disable	Does not process packets with source routing IP options set.

Default

The default is **disable**.

Usage Guidelines

Use this command to specify whether to permit or deny packets with the Loose Source Route or Strict Source Route IP options.

Source routing allows applications to override the routing tables and specify one or more intermediate destinations for outgoing datagrams. This capability is sometimes used for troubleshooting, but renders the network vulnerable to attacks where network traffic is transparently directed to a centralized collection point for packet capture.

Use the **set** form of this command to specify whether or not to process source route IP options.

Use the **delete** form of this command to restore the default behavior for source route IP options.

Use the **show** form of this command to view source route IP option configuration.

firewall log-martians <state>

Specifies whether to log packets with invalid addresses.

Syntax

set firewall log-martians *state*

delete firewall log-martians

show firewall log-martians

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   log-martians state
}
```

Parameters

state	Specifies whether or not to record packets with invalid addresses in the log. Supported values are as follows: **enable:** Logs packets with invalid addresses. **disable:** Does not log packets with invalid addresses.

Default

Packets with invalid addresses are logged.

Usage Guidelines

Use this command to specify whether to log packets with invalid addresses.

Use the **set** form of this command to set the logging behavior for packets with invalid addresses.

Use the **delete** form of this command to restore the default behavior for packets with invalid addresses.

Use the **show** form of this command to view configuration information for packets with invalid addresses.

firewall name <name>

Defines an IPv4 firewall rule set.

Syntax

set firewall name *name* [**enable-default-log**]

delete firewall name [*name* [**enable-default-log**]]

show firewall name [*name*] [**enable-default-log**]]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        enable-default-log
    }
}
```

Parameters

name	Multinode. The name of the firewall rule set. The name must not contain a space or any other of the following special characters: “|”, “;”, “&”, “$”, “<“, or “>”. The name can be up to 28 characters long. You can define multiple IPv4 firewall rule sets by creating more than one **name** configuration node.
enable-default-log	Logs packets that reach the default action. By default, packets reaching the default action are not logged.

Default

None.

Usage Guidelines

Use this command to define an IPv4 firewall rule set.

A firewall rule set is a named collection of up to 9999 packet-filtering rules. Following the configurable rules is an implicit rule, rule 10000, which denies all traffic.

NOTE *The "deny all" rule stays in effect until every reference to the rule set is removed; that is, until every packet filter referencing the rule set has been removed from all interfaces.*

Use the **set** form of this command to create or modify an IPv4 firewall rule set.

Use the **delete** form of this command to remove an IPv4 firewall rule set.

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

firewall name <name> rule <rule-num>

Defines the actions, processing behavior, and match conditions for an IPv4 firewall rule set.

Syntax

set firewall name *name* **rule** *rule-num* [**action** *action* | *match-criteria*]

delete firewall name *name* **rule** *rule-num* [**action** | *match-criteria*]

show firewall name *name* **rule** *rule-num* [**action** | *match-criteria*]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num
            action {
                accept
                drop
                inspect
                reject
            }
            destination {
                address address
                group {
                    address-group addr-group-name
                    network-group net-group-name
                    port-group port-group-name
                }
                port port
            }
            fragment {
                match-frag
                match-non-frag
            }
            icmp {
                type type
                code code
                type-name type-name
            }
            ipsec {
```

```
            match-ipsec
            match-none
        }
        protocol protocol
        recent {
            count count
            time seconds
        }
        source {
            address address
            group {
                address-group addr-group-name
                network-group net-group-name
                port-group port-group-name
            }
            mac-address mac-addr
            port port
        }
        state {
                established state
                invalid state
                new state
                related state
        }
        tcp {
            flags flags
        }
    }
  }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.

action *action*	Specifies the action to be performed when a packet satisfies the match criteria specified in the rule. Supported values are as follows: **accept**: Accepts the packet. **drop**: Drops the packet silently. **inspect**: Forwards matched packets to the intrusion protection system (IPS). Packets forwarded to the IPS are processed by the **content-inspection traffic-filter** command. **reject**: Drops matched packets with an ICMP "Destination Unreachable" message. Exactly one of **action** must be specified. The system does not enforce this at commit time but the configuration will not function unless one of **action** is specified.
match-criteria	One or more match criteria for packets. The supported match criteria are described in Table 4-1. Each match criterion must be set separately, using a separate configuration operation. That is, each match criterion is an independent parameter in the rule's configuration tree, either set or not set.

Table 4-1 shows the matching criteria supported for packets.

Table 4-1 Match Criteria for Packets

Construction	Description
destination address *addr*	The destination address to match. Supported formats are as follows: *ip-address*: An IPv4 address. *ip-address/prefix*: A network address, where 0.0.0.0/0 matches any network. *ip-address–ip-address*: A range of contiguous IP addresses; for example, 192.168.1.1–192.168.1.150. **!***ip-address*: Matches all IP addresses except the one specified. **!***ip-address/prefix*: Matches all network addresses except the one specified. **!***ip-address–ip-address*: Matches all IP addresses except those in the specified range. If both address and port are specified, the packet is considered a match only if both the address and the port match.

Table 4-1 Match Criteria for Packets

Construction	Description
destination port *port*	Applicable only when the protocol is TCP or UDP. The destination port to match. Supported formats are as follows: *port-name*: Matches the name of an IP service; for example, **http**. You can specify any service name in the file **/etc/services**. *port-num*: Matches a port number. The range is 1 to 65535. *start–end*: Matches the specified range of ports; for example, 1001–1005. You can use a combination of these formats in a comma-separated list. You can also negate the entire list by prepending it with an exclamation mark (“**!**”); for example,**!22,telnet,http,123,1001-1005**. If both address and port are specified, the packet is considered a match only if both the address and the port match.

Table 4-1 Match Criteria for Packets

Construction	Description
destination group *group*	Specifies a group or addresses, ports, or networks for packet destination address. Address, port, and network groups are defined using the **firewall group** command; see the *Vyatta Firewall Reference Guide*. Supported values for *group* are as follows: **address-group** *addr-group-name:* Matches the destination host IP address in packets against the specified address group. The packet is considered a match if it matches any address specified in the group. Only one address group may be specified. The address group must already be defined. **network-group** *net-group-name:* Matches the destination network address in packets against the specified network group. The packet is considered a match if it matches any address specified in the group. Only one network group may be specified. The network group must already be defined. **port-group** *port-group-name*: Matches the destination port packets against the specified port group. The packet is considered a match if it matches any port name or number specified in the group. Only one port group may be specified. The port group must already be defined. Address, port, and network groups are defined using the **firewall group** command; see the *Vyatta Firewall Reference Guide*. A packet is considered a match for an address, a network, or a port group if it matches any host IP address, network address, or port name or number, respectively, in the group. However, if more than one group is specified, the packet must be a match for both groups in order to be considered a match. For example, if an address group and a port group are both specified, the packet's destination must match at least one item in the address group and at least one item in the port group. An address group may be specified together with a port group, and a network group may be specified together with a port group. You cannot specify both an address and a network group.

Table 4-1 Match Criteria for Packets

Construction	Description
fragment *frag-rule*	Specifies matching for fragmented packets. Supported values for *frag-rule* are as follows: **match-frag:** Matches the second and later fragments of a fragmented packet. **match-non-frag:** Matches only the first fragment of a fragmented packet or unfragmented packets.
icmp type *type*	Specifies matching for numeric ICMP types. A valid ICMP type from 0 to 255; for example, 8 (Echo Request), or **0** (Echo Reply). For a list of ICMP codes and types, see "Appendix A: ICMP Types."
icmp code *code*	Specifies matching for numeric ICMP codes. The range is 0 to 255. For a list of ICMP codes and types, see "Appendix A: ICMP Types."
icmp type-name *type-name*	Specifies matching for ICMP type names. For a list of ICMP codes and types, see "Appendix A: ICMP Types." The default is **any**.
ipsec *ipsec-rule*	Specifies whether to match IPSec or non-IPSec packets. Supported values for *ipsec-rule* are as follows: **match-ipsec:** Matches inbound IPSec packets. **match-none:** Matches inbound non-IPSec packets.
protocol *protocol*	Matches packets by protocol. Any protocol literals or numbers listed in the file **/etc/protocols** can be specified. The keywords **tcp_udp** (for both TCP and UDP) and **all** (for all protocols) are also supported. Prefixing the protocol name with the negation operation (the exclamation mark character "**!**") matches every protocol **except** the specified protocol. For example, **!tcp** matches all protocols except TCP. You should take care in using more than rule using the negation operation ("**!**") in combination. Firewall rules are evaluated sequentially, and a sequence of negated rules could result in unexpected behavior.

Table 4-1 Match Criteria for Packets

Construction	Description
recent count *count* **recent time** *seconds*	Matches packets as having been recently seen or not. Supported values for arguments are as follows: *count*: The number of times the same source IP address is recorded within the time period specified by the **time** *seconds* construct. The range is 0 to 255. *seconds*: The period of time, in seconds, to count packets from the same source IP address. The most common use for this match criterion is to help prevent "brute force" attacks where an external device is opening a continuous flow of connections (for example, to SSH) in an attempt to break into the system. Because the external host is an unknown source, the "recent" list allows the system to match packets based on the external host's behavior without knowing its address in advance.
source address *addr*	The source address to match. Supported formats are as follows: *ip-address*: An IPv4 address. *ip-address/prefix*: A network address, where 0.0.0.0/0 matches any network. *ip-address–ip-address*: A range of contiguous IP addresses; for example, 192.168.1.1–192.168.1.150. **!***ip-address*: Matches all IP addresses except the one specified. **!***ip-address/prefix*: Matches all network addresses except the one specified. **!***ip-address–ip-address*: Matches all IP addresses except those in the specified range. If both address and port are specified, the packet is considered a match only if both the address and the port match.

