JUNIPER NETWORKS

Junos® OS

Dynamic Firewall Feature Guide for Subscriber Services

Release
13.2

Published: 2013-07-31

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS Dynamic Firewall Feature Guide for Subscriber Services
13.2

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- MX Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see the *CLI User Guide*.

Documentation Conventions

Table 1 on page xv defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page xv defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the**configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
Italic text like this	• Introduces or emphasizes important new terms. • Identifies book names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS System Basics Configuration Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Enclose optional keywords or variables.	**stub <default-metric** ***metric*****>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Enclose a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identify a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at https://www.juniper.net/cgi-bin/docbugreport/ . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://www.juniper.net/alerts/

- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html.

PART 1

Overview

CHAPTER 1

Dynamic Firewall Services and Service Sets in Subscriber Access Networks

Dynamic Firewall Filters Overview

Firewall filters provide rules that define whether to accept or reject packets that are transiting an interface on a router. The subscriber management feature supports four categories of firewall filters—classic filters, parameterized filters, Ascend-Data-Filters, and fast update filters.

- Classic filters are compiled at commit time and then, when a service is activated, an interface-specific clone of the filter is created and attached to a logical interface. Classic filters are static filters; they cannot contain subscriber-specific terms (also called rules). Classic filters can be applied to interfaces dynamically. This dynamic application is performed by associating input or output filters with a dynamic profile. When triggered, a dynamic profile can apply a named filter or a filter specified in RADIUS to an interface.
- Parameterized filters add the ability to configure firewall filters under a dynamic profile. The filter definitions utilize dynamic-profile variables, which allow you to customize your configuration at session creation time. You can configure a general filter under a dynamic profile and then provide policing rates, destination addresses, ports, and so forth when a dynamic session is activated.
- Ascend-Data-Filters create policies for subscriber traffic. The filter is configured on the RADIUS server and contains rules that specifically match conditions for traffic and define an action for the router to perform.
- Fast update filters are similar to classic filters in many ways. However, fast update filters support subscriber-specific, rather than interface-specific, filter values. Fast update filters also allow individual filter terms to be incrementally added or removed from filters without requiring that the entire filter be recompiled for each modification. Fast update filters are essential for networking environments in which multiple subscribers might share the same logical interface.

You configure firewall filters to determine whether to accept or reject traffic before it enters or exits an interface to which the firewall filter is applied. An *input* (or *ingress*) firewall filter is applied to packets that are entering a network. An *output* (or *egress*) firewall filter is applied to packets that are exiting a network. You can configure firewall filters to subject packets to filtering or class-of-service (CoS) marking (grouping similar types of traffic together and treating each type of traffic as a class with its own level of service priority).

Related Documentation

- Classic Filters Overview on page 7
- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11
- Parameterized Filters Overview on page 19
- Fast Update Filters Overview on page 40
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53

- Dynamically Attaching Filters Using RADIUS Variables on page 55

Dynamic Service Sets Overview

A service set is a collection of services to be performed by an Adaptive Services (AS) or Multiservices PIC. You configure a service-set definition at the **[edit services]** hierarchy level. You can then apply the service set to one or more interfaces on the router. The service set can be applied either dynamically or statically.

To dynamically associate a service set to interfaces you include the **service-set** statement with the **input** or **output** statement at the **[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service]** hierarchy level.

To statically associate a defined service set with an interface, you include the **service-set** statement with the **input** or **output** statement at the **[edit interfaces *interface-name* unit *logical-unit-number* family *family* service]** hierarchy level.

Related Documentation

- Associating Service Sets with Interfaces in a Dynamic Profile on page 73
- Verifying and Managing Service Sets Information on page 192
- For information about creating service sets, see “Service Set Configuration Guidelines” in the *Junos OS Services Interfaces Library for Routing Devices*.
- For information about statically applying service sets to interfaces, see “Applying Filters and Services to Interfaces” in the *Junos OS Services Interfaces Library for Routing Devices*.

CHAPTER 2

Classic Filters

Classic Filters Overview

The dynamic firewall feature supports classic filters, parameterized filters, and fast update filters. Classic filters are compiled at commit time. When a service activation takes place, the router creates an interface-specific clone of the filter and attaches the clone to the specified logical interface. Classic filters are static filters; they cannot contain subscriber-specific terms, as opposed to fast update filters, which are subscriber-specific. Parameterized filters and policers have their configuration customized at session creation time.

This overview covers:

Classic Filter Types

The following classic filter types are supported:

- Port (Layer 2) firewall filter—Port firewall filters apply to Layer 2 switch ports. You can apply port firewall filters only in the ingress direction on a physical port.
- VLAN firewall filter—VLAN firewall filters provide access control for packets that enter a VLAN, are bridged within a VLAN, and leave a VLAN. You can apply VLAN firewall filters in both ingress and egress directions on a VLAN. VLAN firewall filters are applied to all packets that are forwarded to or forwarded from the VLAN.
- Router (Layer 3) firewall filter—You can apply a router firewall filter in both ingress and egress directions on Layer 3 (routed) interfaces.

Classic Filter Components

When creating a classic filter, you first define the family address type (**inet** or **inet6**) and then you define one or more terms that specify the filtering criteria and the action to take when a match occurs.