Table 4-1 Match Criteria for Packets

Construction	Description
source group *group*	Specifies a group or addresses, ports, or networks for packet source address. Address, port, and network groups are defined using the **firewall group** command; see the *Vyatta Firewall Reference Guide*. Supported values for *group* are as follows: **address-group** *addr-group-name:* Matches the source host IP address in packets against the specified address group. The packet is considered a match if it matches any address specified in the group. Only one address group may be specified. The address group must already be defined. **network-group** *net-group-name:* Matches the source network address in packets against the specified network group. The packet is considered a match if it matches any address specified in the group. Only one network group may be specified. The network group must already be defined. **port-group** *port-group-name*: Matches the source port packets against the specified port group. The packet is considered a match if it matches any port name or number specified in the group. Only one port group may be specified. The port group must already be defined. Address, port, and network groups are defined using the **firewall group** command; see the *Vyatta Firewall Reference Guide*. A packet is considered a match for an address, a network, or a port group if it matches any host IP address, network address, or port name or number, respectively, in the group. However, if more than one group is specified, the packet must be a match for both groups in order to be considered a match. For example, if an address group and a port group are both specified, the packet's destination must match at least one item in the address group and at least one item in the port group. An address group may be specified together with a port group, and a network group may be specified together with a port group. You cannot specify both an address and a network group.
source mac-address *mac-addr*	Matches the media access control (MAC) address in the source address. The format is 6 colon-separated 8-bit numbers in hexadecimal; for example, 00:0a:59:9a:f2:ba.

Table 4-1 Match Criteria for Packets

Construction	Description
source port *port*	Applicable only when the protocol is TCP or UDP. The source port to match. Supported formats are as follows: *port-name*: Matches the name of an IP service; for example, **http**. You can specify any service name in the file **/etc/services**. *port-num*: Matches a port number. The range is 1 to 65535. *start–end*: Matches the specified range of ports; for example, 1001–1005. You can use a combination of these formats in a comma-separated list. You can also negate the entire list by prepending it with an exclamation mark (“**!**”); for example,**!22,telnet,http,123,1001-1005**. If both address and port are specified, the packet is considered a match only if both the address and the port match.
state established *state*	Matches or fails to established packets, depending on the value of *state*. Established packets are packets that are part of a connection that has seen packets in both directions; for example, a reply packet, or an outgoing packet on a connection that has been replied to. Supported values for *state* are as follows: **enable**: Matches established flows. **disable**: Does not match established flows.
state invalid *state*	Matches or fails to match invalid packets, depending on the value of *state*. Invalid packets are packets that could not be identified for some reason. Reasons might include the system running out of resource or ICMP errors that do not correspond to any known connection. Generally these packets should be dropped. Supported values for *state* are as follows: **enable**: Matches invalid flows. **disable**: Does not match invalid flows.

Table 4-1 Match Criteria for Packets

Construction	Description
state new *state*	Matches or fails to match new packets, depending on the value of *state*. New packets are packets that are creating new connections. For example, for TCP, this will be packets with the SYN flag set. Supported values for *state* are as follows: **enable**: Matches new flows. **disable**: Does not match new flows.
state related *state*	Matches or fails to match related packets, depending on the value of *state*. Related packets are packets related to existing connections. Supported values for *state* are as follows: **enable**: Matches related flows. **disable**: Does not match related flows.
tcp *flags*	Matches the specified TCP flags in a packet. Supported values are **SYN**, **ACK**, **FIN**, **RST**, **URG**, **PSH**, and **ALL**. You can specify more than one flag in a comma-separated list. Prefixing the flag name with the negation operator ("**!**") matches packets with the specified flag unset. You can also use the negation operator ("**!**") to match packets not using a given TCP flag. For example, the list **SYN, !ACK, !FIN, !RST** matches only packets with the **SYN** flag set and the **ACK**, **FIN**, and **RST** flags unset. **ALL** can be used to check if all flags are set and **!ALL** can be used to check for no flags set.

Default

None.

Usage Guidelines

Use this command to define a rule within an IPv4 firewall rule set.

A firewall rule set consists of up to 9999 configurable rules. Following the last configured rule, a system rule (rule 10000) with an action of "deny all" is applied.

Firewall rules are executed in numeric sequence, from lowest to highest. You cannot directly change a rule number, because it is the identifier of a configuration node; however, you can renumber rules using the **rename** command.

To avoid having to renumber firewall rules, a good practice is to number rules in increments of 10. This allows room for the insertion of new rules within the rule set.

Use the **set** form of this command to create or modify a firewall rule within an IPv4 firewall rule set.

Use the **delete** form of this command to remove a rule from an IPv4 firewall rule set.

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

firewall name <name> rule <rule-num> description <desc>

Specifies a brief description for an IPv4 firewall rule.

Syntax

set firewall name *name* **rule** *rule-num* **description** *desc*

delete firewall name *name* **rule** *rule-num* **description**

show firewall name *name* **rule** *rule-num* **description**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num {
            description desc
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
desc	A brief description for this rule. If the description contains spaces, it must be enclosed in double quotes.

Default

None.

Usage Guidelines

Use this command to specify a brief description for a firewall rule.

Use the **set** form of this command to set the description.

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

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

firewall name <name> rule <rule-num> disable

Disables a firewall rule.

Syntax

set firewall name *name* **rule** *rule-num* **disable**

delete firewall name *name* **rule** *rule-num* **disable**

show firewall name *name* **rule** *rule-num*

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num {
            disable
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
disable	Disables the specified firewall rule.

Default

The rule is enabled.

Usage Guidelines

Use this command to disable a firewall rule. This is a useful way to test how the firewall performs without a specific rule without having to delete and then re-enter the rule.

Use the **set** form of this command to disable a firewall rule.

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

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

firewall name <name> rule <rule-num> limit

Specifies traffic rate limiting parameters for a firewall rule.

Syntax

set firewall name *name* **rule** *rule-num* **limit** {**burst** *size* | **rate** *rate*}

delete firewall name *name* **rule** *rule-num* **limit** [**burst** | **rate**]

show firewall name *name* **rule** *rule-num* **limit** [**burst** | **rate**]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num {
            limit {
                burst size
                rate rate
            }
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
size	The size of the burst buffer. This is the maximum number of packets that can be sent as a burst in excess of the specified token rate given available tokens in the buffer. The default is 1, which provides no bursting above the specified rate.
rate	The maximum average rate of data traffic for packets matching the rule. Supported time units are: second, minute, hour, and day. The rate is specified in the format “X/<time unit>”. For example “2/second” limits the packets matching the rule to two per second.

Default

No imposed limit.

Usage Guidelines

Use this command to limit the traffic rate of packets matching the rule. The limit option employs the Token Bucket Filter (TBF) queuing mechanism within firewall to limit the rate of incoming packets to an administatively set rate but with the possibility of allowing short bursts in excess of this rate.

The TBF implementation consists of a buffer (bucket), constantly filled by some virtual pieces of information called tokens, at a specific rate (token rate). The most important parameter of the bucket is its size, that is the number of tokens it can store. Each arriving token collects one incoming data packet from the data queue and is then deleted from the bucket. Associating this algorithm with the two flows -- token and data, gives us three possible scenarios:

1) The data arrives in the TBF at a rate that's equal to the rate of incoming tokens. In this case each incoming packet has its matching token and passes the queue without delay.

2) The data arrives in the TBF at a lower rate than the token rate. Only a part of the tokens are deleted at output of each data packet that's sent out the queue, so the tokens accumulate, up to the bucket size. The unused tokens can then be used to send data at a speed that's exceeding the standard token rate, in case short data bursts occur.

3) The data arrives in the TBF at a greater rate than the token rate. This means that the bucket will soon be devoid of tokens, which causes the TBF to throttle itself for a while. This is called an 'overlimit situation'. If packets keep coming in, packets will start to get dropped.

The limit option "rate" relates to the "token rate" as described in the above algorithm while the limit option "burst" relates to the "bucket size". The implementation of these values is explained below :

rate - If set, this rule will match packets at the specified maximum average rate. Any of the following time units can be used to specify rate : second, minute, hour, day.

For example, a value of 1/second implies that the rule be matched at an average of once per second.

burst - If set, this rule will match packets specified by this value in excess of rate. By default, this value is set to 1. so if you don't want to bother with short bursts of packets and want to simply rate limit at the specified rate then you do not have to worry about this option.

Use the **set** form of this command to specify the traffic limit for the specified rule.

Use the **delete** form of this command to remove the traffic limit for the specified rule.

Use the **show** form of this command to view the traffic limit for the specified rule.

firewall name <name> rule <rule-num> log <state>

Enables or disables logging of firewall rule actions.

Syntax

set firewall name *name* **rule** *rule-num* **log** *state*

delete firewall name *name* **rule** *rule-num* **log**

show firewall name *name* **rule** *rule-num* **log**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num {
            log state
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
state	Enables or disables logging of firewall actions. Supported values are as follows: **enable**: Log when action is taken. **disable**: Do not log when action is taken.

Default

Actions are not logged.

Usage Guidelines

Use this command to log packets that reach the default action.

Use the **set** form of this command to log packets that reach the default action.

Use the **delete** form of this command to restore the default behavior for packets that reach the default action.

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

firewall name <name> rule <rule-num> time

Specifies the times at which this rule is applied.

Syntax

set firewall name *name* **rule** *rule-num* **time** {**monthdays** *days-of-month* | **startdate** *date* | **starttime** *time* | **stopdate** *date* | **stoptime** *time* | *utc* | **weekdays** *days-of-week*}

delete firewall name *name* **rule** *rule-num* **time**

show firewall name *name* **rule** *rule-num* **time**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    name name {
        rule rule-num {
            time {
                monthdays days-of-month
                startdate date
                starttime time
                stopdate date
                stoptime time
                utc
                weekdays days-of-week
            }
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
monthdays *days-of-month*	Specifies which days in the month the rule will be applied. Supported values are days of the month (1 to 31) within a comma-separated list (e.g. 2,12,21). The "!" character can be used to negate a list of values (e.g. !2,12,21). This indicates that the firewall rule is to be applied on all but the specified days.

startdate *date*	Specifies the start of a period of time in which the firewall rule will be applied. Set the date, and optionally the time, using one of the following formats: **yyyy-mm-dd** (e.g. 2009-03-12) **yyyy-mm-ddThh:mm:ss** (e.g. 2009-03-12T17:30:00) The default is 1970-01-01. Note that if the time is specified that it is 24 hour format (valid values are from 00:00:00 to 23:59:59). If the time is not specified the default is the start of the day on the specified date (i.e. 00:00:00). Use **stopdate** to end the activation period.
starttime *time*	Specifies the start of a period of time in the day to which the firewall rule will be applied. Set the start time using the following format: **hh:mm:ss** (e.g. 17:30:00) Note that the time is specified in 24 hour format (valid values are from 00:00:00 to 23:59:59). Use **stoptime** to end the activation period.
stopdate *date*	Specifies the end of a period of time in which the firewall rule will be applied. Set the date, and optionally the time, using one of the following formats: **yyyy-mm-dd** (e.g. 2009-03-12) **yyyy-mm-ddThh:mm:ss** (e.g. 2009-03-12T17:30:00) The default is 2038-01-19. Note that if the time is specified that it is 24 hour format (valid values are from 00:00:00 to 23:59:59). If the time is not specified the default is the start of the day on the specified date (i.e. 00:00:00). Use **startdate** to begin the activation period.
stoptime *time*	Specifies the end of a period of time in the day to which the firewall rule will be applied. Set the stop time using the following format: **hh:mm:ss** (e.g. 17:30:00) Note that the time is specified in 24 hour format (valid values are from 00:00:00 to 23:59:59). Use **starttime** to begin the activation period.
utc	Specifies that times given using **startdate**, **stopdate**, **starttime**, and **stoptime**, should be interpreted as UTC time rather than local time.

weekdays *days-of-week*	Specifies which days in the week the rule will be applied. Supported values are days of the week (Mon, Tue, Wed, Thu, Fri, Sat, and Sun) within a comma-separated list (e.g. Mon,Wed,Fri). The "!" character can be used to negate a list of values (e.g. !Mon,Wed,Fri). This indicates that the firewall rule is to be applied on all but the specified days of the week.

Default

The rule is applied at all times.

Usage Guidelines

Use this command to restrict the times during which the rule will be applied. All values are optional and are ANDed when specified.

Use the **set** form of this command to specify the times at which a firewall rule will be applied.

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

Use the **show** form of this command to view time configuration for a firewall rule.

firewall receive-redirects <state>

Specifies whether to process IPv4 ICMP redirect messages.

Syntax

set firewall receive-redirects {**enable** | **disable**}

delete firewall receive-redirects

show firewall receive-redirects

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   receive-redirects state
}
```

Parameters

state	Permits or denies receiving IPv4 ICMP redirect messages. Supported values are as follows: **enable**: Permits IPv4 ICMP redirects to be received. **disable**: Denies IPv4 ICMP redirects from being received.

Default

The default is **disable**.

Usage Guidelines

Use this command to specify whether to accept IPv4 ICMP redirects. ICMP redirects can allow an arbitrary sender to forge packets and alter the system's routing table. This can leave the system open to a man-in-the-middle attack.

Use the **set** form of this command to specify whether to accept IPv4 ICMP redirects.

Use the **delete** form of this command to remove the specified value.

Use the **show** form of this command to view the specified value.

firewall send-redirects <state>

Specifies whether to allow sending of IPv4 ICMP redirect messages.

Syntax

set firewall send-redirects {**enable** | **disable**}

delete firewall send-redirects

show firewall send-redirects

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   send-redirects state
}
```

Parameters

state	Permits or denies transmitting packets IPv4 ICMP redirect messages. Supported values are as follows: **enable**: Permits IPv4 ICMP redirects to be sent. **disable**: Denies IPv4 ICMP redirects from being sent.

Default

The default is **enable**.

Usage Guidelines

Use this command to specify whether to allow sending of IPv4 ICMP redirect messages. Sending a redirect will potentially alter the routing table of the host or router to which the redirect is sent.

Use the **set** form of this command to specify whether to permit or deny the sending IPv4 ICMP redirects.

Use the **delete** form of this command to remove the specified value.

Use the **show** form of this command to view the specified value.

firewall source-validation <state>

Specifies a policy for source validation by reversed path, as defined in RFC 3704.

Syntax

set firewall source-validation {**disable** | **loose** | **strict**}

delete firewall source-validation

show firewall source-validation

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   source-validation state
}
```

Parameters

state	Specifies policy for source validation by reversed path, as specified in RFC3704. Supported values are as follows: **disable**: No source validation is performed. **loose**: Enable Loose Reverse Path Forwarding as defined in RFC3704. **strict**: Enable Strict Reverse Path Forwarding as defined in RFC3704.

Default

The default is **disable**.

Usage Guidelines

Use this command to specify policy for source validation by reversed path, as specified in RFC3704.

Use the **set** form of this command to specify policy for source validation by reversed path, as specified in RFC3704.

Use the **delete** form of this command to remove the specified value.

Use the **show** form of this command to view the specified value.

firewall syn-cookies <state>

Specifies policy for using TCP SYN cookies with IPv4.

Syntax

set firewall syn-cookies {**enable** | **disable**}

delete firewall syn-cookies

show firewall syn-cookies

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   syn-cookies state
}
```

Parameters

state	Enables or disables TCP SYN cookies option. Supported values are as follows: **enable**: Enables TCP SYN cookies with IPv4. **disable**: Disables TCP SYN cookies with IPv4.

Default

The default is **enable**.

Usage Guidelines

Use this command to specify whether to use TCP SYN cookies with IPv4. Enabling this option can help protect the system from a TCP SYN Flood Denial of Service (DoS) attack. To start a TCP connection, a source sends a SYN (synchronize/start) packet. The destination sends back a SYN ACK (synchronize acknowledge). Then the source sends an ACK (acknowledge), and the connection is established. This is referred to as the "TCP three-way handshake."

After a destination server sends a SYN ACK, it uses a connection queue to keep track of the connections waiting to be completed. An attacker can fill up the connection queue by generating phony TCP SYN packets from random IP addresses at a rapid rate. When the connection queue is full, all subsequent TCP services are denied.

When this option is enabled, the system creates a hash entry when it receives a SYN packet, and returns a SYN ACK cookie only, without retaining all the SYN information. When it receives the ACK from the client, it validates it against the hash and, if it is valid, rebuilds the SYN packet information and accepts the packet.

Use the **set** form of this command to specify whether to enable or disable the TCP SYN cookies option.

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

Use the **show** form of this command to view TCP SYN cookies option configuration.

interfaces <interface> firewall <direction> name <fw-name>

Applies an IPv4 firewall instance to the defined interface.

Syntax

set interfaces *interface* **firewall** {**in name** *fw-name* | **local name** *fw-name* | **out name** *fw-name*}

delete interfaces *interface* **firewall** [**in name** | **local name** | **out name**]

show interfaces *interface* **firewall** [**in name** | **local name** | **out name**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces interface {
    firewall {
        in {
            name fw-name
        }
        local {
            name fw-name
        }
        out {
            name fw-name
        }
    }
}
```

Parameters

interface	Mandatory. The type of interface. For detailed keywords and arguments that can be specified as interface types, see the table in the Usage Guidelines below.
in name *fw-name*	Applies the specified IPv4 firewall instance to inbound traffic on the specified interface.

local name *fw-name*	Applies the specified IPv4 firewall instance to traffic arriving on the specified interface and bound for the local system.
out name *fw-name*	Applies the specified IPv4 firewall instance to outbound traffic on the specified interface.

Default

None.

Usage Guidelines

Use this command to apply an IPv4 firewall instance, or rule set, to an interface.

A firewall has no effect on traffic traversing the system or destined to the system until a firewall rule set has been applied to an interface or a vif using this command.

To use the firewall feature, you define a firewall rule set as a named firewall instance, using firewall command. You then apply the firewall instance to interfaces and/or vifs using a statement like this one. Once applied, the instance acts as a packet filter.

The firewall instance will filter packets in one of the following ways, depending on what you specify when you apply it:

- **in.** If you apply the rule set as **in**, the firewall will filter packets entering the interface.
- **out.** If you apply the rule set as **out**, the firewall will filter packets leaving the interface.
- **local.** If you apply the rule set as **local**, the firewall will filter packets destined for the system itself.

For each interface, you can apply up to three firewall instances: one firewall **in** instance, one firewall **out** instance, and one firewall **local** instance.

Make sure the firewall instance you apply to an interface is already defined, or you may experience unintended results. If you apply a firewall instance that does not exist to an interface, the implicit firewall rule of **allow all** will be applied.

The following table shows the syntax and parameters for supported interface types.

Interface Type	Syntax	Parameters
ADSL Bridged Ethernet	adsl *adslx* pvc *pvc-id* bridged-ethernet	*adslx* The name of a Bridged Ethernet- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.
ADSL Classical IPOA	adsl *adslx* pvc *pvc-id* classical-ipoa	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.
ADSL PPPoA	adsl *adslx* pvc *pvc-id* pppoa *num*	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically. *num* The PPPoA unit number. This number must be unique across all PPPoA interfaces. In addition, only one PPPoA instance can be configured on a PVC. PPPoA units range from 0 to 15 and the resulting interfaces are named pppoa0 to pppoa15.
ADSL PPPoE	adsl *adslx* pvc *pvc-id* pppoe *num*	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Bonding	bonding *bondx*	*bondx* The identifier for the bonding interface. Supported values are **bond0** through **bond99**.
Bonding Vif	bonding *bondx* vif *vlan-id*	*bondx* The identifier for the bonding interface. Supported values are **bond0** through **bond99**. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094.

Interface Type	Syntax	Parameters
Bridge	bridge *brx*	*brx* The name of a Bridge group. The range is **br0** through **br999**.
Ethernet	ethernet *ethx*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system.
Ethernet PPPoE	ethernet *ethx* pppoe *num*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Ethernet Vif	ethernet *ethx* vif *vlan-id*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094.
Ethernet Vif PPPoE	ethernet *ethx* vif *vlan-id* pppoe *num*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Loopback	loopback *lo*	*lo* The name of the loopback interface.
Multilink	multilink *mlx* vif *1*	*mlx* The identifier of the multilink bundle. You can create up to two multilink bundles. Supported values are **ml0** (“em ell zero”) through **ml23** (“em ell twenty-three”). *1* The identifier of the virtual interface. Currently, only one vif is supported for multilink interfaces, and the identifier must be 1. The vif must already have been defined.
OpenVPN	openvpn *vtunx*	*vtunx* The identifier for the OpenVPN interface. This may be **vtun0** to **vtun*x***, where ***x*** is a non-negative integer.
Pseudo-Ethernet	pseudo-ethernet *pethx*	*pethx* The name of a pseudo-Ethernet interface. The range is **peth0** through **peth999**.
Serial Cisco HDLC	serial *wanx* cisco-hdlc vif *1*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *1* The identifier of the virtual interface. Currently, only one vif is supported for Cisco HDLC interfaces, and the identifier must be 1. The vif must already have been defined.

Interface Type	Syntax	Parameters
Serial Frame Relay	serial *wanx* frame-relay vif *dlci*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *dlci* The identifier of the virtual interface. For Frame Relay interfaces, this is the DLCI number for the interface. the range is 16 to 991. The vif must already have been defined.
Serial PPP	serial *wanx* ppp vif *1*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *1* The identifier of the virtual interface. Currently, only one vif is supported for point-to-point interfaces, and the identifier must be 1. The vif must already have been defined.
Tunnel	tunnel *tunx*	*tunx* An identifier for the tunnel interface you are defining. This may be **tun0** to **tun*x***, where ***x*** is a non-negative integer.
Virtual Tunnel	vti *vtix*	*vtix* An identifier for the virtual tunnel interface you are defining. This may be **vti0** to **vti*x***, where ***x*** is a non-negative integer. **Note:** This interface does not support IPv6.
VRRP	interface *parent-if* vrrp vrrp-group *group* interface	*parent-if* The type and identifier of the parent interface; for example, **ethernet eth0** or **bonding bond0**. *group* The VRRP group identifier. The name of the VRRP interface is not specified. The system internally constructs the interface name from the parent interface identifier plus the VRRP group number—for example, **eth0v99**, **eth0.15v99**, **bond0v99**, ot **bond0.15v99**. Note that VRRP interfaces support the same feature set as the parent interface does.
Wireless	wireless *wlanx*	*wlanx* The identifier for the wireless interface you are using. This may be **wlan0** to **wlan999**.
Wireless Modem	wirelessmodem *wlmx*	*wlmx* The identifier for the wireless modem interface you are using. This may be **wlm0** to **wlm999**.

Use the **set** form of this command to apply an IPv4 firewall instance to an interface.

Use the **delete** form of this command to remove an IPv4 firewall instance from an interface.

Use the **show** form of this command to view an IPv4 firewall configuration for an interface.

Chapter 5: IPv6 Firewall Commands

This chapter describes commands for defining IPv6 firewall packet filters on the Vyatta system.

This chapter presents the following topics:

IPv6 Firewall Commands

This chapter contains the following commands.

Configuration Commands	
Interface Commands	
interfaces <interface> firewall <direction> ipv6-name <fw-name>	Applies an IPv6 firewall instance to the defined interface.
General Detection	
firewall ipv6-receive-redirects <state>	Specifies whether to process received ICMPv6 redirect messages.
firewall ipv6-src-route <state>	Specifies whether to process IPv6 packets with routing extension header.
Rules and Rule Sets	
firewall ipv6-name <name>	Defines an IPv6 firewall rule set.
firewall ipv6-name <name> rule <rule-num>	Defines the actions, processing behavior, and match conditions for an IPv6 firewall rule set.
firewall ipv6-name <name> rule <rule-num> description <desc>	Specifies a brief description for an IPv6 firewall rule.
firewall ipv6-name <name> rule <rule-num> disable	Disables the IPv6 firewall rule.
firewall ipv6-name <name> rule <rule-num> limit	Specifies traffic rate limiting parameters for an IPv6 firewall rule.
firewall ipv6-name <name> rule <rule-num> log <state>	Enables or disables logging of IPv6 firewall rule actions.
firewall ipv6-name <name> rule <rule-num> time	Specifies the times at which this rule is applied.
Operational Commands	
See operational commands in "Chapter 3: Global Firewall Commands."	

firewall ipv6-name <name>

Defines an IPv6 firewall rule set.

Syntax

set firewall ipv6-name *name* [**enable-default-log**]

delete firewall ipv6-name [*name*] [**enable-default-log**]]

show firewall ipv6-name [*name*] [**enable-default-log**]]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {}
        enable-default-log
    }
}
```

Parameters

name	Multinode. The name of the firewall rule set. The name must not contain a space or any other of the following special characters: “|”, “;”, “&”, “$”, “<“, or “>”. The name can be up to 28 characters long. You can define multiple IPv6 firewall rule sets by creating more than one **name** configuration node.
enable-default-log	Logs packets that reach the default action. By default, packets reaching the default action are not logged.

Default

None.

Usage Guidelines

Use this command to define an IPv6 firewall rule set.

A firewall rule set is a named collection of up to 9999 packet-filtering rules . Following the configurable rules is an implicit rule, rule 10000, which denies all traffic.

NOTE *The "deny all" rule stays in effect until every reference to the rule set is removed; that is, until every packet filter referencing the rule set has been removed from all interfaces.*

Use the **set** form of this command to create or modify an IPv6 firewall rule set.

Use the **delete** form of this command to remove an IPv6 firewall rule set.

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

firewall ipv6-name <name> rule <rule-num>

Defines the actions, processing behavior, and match conditions for an IPv6 firewall rule set.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* [**action** *action* | *match-criteria*]

delete firewall ipv6-name *name* **rule** [*rule-num*] [**action** | *match-criteria*]

show firewall ipv6-name *name* **rule** [*rule-num*] [**action** | *match-criteria*]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num
            action {
                accept
                drop
                inspect
                reject
            }
            destination {
                address address
                port port
            }
            icmpv6 {
                type type
            }
            ipsec {
                match-ipsec
                match-none
            }
            protocol protocol
            recent {
                count count
                time seconds
            }
            source {
                address address
                mac-address mac-addr
```

```
            port port
        }
        state {
            established state
            invalid state
            new state
            related state
        }
        tcp {
            flags flags
        }
    }
  }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	Multinode. The numeric identifier of the rule. Rule numbers determine the order in which rules are executed. Each rule must have a unique rule number. The range is 1 to 9999. You can define multiple rules by creating more than one **rule** configuration node.
action *action*	Specifies the action to be performed when a packet satisfies the match criteria specified in the rule. Supported values are as follows: **accept**: Accepts the packet. **drop**: Drops the packet silently. **inspect**: Forwards matched packets to the intrusion protection system (IPS). Packets forwarded to the IPS are processed by the **content-inspection traffic-filter** command. **reject**: Drops matched packets with an ICMP "Destination Unreachable" message. Exactly one of **action** must be specified. The system does not enforce this at commit time but the configuration will not function unless one of **action** is specified.

match-criteria	One or more match criteria for packets. The supported match criteria are described in Table 5-1. Each match criterion must be set separately, using a separate configuration operation. That is, each match criterion is an independent parameter in the rule's configuration tree, either set or not set.

Table 5-1 shows the matching criteria supported for packets.

Table 5-1 Match Criteria for Packets

Construction	Description
destination address *addr*	The destination address to match. Supported formats are as follows: *ipv6-address*: Matches the specified IPv6 address; for example, fe80::20c:29fe:fe47:f89. *ipv6-address/prefix*: A network address, where ::/0 matches any network; for example, fe80::20c:29fe:fe47:f88/64 *ipv6-address–ipv6-address*: Matches a range of contiguous IP addresses; for example, fe80::20c:29fe:fe47:f00–fe80::20c:29fe:fe47:f89. **!***ipv6-address*: Matches all IP addresses except the one specified. **!***ipv6-address/prefix*: Matches all network addresses except the one specified. **!***ipv6-address–ipv6-address*: Matches all IP addresses except those in the specified range. If both address and port are specified, the packet is considered a match only if both the address and the port match.

Table 5-1 Match Criteria for Packets

Construction	Description
destination port *port*	Applicable only when the protocol is TCP or UDP. The destination port to match. Supported formats are as follows: *port-name*: Matches the name of an IP service; for example, **http**. You can specify any service name in the file **/etc/services**. *port-num*: Matches a port number. The range is 1 to 65535. *start–end*: Matches the specified range of ports; for example, 1001–1005. You can use a combination of these formats in a comma-separated list. You can also negate the entire list by prepending it with an exclamation mark (“**!**”); for example,**!22,telnet,http,123,1001-1005**. If both address and port are specified, the packet is considered a match only if both the address and the port match.
icmpv6 type *type*	A valid ICMPv6 type code from 0 to 255; for example, **128** (Echo Request), or a type/code pair (each from 0 to 255); for example, **1/4** (Port unreachable). Alternatively, you can specify an ICMPv6 type code literal; for example, **echo-request** (Echo Request). For a list of ICMP codes and types, see “Appendix B: ICMPv6 Types.”
ipsec *ipsec-rule*	Specifies whether to match IPSec or non-IPSec packets. Supported values for *ipsec-rule* are as follows: **match-ipsec:** Matches inbound IPSec packets. **match-none:** Matches inbound non-IPSec packets.

Table 5-1 Match Criteria for Packets

Construction	Description
protocol *protocol*	Matches packets by protocol. Any protocol literals or numbers listed in the file **/etc/protocols** can be specified. The keywords **icmpv6** and **all** (for all protocols) are also supported. Prefixing the protocol name with the negation operator (the exclamation mark character "**!**") matches every protocol **except** the specified protocol. For example, **!tcp** matches all protocols except TCP. Note that this parameter works slightly different than its IPv4 counterpart. In IPv4, this field strictly matches the "protocol ID" field in the IPv4 header. In IPv6, this parameter matches the "last" next-header field in the IPv6 header chain. This means that if the IPv6 packet has no extension headers, it matches the next-header field in the main IPv6 header. If the packet does have extension headers, the parameter will match the next-header field of the last extension header in the chain. In other words, the parameter always matches the ID of the transport-layer packet that is being carried. You should take care in using more than rule using the negation operation ("!") in combination. Routing policy rules are evaluated sequentially, and a sequence of negated rules could result in unexpected behavior.
recent count *count* **recent time** *seconds*	Matches packets as having been recently seen or not. Supported values for arguments are as follows: *count*: The number of times the same source IP address is recorded within the time period specified by the **time** *seconds* construct. The range is 0 to 255. *seconds*: The period of time, in seconds, to count packets from the same source IP address. The most common use for this match criterion is to help prevent "brute force" attacks where an external device is opening a continuous flow of connections (for example, to SSH) in an attempt to break into the system. Because the external host is an unknown source, the "recent" list allows the system to match packets based on the external host's behavior without knowing its address in advance.

Table 5-1 Match Criteria for Packets

Construction	Description
source address *addr*	The source address to match. Supported formats are as follows: *ipv6-address*: Matches the specified IPv6 address; for example, fe80::20c:29fe:fe47:f89. *ipv6-address/prefix*: A network address, where ::/0 matches any network; for example, fe80::20c:29fe:fe47:f88/64 *ipv6-address–ipv6-address*: Matches a range of contiguous IP addresses; for example, fe80::20c:29fe:fe47:f00–fe80::20c:29fe:fe47:f89. **!***ipv6-address*: Matches all IP addresses except the one specified. **!***ipv6-address/prefix*: Matches all network addresses except the one specified. **!***ipv6-address*–ipv6-address: Matches all IP addresses except those in the specified range. If both address and port are specified, the packet is considered a match only if both the address and the port match.
source mac-address *mac-addr*	Matches the media access control (MAC) address in the source address. The format is 6 colon-separated 8-bit numbers in hexadecimal; for example, 00:0a:59:9a:f2:ba.
source port *port*	Applicable only when the protocol is TCP or UDP. The source port to match. Supported formats are as follows: *port-name*: Matches the name of an IP service; for example, **http**. You can specify any service name in the file **/etc/services**. *port-num*: Matches a port number. The range is 1 to 65535. *start–end*: Matches the specified range of ports; for example, 1001–1005. You can use a combination of these formats in a comma-separated list. You can also negate the entire list by prepending it with an exclamation mark (“**!**”); for example,**!22,telnet,http,123,1001-1005**. If both address and port are specified, the packet is considered a match only if both the address and the port match.

Table 5-1 Match Criteria for Packets

Construction	Description
state established *state*	Matches or fails to established packets, depending on the value of *state*. Established packets are packets that are part of a connection that has seen packets in both directions; for example, a reply packet, or an outgoing packet on a connection that has been replied to. Supported values for *state* are as follows: **enable**: Matches established flows. **disable**: Does not match established flows.
state invalid *state*	Matches or fails to match invalid packets, depending on the value of *state*. Invalid packets are packets that could not be identified for some reason. Reasons might include the system running out of resource or ICMP errors that do not correspond to any known connection. Generally these packets should be dropped. Supported values for *state* are as follows: **enable**: Matches invalid flows. **disable**: Does not match invalid flows.
state new *state*	Matches or fails to match new packets, depending on the value of *state*. New packets are packets that are creating new connections. For example, for TCP, this will be packets with the SYN flag set. Supported values for *state* are as follows: **enable**: Matches new flows. **disable**: Does not match new flows.
state related *state*	Matches or fails to match related packets, depending on the value of *state*. Related packets are packets related to existing connections. Supported values for *state* are as follows: **enable**: Matches related flows. **disable**: Does not match related flows.

Table 5-1 Match Criteria for Packets

Construction	Description
tcp *flags*	Matches the specified TCP flags in a packet. Supported values are **SYN**, **ACK**, **FIN**, **RST**, **URG**, **PSH**, and **ALL**. You can specify more than one flag in a comma-separated list. Prefixing the flag name with the negation operator ("**!**") matches packets with the specified flag unset. You can also use the negation operator ("**!**") to match packets not using a given TCP flag. For example, the list **SYN, !ACK, !FIN, !RST** matches only packets with the **SYN** flag set and the **ACK**, **FIN**, and **RST** flags unset. **ALL** can be used to check if all flags are set and **!ALL** can be used to check for no flags set.

Default

None.

Usage Guidelines

Use this command to define a rule within an IPv6 firewall rule set.

A firewall rule set consists of up to 9999 configurable rules. Following the last configured rule, a system rule (rule 10000) with an action of "deny all" is applied.

Firewall rules are executed in numeric sequence, from lowest to highest. You cannot directly change a rule number, because it is the identifier of a configuration node; however, you can renumber rules using the **rename** command.

To avoid having to renumber firewall rules, a good practice is to number rules in increments of 10. This allows room for the insertion of new rules within the rule set.

Use the **set** form of this command to create or modify a firewall rule within an IPv6 firewall rule set.

Use the **delete** form of this command to remove a rule from an IPv6 firewall rule set.

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

firewall ipv6-name <name> rule <rule-num> description <desc>

Specifies a brief description for an IPv6 firewall rule.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* **description** *desc*

delete firewall ipv6-name *name* **rule** *rule-num* **description**

show firewall ipv6-name *name* **rule** *rule-num* **description**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num {
            description desc
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
desc	A brief description for this rule. If the description contains spaces, it must be enclosed in double quotes.

Default

None.

Usage Guidelines

Use this command to specify a brief description for an IPv6 firewall rule.

Use the **set** form of this command to set the description.

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

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

firewall ipv6-name <name> rule <rule-num> disable

Disables the IPv6 firewall rule.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* **disable**

delete firewall ipv6-name *name* **rule** *rule-num* **disable**

show firewall ipv6-name *name* **rule** *rule-num*

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num {
            disable
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.

Default

The firewall rule is enabled.

Usage Guidelines

Use this command to disable an IPv6 firewall rule.

Use the **set** form of this command to disable the specified rule

Use the **delete** form of this command to enable the specified rule.

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

firewall ipv6-name <name> rule <rule-num> limit

Specifies traffic rate limiting parameters for an IPv6 firewall rule.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* **limit** {**burst** *size* | **rate** *rate*}

delete firewall ipv6-name *name* **rule** *rule-num* **limit** [**burst** | **rate**]

show firewall ipv6-name *name* **rule** *rule-num* **limit** [**burst** | **rate**]

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num {
            limit {
                burst size
                rate rate
            }
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
size	The size of the burst buffer. This is the maximum number of packets that can be sent as a burst in excess of the specified token rate given available tokens in the buffer. The default is 1, which provides no bursting above the specified rate.
rate	The maximum average rate of data traffic for packets matching the rule. Supported time units are: second, minute, hour, and day. The rate is specified in the format “X/<time unit>”. For example “2/second” limits the packets matching the rule to two per second.

Default

No imposed limit.

Usage Guidelines

Use this command to limit the traffic rate of packets matching the rule. The limit option employs the Token Bucket Filter (TBF) queuing mechanism within firewall to limit the rate of incoming packets to an administatively set rate but with the possibility of allowing short bursts in excess of this rate.

The TBF implementation consists of a buffer (bucket), constantly filled by some virtual pieces of information called tokens, at a specific rate (token rate). The most important parameter of the bucket is its size, that is the number of tokens it can store. Each arriving token collects one incoming data packet from the data queue and is then deleted from the bucket. Associating this algorithm with the two flows -- token and data, gives us three possible scenarios:

1) The data arrives in the TBF at a rate that's equal to the rate of incoming tokens. In this case each incoming packet has its matching token and passes the queue without delay.

2) The data arrives in the TBF at a lower rate than the token rate. Only a part of the tokens are deleted at output of each data packet that's sent out the queue, so the tokens accumulate, up to the bucket size. The unused tokens can then be used to send data at a speed that's exceeding the standard token rate, in case short data bursts occur.

3) The data arrives in the TBF at a greater rate than the token rate. This means that the bucket will soon be devoid of tokens, which causes the TBF to throttle itself for a while. This is called an 'overlimit situation'. If packets keep coming in, packets will start to get dropped.

The limit option "rate" relates to the "token rate" as described in the above algorithm while the limit option "burst" relates to the "bucket size". The implementation of these values is explained below :

rate - If set, this rule will match packets at the specified maximum average rate. Any of the following time units can be used to specify rate : second, minute, hour, day.

For example, a value of 1/second implies that the rule be matched at an average of once per second.

burst - If set, this rule will match packets specified by this value in excess of rate. By default, this value is set to 1. so if you don't want to bother with short bursts of packets and want to simply rate limit at the specified rate then you do not have to worry about this option.

Use the **set** form of this command to specify the traffic limit for the specified rule

Use the **delete** form of this command to remove the traffic limit for the specified rule.

Use the **show** form of this command to view the traffic limit for the specified rule.

firewall ipv6-name <name> rule <rule-num> log <state>

Enables or disables logging of IPv6 firewall rule actions.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* **log** *state*

delete firewall ipv6-name *name* **rule** *rule-num* **log**

show firewall ipv6-name *name* **rule** *rule-num* **log**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num {
            log state
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.
state	Enables or disables logging of firewall actions. Supported values are as follows: **enable**: Log when action is taken. **disable**: Do not log when action is taken.

Default

Actions are not logged.

Usage Guidelines

Use this command to enable or disable logging for the specified rule. When enabled, any actions taken will be logged.

Use the **set** form of this command to specify logging for the specified rule

Use the **delete** form of this command to remove the logging value for the specified rule.

Use the **show** form of this command to view the logging value for the specified rule.

firewall ipv6-name <name> rule <rule-num> time

Specifies the times at which this rule is applied.

Syntax

set firewall ipv6-name *name* **rule** *rule-num* **time** {**monthdays** *days-of-month* | **startdate** *date* | **starttime** *time* | **stopdate** *date* | **stoptime** *time* | *utc* | **weekdays** *days-of-week*}

delete firewall ipv6-name *name* **rule** *rule-num* **time**

show firewall ipv6-name *name* **rule** *rule-num* **time**

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
    ipv6-name name {
        rule rule-num {
            time {
                monthdays days-of-month
                startdate date
                starttime time
                stopdate date
                stoptime time
                utc
                weekdays days-of-week
            }
        }
    }
}
```

Parameters

name	The name of the firewall rule set.
rule-num	The numeric identifier of the rule. The range is 1 to 9999.

monthdays *days-of-month*	Specifies which days in the month the rule will be applied. Supported values are days of the month (1 to 31) within a comma-separated list (e.g. 2,12,21). The "!" character can be used to negate a list of values (e.g. !2,12,21). This indicates that the firewall rule is to be applied on all but the specified days.
startdate *date*	Specifies the start of a period of time in which the firewall rule will be applied. Set the date, and optionally the time, using one of the following formats: **yyyy-mm-dd** (e.g. 2009-03-12) **yyyy-mm-ddThh:mm:ss** (e.g. 2009-03-12T17:30:00) The default is 1970-01-01. Note that if the time is specified that it is 24 hour format (valid values are from 00:00:00 to 23:59:59). If the time is not specified the default is the start of the day on the specified date (i.e. 00:00:00). Use **stopdate** to end the activation period.
starttime *time*	Specifies the start of a period of time in the day to which the firewall rule will be applied. Set the start time using the following format: **hh:mm:ss** (e.g. 17:30:00) Note that the time is specified in 24 hour format (valid values are from 00:00:00 to 23:59:59). Use **stoptime** to end the activation period.
stopdate *date*	Specifies the end of a period of time in which the firewall rule will be applied. Set the date, and optionally the time, using one of the following formats: **yyyy-mm-dd** (e.g. 2009-03-12) **yyyy-mm-ddThh:mm:ss** (e.g. 2009-03-12T17:30:00) The default is 2038-01-19. Note that if the time is specified that it is 24 hour format (valid values are from 00:00:00 to 23:59:59). If the time is not specified the default is the start of the day on the specified date (i.e. 00:00:00). Use **startdate** to begin the activation period.

stoptime *time*	Specifies the end of a period of time in the day to which the firewall rule will be applied. Set the stop time using the following format: **hh:mm:ss** (e.g. 17:30:00) Note that the time is specified in 24 hour format (valid values are from 00:00:00 to 23:59:59). Use **starttime** to begin the activation period.
utc	Specifies that times given using **startdate**, **stopdate**, **starttime**, and **stoptime**, should be interpreted as UTC time rather than local time.
weekdays *days-of-week*	Specifies which days in the week the rule will be applied. Supported values are days of the week (Mon, Tue, Wed, Thu, Fri, Sat, and Sun) within a comma-separated list (e.g. Mon,Wed,Fri). The "!" character can be used to negate a list of values (e.g. !Mon,Wed,Fri). This indicates that the firewall rule is to be applied on all but the specified days of the week.

Default

The rule is applied at all times.

Usage Guidelines

Use this command to restrict the times during which the rule will be applied. All values are optional and are ANDed when specified.

Use the **set** form of this command to specify the times at which an IPv6 firewall rule will be applied.

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

Use the **show** form of this command to view time configuration for a firewall rule.

firewall ipv6-receive-redirects <state>

Specifies whether to process received ICMPv6 redirect messages.

Syntax

set firewall ipv6-receive-redirects {**enable** | **disable**}

delete firewall ipv6-receive-redirects

show firewall ipv6-receive-redirects

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   ipv6-receive-redirects state
}
```

Parameters

enable	Process received ICMPv6 redirect messages.
disable	Does not process received ICMPv6 redirect messages.

Default

The default is **disable**.

Usage Guidelines

Use this command to specify whether to process received ICMPv6 redirect messages.

Use the **set** form of this command to specify whether to process received ICMPv6 redirect messages.

Use the **delete** form of this command to remove the specified value.

Use the **show** form of this command to view the specified value.

firewall ipv6-src-route <state>

Specifies whether to process IPv6 packets with routing extension header.

Syntax

set firewall ipv6-src-route {**enable** | **disable**}

delete firewall ipv6-src-route

show firewall ipv6-src-route

Command Mode

Configuration mode.

Configuration Statement

```
firewall {
   ipv6-src-route state
}
```

Parameters

enable	Process IPv6 packets with routing header type 2.
disable	Does not process IPv6 packets with routing header.

Default

The default is **disable**.

Usage Guidelines

Source routing allows applications to override the routing tables and specify one or more intermediate destinations for outgoing datagrams. This capability is sometimes used for troubleshooting, but renders the network vulnerable to attacks where network traffic is transparently directed to a centralized collection point for packet capture.

Use this command to specify whether to process IPv6 packets with routing extension header.

Use the **set** form of this command to specify whether to process IPv6 packets with routing extension header.

Use the **delete** form of this command to remove the specified value.

Use the **show** form of this command to view the specified value.

interfaces <interface> firewall <direction> ipv6-name <fw-name>

Applies an IPv6 firewall instance to the defined interface.

Syntax

set interfaces *interface* **firewall** {**in ipv6-name** *fw-name* | **local ipv6-name** *fw-name* | **out ipv6-name** *fw-name*}

delete interfaces *interface* **firewall** [**in ipv6-name** | **local ipv6-name** | **out ipv6-name**]

show interfaces *interface* **firewall** [**in ipv6-name** | **local ipv6-name** | **out ipv6-name**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces interface {
   firewall {
      in {
         ipv6-name fw-name
      }
      local {
         ipv6-name fw-name
      }
      out {
         ipv6-name fw-name
      }
   }
}
```

Parameters

interface	Mandatory. The type of interface. For detailed keywords and arguments that can be specified as interface types, see the table in the Usage Guidelines below.
in ipv6-name *fw-name*	Applies the specified IPv6 firewall instance to inbound traffic on the specified interface.

local ipv6-name *fw-name*	Applies the specified IPv6 firewall instance to traffic arriving on the specified interface and bound for the local system.
out ipv6-name *fw-name*	Applies the specified IPv6 firewall instance to outbound traffic on the specified interface.

Default

None.

Usage Guidelines

Use this command to apply an IPv6 firewall instance, or rule set, to an interface.

A firewall has no effect on traffic traversing the system or destined to the system until a firewall rule set has been applied to an interface or a vif using this command.

To use the firewall feature, you define a firewall rule set as a named firewall instance, using firewall command. You then apply the firewall instance to interfaces and/or vifs using a statement like this one. Once applied, the instance acts as a packet filter.

The firewall instance will filter packets in one of the following ways, depending on what you specify when you apply it:

- **in.** If you apply the rule set as **in**, the firewall will filter packets entering the interface.
- **out.** If you apply the rule set as **out**, the firewall will filter packets leaving the interface.
- **local.** If you apply the rule set as **local**, the firewall will filter packets destined for the system itself.

For each interface, you can apply up to three firewall instances: one firewall **in** instance, one firewall **out** instance, and one firewall **local** instance.

Make sure the firewall instance you apply to an interface is already defined, or you may experience unintended results. If you apply a firewall instance that does not exist to an interface, the implicit firewall rule of **allow all** will be applied.

The following table shows the syntax and parameters for supported interface types.

Interface Type	Syntax	Parameters
ADSL Bridged Ethernet	adsl *adslx* pvc *pvc-id* bridged-ethernet	*adslx* The name of a Bridged Ethernet- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.
ADSL Classical IPOA	adsl *adslx* pvc *pvc-id* classical-ipoa	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.
ADSL PPPoA	adsl *adslx* pvc *pvc-id* pppoa *num*	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically. *num* The PPPoA unit number. This number must be unique across all PPPoA interfaces. In addition, only one PPPoA instance can be configured on a PVC. PPPoA units range from 0 to 15 and the resulting interfaces are named pppoa0 to pppoa15.
ADSL PPPoE	adsl *adslx* pvc *pvc-id* pppoe *num*	*adslx* The name of a Classical IPoA- encapsulated DSL interface. *pvc-id* The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Bonding	bonding *bondx*	*bondx* The identifier for the bonding interface. Supported values are **bond0** through **bond99**.
Bonding Vif	bonding *bondx* vif *vlan-id*	*bondx* The identifier for the bonding interface. Supported values are **bond0** through **bond99**. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094.

Interface Type	Syntax	Parameters
Bridge	bridge *brx*	*brx* The name of a Bridge group. The range is **br0** through **br999**.
Ethernet	ethernet *ethx*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system.
Ethernet PPPoE	ethernet *ethx* pppoe *num*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Ethernet Vif	ethernet *ethx* vif *vlan-id*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094.
Ethernet Vif PPPoE	ethernet *ethx* vif *vlan-id* pppoe *num*	*ethx* The name of an Ethernet interface. The range is **eth0** through **eth23**, depending on the physical interfaces available on your system. *vlan-id* The VLAN ID for the vif. The range is 0 to 4094. *num* The name of a defined PPPoE unit. The range is 0 to 15.
Loopback	loopback *lo*	*lo* The name of the loopback interface.
Multilink	multilink *mlx* vif *1*	*mlx* The identifier of the multilink bundle. You can create up to two multilink bundles. Supported values are **ml0** (“em ell zero”) through **ml23** (“em ell twenty-three”). *1* The identifier of the virtual interface. Currently, only one vif is supported for multilink interfaces, and the identifier must be 1. The vif must already have been defined.
OpenVPN	openvpn *vtunx*	*vtunx* The identifier for the OpenVPN interface. This may be **vtun0** to **vtun*x***, where ***x*** is a non-negative integer.
Pseudo-Ethernet	pseudo-ethernet *pethx*	*pethx* The name of a pseudo-Ethernet interface. The range is **peth0** through **peth999**.
Serial Cisco HDLC	serial *wanx* cisco-hdlc vif *1*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *1* The identifier of the virtual interface. Currently, only one vif is supported for Cisco HDLC interfaces, and the identifier must be 1. The vif must already have been defined.

Interface Type	Syntax	Parameters
Serial Frame Relay	serial *wanx* frame-relay vif *dlci*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *dlci* The identifier of the virtual interface. For Frame Relay interfaces, this is the DLCI number for the interface. the range is 16 to 991. The vif must already have been defined.
Serial PPP	serial *wanx* ppp vif *1*	*wanx* The serial interface you are configuring: one of **wan0** through **wan23**. The interface must already have been defined. *1* The identifier of the virtual interface. Currently, only one vif is supported for point-to-point interfaces, and the identifier must be 1. The vif must already have been defined.
Tunnel	tunnel *tunx*	*tunx* An identifier for the tunnel interface you are defining. This may be **tun0** to **tun*x***, where ***x*** is a non-negative integer.
Virtual Tunnel	vti *vtix*	*vtix* An identifier for the virtual tunnel interface you are defining. This may be **vti0** to **vti*x***, where ***x*** is a non-negative integer. **Note:** This interface does not support IPv6.
VRRP	interface *parent-if* vrrp vrrp-group *group* interface	*parent-if* The type and identifier of the parent interface; for example, **ethernet eth0** or **bonding bond0**. *group* The VRRP group identifier. The name of the VRRP interface is not specified. The system internally constructs the interface name from the parent interface identifier plus the VRRP group number—for example, **eth0v99**, **eth0.15v99**, **bond0v99**, ot **bond0.15v99**. Note that VRRP interfaces support the same feature set as the parent interface does.
Wireless	wireless *wlanx*	*wlanx* The identifier for the wireless interface you are using. This may be **wlan0** to **wlan999**.
Wireless Modem	wirelessmodem *wlmx*	*wlmx* The identifier for the wireless modem interface you are using. This may be **wlm0** to **wlm999**.

Use the **set** form of this command to apply an IPv6 firewall instance to an interface.

Use the **delete** form of this command to remove an IPv6 firewall instance from an interface.

Use the **show** form of this command to view an IPv6 firewall configuration for an interface.

Chapter 6: Zone-Based Firewall Commands

This chapter describes commands for implementing zone-based firewall on the Vyatta system.

Zone-Based Firewall Commands

Configuration Commands	
zone-policy zone <to-zone>	Defines a security zone.
zone-policy zone <to-zone> default-action <action>	Defines the default action for traffic arriving at a security zone.
zone-policy zone <to-zone> description <desc>	Specifies a description for a security zone.
zone-policy zone <to-zone> from <from-zone>	Names the traffic source zone to which this policy applies.
zone-policy zone <to-zone> from <from-zone> firewall ipv6-name <name>	Applies packet filtering as defined in an IPv6 firewall rule set to traffic arriving from the specified “from” zone.
zone-policy zone <to-zone> from <from-zone> firewall name <name>	Applies packet filtering as defined in an IPv4 firewall rule set to traffic arriving from the specified “from” zone.
zone-policy zone <to-zone> interface <if-name>	Adds an interface to a security zone.
zone-policy zone <to-zone> local-zone	Designates this zone as the “local” zone.
Operational Commands	
show zone-policy	Displays security zone policy information.

show zone-policy

Displays security zone policy information.

Syntax

show zone-policy [**zone** *zone-name*]

Command Mode

Operational mode.

Parameters

zone-name	The name of a specific security zone.

Default

Security zone policies for all security zones are displayed.

Usage Guidelines

Use this command to display security zone policies.

Examples

Example 6-1 shows security zone policies for all security zones on R1.

Example 6-1 "show zone-policy": Displaying information for all security zones

```
vyatta@R1:~$ show zone-policy
-------------------
Name: inside [DESC]

Interfaces: peth0 peth1 peth2 peth3

From Zone:
  Name              firewall               content-inspection
  ----              ---------              ------------------
  local             local-to-inside        disabled
                    local-to-inside-6 [v6]
  outside           outside-to-inside      enabled
```

```
-------------------
Name: local [DESC]

Interfaces: local-zone

From Zone:
  Name              firewall               content-inspection
  ----              ---------              ------------------
  inside            inside-to-local        disabled
  outside           outside-to-local       enabled

-------------------
Name: outside [DESC]

Interfaces: eth0

From Zone:
  Name              firewall               content-inspection
  ----              ---------              ------------------
  inside            inside-to-outside      disabled
  local             local-to-outside       disabled
```

Example 6-2 shows security zone policies for a specific security zone (inside) on R1.

Example 6-2 "show zone-policy zone inside": Displaying information for a specific security zone

```
vyatta@R1:~$ show zone-policy zone inside
-------------------
Name: inside [DESC]

Interfaces: peth0 peth1 peth2 peth3

From Zone:
  Name              firewall               content-inspection
  ----              ---------              ------------------
  local             local-to-inside        disabled
                    local-to-inside-6 [v6]
  outside           outside-to-inside      enabled
```

zone-policy zone <to-zone>

Defines a security zone.

Syntax

set zone-policy zone *zone*

delete zone-policy zone *zone*

show zone-policy *zone*

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone zone {}
```

Parameters

zone	Multimode. The name of the security zone. The name can be up to 18 characters long. You can define multiple security zones by creating more than one **zone-policy zone** configuration node.

Default

None.

Usage Guidelines

Use this command to define a security zone.

In the Vyatta system, a zone is defined as a group of interfaces with the same security level. Once the zone is defined, a filtering policy can be applied to traffic flowing between zones.

By default, traffic to a zone is dropped unless a policy has been defined for the zone sending the traffic. Traffic flowing within a zone is not filtered.

When defining zones, keep the following in mind:

- An interface can be a member of only one zone.

- An interface that is a member of a zone cannot have a firewall rule set directly applied to it.
- For interfaces not assigned to a zone, traffic is unfiltered by default. These interfaces can have rule sets directly applied to them.

Use the **set** form of this command to define a security zone.

Use the **delete** form of this command to remove a security zone.

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

zone-policy zone <to-zone> default-action <action>

Defines the default action for traffic arriving at a security zone.

Syntax

set zone-policy zone *to-zone* **default-action** *action*

delete zone-policy zone *to-zone* **default-action**

show zone-policy zone *to-zone* **default-action**

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   default-action action
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
action	The action to be taken for traffic arriving at a security zone. Supported values are as follows: **drop**: Traffic is silently dropped. **reject**: Traffic is dropped with an ICMP "Destination Unreachable" message.

Default

Traffic is silently dropped.

Usage Guidelines

Use this command to specify the default action to take for traffic arriving at a security zone.

This is the action that will be taken for all traffic arriving from zones for which a policy has not been defined. That is, in order for traffic from a given zone to be allowed, a policy must be explicitly defined allowing traffic from that zone.

Use the **set** form of this command to set the default action.

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

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

zone-policy zone <to-zone> description <desc>

Specifies a description for a security zone.

Syntax

set zone-policy zone *to-zone* **description** *desc*

delete zone-policy zone *to-zone* **description**

show zone-policy zone *to-zone* **description**

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   description desc
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
descr	A string providing a brief description for the security zone. If the string contains spaces, it must be enclosed in double quotes.

Default

None.

Usage Guidelines

Use this command to record a brief description for security zone.

Use the **set** form of this command to specify the description.

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

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

zone-policy zone <to-zone> from <from-zone>

Names the traffic source zone to which this policy applies.

Syntax

set zone-policy zone *to-zone* **from** *from-zone*

delete zone-policy zone *to-zone* **from** *from-zone*

show zone-policy zone *to-zone* **from** *from-zone*

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   from from-zone
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
from-zone	The name of the security zone from which traffic is originating.

Default

None.

Usage Guidelines

Use this command to specify a zone from which traffic will be arriving (the "from" zone). The packet filtering policy for this "from" zone is applied to all traffic arriving from this zone.

Use the **set** form of this command to specify the description.

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

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

zone-policy zone <to-zone> from <from-zone> firewall ipv6-name <name>

Applies packet filtering as defined in an IPv6 firewall rule set to traffic arriving from the specified “from” zone.

Syntax

set zone-policy zone *to-zone* **from** *from-zone* **firewall ipv6-name** *name*

delete zone-policy zone *to-zone* **from** *from-zone* **firewall ipv6-name**

show zone-policy zone *to-zone* **from** *from-zone* **firewall ipv6-name**

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   from from-zone
      firewall {
         ipv6-name name
      }
   }
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
from-zone	The name of the security zone from which traffic is originating.
name	The name of an IPv6 firewall rule set.

Default

None.

Usage Guidelines

Use this command to apply an IP version 6 (IPv6) rule set as a packet filter to traffic arriving from a “from” zone.

You can apply one IPv6 rule set and one IP version 4 (IPv4) rule set as packet filters for a “from” zone.

Use the **set** form of this command to specify an IPv6 rule set as a packet filter for a “from” zone.

Use the **delete** form of this command to remove IPv6 rule set from the packet filters defined for a “from” zone.

Use the **show** form of this command to see what packet filter, if any, has been applied to a “from” zone.

zone-policy zone <to-zone> from <from-zone> firewall name <name>

Applies packet filtering as defined in an IPv4 firewall rule set to traffic arriving from the specified “from” zone.

Syntax

set zone-policy zone *to-zone* **from** *from-zone* **firewall name** *name*

delete zone-policy zone *to-zone* **from** *from-zone* **firewall name**

show zone-policy zone *to-zone* **from** *from-zone* **firewall name**

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   from from-zone
      firewall {
         name name
      }
   }
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
from-zone	The name of the security zone from which traffic is originating.
name	The name of an IPv4 firewall rule set.

Default

None.

Usage Guidelines

Use this command to apply an IP version 4 (IPv4) rule set as a packet filter to traffic arriving from a “from” zone.

You can apply one IPv4 rule set and one IP version 6 (IPv6) rule set as packet filters for a “from” zone.

Use the **set** form of this command to specify an IPv4 rule set as a packet filter for a “from” zone.

Use the **delete** form of this command to remove an IPv4 rule set from the packet filters defined for a “from” zone.

Use the **show** form of this command to see what IPv4 packet filter, if any, has been applied to a “from” zone.

zone-policy zone <to-zone> interface <if-name>

Adds an interface to a security zone.

Syntax

set zone-policy zone *to-zone* **interface** *if-name*

delete zone-policy zone *to-zone* **interface** *if-name*

show zone-policy zone *to-zone* **interface** *if-name*

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   interface if-name {}
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.
if-name	Multinode. The name of an interface; for example, **eth0**, **wan1**, or **ppp1**.

Default

None.

Usage Guidelines

Use this command to add an interface to a security zone.

All interfaces in the zone have the same security level; traffic arriving to those interfaces from other zones is all treated in the same way. Traffic flowing between interfaces in the same security zone is not filtered.

Use the **set** form of this command to add an interface to the zone.

Use the **delete** form of this command to remove an interface from the zone.

Use the **show** form of this command to see what interfaces are members of this zone.

zone-policy zone <to-zone> local-zone

Designates this zone as the “local” zone.

Syntax

set zone-policy zone *to-zone* **local-zone**

delete zone-policy zone *to-zone* **local-zone**

show zone-policy zone *to-zone*

Command Mode

Configuration mode.

Configuration Statement

```
zone-policy zone to-zone {
   local-zone
}
```

Parameters

to-zone	The name of the security zone that traffic is destined for.

Default

None.

Usage Guidelines

Use this command to designate security zone as the “local” zone.

The local zone is a special zone which refers to the local Vyatta device itself. If you specify a security zone as local, the firewall policies specified for the zone filter packets destined for the Vyatta system.

By default, all traffic destined for the router and originating from the router is allowed.

Only one zone may be designated as the local zone.

Use the **set** form of this command to designate a security zone as the local zone.

Use the **delete** form of this command to stop a security zone from being the local zone.

Use the **show** form of this command to see security zone configuration.

Appendix A: ICMP Types

This appendix lists the ICMP types defined by the Internet Assigned Numbers Authority (IANA).

The Internet Assigned Numbers Authority (IANA) has developed a standard that maps a set of integers onto ICMP types.Table A-1 lists the ICMP types and codes defined by the IANA and maps them to the strings literal strings available in the Vyatta system.

Table A-1 ICMP types

ICMP Type	Code	Literal	Description
0 - Echo reply	0	echo-reply	Echo reply (pong)
3 - Destination unreachable		destination- unreachable	
	0	network-unreachable	Destination network unreachable
	1	host-unreachable	Destination host unreachable
	2	protocol-unreachable	Destination protocol unreachable
	3	port-unreachable	Destination port unreachable
	4	fragmentation-needed	Fragmentation required
	5	source-route-failed	Source route failed
	6	network-unknown	Destination network unknown
	7	host-unknown	Destination host unknown
	9	network-prohibited	Network administratively prohibited
	10	host-prohibited	Host administratively prohibited
	11	TOS-network-unreachable	Network unreachable for TOS
	12	TOS-host-unreachable	Host unreachable for TOS
	13	communication-prohibited	Communication administratively prohibited
	14	host-precedence-violation	Requested precedence is not permitted.

Table A-1 ICMP types

ICMP Type	Code	Literal	Description
	15	precedence-cutoff	Datagram sent with precedence lower than required minimum.
4 - Source quench	0	source-quench	Source quench (congestion control)
5 - Redirect message		redirect	
	0	network-redirect	Redirect datagrams for the network
	1	host-redirect	Redirect datagrams for the host
	2	TOS-network-redirect	Redirect datagrams for the TOS and network
	3	TOS-host-redirect	Redirect datagrams for the TOS and host
8 - Echo request	0	echo-request	Echo request (ping)
9 - Router advertisement	0	router-advertisement	Router advertisement
10 - Router solicitation	0	router-solicitation	Router solicitation
11 - Time exceeded		time-exceeded	
	0	ttl-zero-during-transit	TTL expired in transit
	1	ttl-zero-during-reassembly	Fragment reassembly time exceeded
12 - Parameter problem: Bad IP header		parameter-problem	
	0	ip-header-bad	Pointer indicates the error
	1	required-option-missing	Missing required option
13 - Timestamp	0	timestamp-request	Timestamp
14 - Timestamp reply	0	timestamp-reply	Timestamp reply
15 - Information request	0		Information request

Table A-1 ICMP types

ICMP Type	Code	Literal	Description
16 - Information reply	0		Information reply
17 - Address mask request	0	address-mask-request	Address mask request
18 - Address mask reply	0	address-mask-reply	Address mask reply

Appendix B: ICMPv6 Types

This appendix lists the ICMPv6 types defined by the Internet Assigned Numbers Authority (IANA).

The Internet Assigned Numbers Authority (IANA) has developed a standard that maps a set of integers onto ICMPv6 types. Table B-2 lists the ICMPv6 types and codes defined by the IANA and maps them to the strings literal strings available in the Vyatta system.

Table B-1 ICMPv6 types

ICMPv6 Type	Code	Literal	Description
1 - Destination unreachable		destination- unreachable	
	0	no-route	No route to destination
	1	communication-prohibited	Communication with destination administratively prohibited
	2		Beyond scope of source address
	3	address-unreachable	Address unreachable
	4	port-unreachable	Port unreachable
	5		Source address failed ingress/egress policy
	6		Reject route to destination
2 - Packet too big	0	packet-too-big	
3 - Time exceeded		time-exceeded	
	0	ttl-zero-during-transit	Hop limit exceeded in transit
	1	ttl-zero-during-reassembly	Fragment reassembly time exceeded
4 - Parameter problem		parameter-problem	
	0	bad-header	Erroneous header field encountered
	1	unknown-header-type	Unrecognized Next Header type encountered
	2	unknown-option	Unrecognized IPv6 option encountered

Table B-1 ICMPv6 types

ICMPv6 Type	Code	Literal	Description
128 - Echo request	0	echo-request (ping)	Echo request
129 - Echo reply	0	echo-reply (pong)	Echo reply
133 - Router solicitation	0	router-solicitation	Router solicitation
134 - Router advertisement	0	router-advertisement	Router advertisement
135 - Neighbor solicitation	0	neighbor-solicitation (neighbour-solicitation)	Neighbor solicitation
136 - Neighbor advertisement	0	neighbor-advertisement (neighbour-advertisement)	Neighbor advertisement

The Internet Assigned Numbers Authority (IANA) has developed a standard that maps a set of integers onto ICMP types.Table B-2 lists the ICMP types and codes defined by the IANA and maps them to the strings literal strings available in the Vyatta system.

Table B-2 ICMP types

ICMP Type	Code	Literal	Description
0 - Echo reply	0	echo-reply	Echo reply (pong)
3 - Destination unreachable		destination- unreachable	
	0	network-unreachable	Destination network unreachable
	1	host-unreachable	Destination host unreachable
	2	protocol-unreachable	Destination protocol unreachable
	3	port-unreachable	Destination port unreachable
	4	fragmentation-needed	Fragmentation required
	5	source-route-failed	Source route failed

Table B-2 ICMP types

ICMP Type	Code	Literal	Description
	6	network-unknown	Destination network unknown
	7	host-unknown	Destination host unknown
	9	network-prohibited	Network administratively prohibited
	10	host-prohibited	Host administratively prohibited
	11	TOS-network-unreachable	Network unreachable for TOS
	12	TOS-host-unreachable	Host unreachable for TOS
	13	communication-prohibited	Communication administratively prohibited
	14	host-precedence-violation	Requested precedence is not permitted.
	15	precedence-cutoff	Datagram sent with precedence lower than required minimum.
4 - Source quench	0	source-quench	Source quench (congestion control)
5 - Redirect message		redirect	
	0	network-redirect	Redirect datagrams for the network
	1	host-redirect	Redirect datagrams for the host
	2	TOS-network-redirect	Redirect datagrams for the TOS and network
	3	TOS-host-redirect	Redirect datagrams for the TOS and host
8 - Echo request	0	echo-request	Echo request (ping)
9 - Router advertisement	0	router-advertisement	Router advertisement

Table B-2 ICMP types

ICMP Type	Code	Literal	Description
10 - Router solicitation	0	router-solicitation	Router solicitation
11 - Time exceeded		time-exceeded	
	0	ttl-zero-during-transit	TTL expired in transit
	1	ttl-zero-during-reassembly	Fragment reassembly time exceeded
12 - Parameter problem: Bad IP header		parameter-problem	
	0	ip-header-bad	Pointer indicates the error
	1	required-option-missing	Missing required option
13 - Timestamp	0	timestamp-request	Timestamp
14 - Timestamp reply	0	timestamp-reply	Timestamp reply
15 - Information request	0		Information request
16 - Information reply	0		Information reply
17 - Address mask request	0	address-mask-request	Address mask request
18 - Address mask reply	0	address-mask-reply	Address mask reply

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
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
PTMU	Path Maximum Transfer Unit
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
TBF	Token Bucket Filter
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol

ToS	Type of Service
TSS	TCP Maximum Segment Size
Tx	transmit
UDP	User Datagram Protocol
VHD	virtual hard disk
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