Each term, or rule, consists of the following components:

- Match conditions—Specifies values or fields that the packet must contain. You can define various match conditions, including:
  - IP source address field
  - IP destination address field
  - Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) source port field
  - IP protocol field
  - Internet Control Message Protocol (ICMP) packet type
  - TCP flags
  - interfaces
- Actions—Specifies what to do when a match condition occurs. Possible actions are to accept or discard a packet. In addition, packets can be counted to collect statistical information. If no action is specified for a term, the default action is to accept the packet.

Classic Filter Processing

The order of the terms within a classic filter is important. Packets are tested against each term in the order in which the terms are listed in the firewall filter configuration. When a firewall filter contains multiple terms, the router takes a top-down approach and compares a packet against the first term in the firewall filter. If the packet matches the first term, the router executes the action defined by that term to either accept or reject the packet, and no other terms are evaluated. If the router does not find a match between the packet and first term, it then compares the packet to the next term in the firewall filter by using the same match process. If no match occurs between the packet and the second term, the router continues to compare the packet to each successive term defined in the firewall filter until a match is found. If a packet does not match any terms in a firewall filter, the default action is to discard the packet.

You can also specify a precedence (from 0 through 255) for input and output filters within a dynamic profile to force filter processing in a particular order. Setting a lower precedence value for a filter gives it a higher precedence within the dynamic profile. Filters with lower precedence values are applied to interfaces before filters with higher precedence values. A precedence of zero (the default) gives the filter the highest precedence. If no precedence is specified, the filter receives a precedence of zero (highest precedence). Filters with matching precedence (zero or otherwise) are applied in random order.

NOTE: Dynamic filters do not process outbound packets that are sourced from the routing engine. To filter outbound packets that are sourced from the routing engine, you can create static outbound filters for each interface.

Guidelines for Creating and Applying Classic Filters for Subscriber Interfaces

This release supports the dynamic configuration of firewall filters. However, you can also continue to create static firewall filters for interfaces as you do normally, and then dynamically apply those filters to statically created interfaces using dynamic profiles. You can also use dynamic profiles to attach input and output filters through RADIUS.

When creating and applying filters, keep the following in mind:

- This release supports dynamic application of only input and output filters.
- The filters must be interface-specific.
- You can create family-specific **inet** and **inet6** filters.
- You can create interface-specific filters at the **unit** level that apply to any family type (**inet** or **inet6**) configured on the interface.
- You can add or remove both IPv4 and IPv6 filters with the same service activation or deactivation.
- You can remove one filter type without impacting the other type of filter. For example, you can remove IPv6 filters and leave the current IPv4 filters active.
- You can chain up to five input filters and four output filters together.
- If you do not configure and apply a filter, the interface uses the default group filter configuration.
- You cannot modify or delete a firewall filter while subscribers on the same logical interface are bound.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Fast Update Filters Overview on page 40
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- Dynamically Attaching Filters Using RADIUS Variables on page 55
- Verifying and Managing Firewall Filter Configuration on page 191

Basic Classic Filter Syntax

This section provides the basic classic filter CLI statement syntax. The first part of this syntax provides the CLI statements to associate an input and output filter with a dynamic profile. The second part of this syntax represents the configured input and output filters applied to the dynamic profile. When a DHCP event occurs, the dynamic profile applies the specified filters to the DHCP client interface on the router.

```
[edit]
dynamic-profiles [profile-name] {
  interfaces {
    [$junos-interface-ifd-name] {
      unit [$junos-underlying-interface-unit] {
        family  family] {
          filter {
            input {
              [filter-name];
              precedence [precedence];
            }
            output {
              [filter-name];
              precedence [precedence];
            }
          }
        }
      }
    }
  }
[edit]
firewall {
  family [family] {
    filter [filter-name] {
      [desired filter configuration]
    }
    filter [filter-name] {
      [desired filter configuration]
    }
  }
}
```

Related Documentation

- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- Dynamic Firewall Filters Overview on page 4

Ascend-Data-Filter Policies for Subscriber Management Overview

Subscriber management enables you to use Ascend-Data-Filters to create policies for subscriber traffic. An Ascend-Data-Filter is a binary value that is configured on the RADIUS server. The filter contains rules that specify match conditions for traffic and an action for the router to perform (such as accept or discard the traffic). The match conditions might include the source and destination IP address or port, the protocol, the filter direction, the traffic class, and policer information.

Subscriber management uses a dynamic profile to obtain the Ascend-Data-Filter attribute (RADIUS attribute 242) from the RADIUS server and apply the policy to a subscriber session. Dynamic profiles support Ascend-Data-Filters for **inet** and **inet6** family types, and both families can be present in a dynamic profile. You include Junos OS predefined variables in the dynamic profiles — **$junos-adf-rule-v4** for family **inet** and **$junos-adf-rule-v6** for **inet6**. The Ascend-Data-Filter attribute can include rules for both address families. The predefined variables map the Ascend-Data-Filter rules for the respective family to the Junos OS firewall filter process. A firewall filter is created and attached to the subscriber's logical interface.

You can also configure a static Ascend-Data-Filter by manually entering the required binary data as a hexadecimal string in a dynamic profile. A statically configured Ascend-Data-Filter in a dynamic profile takes precedence over an Ascend-Data-Filter attribute that is received from RADIUS. The static method is time-consuming to configure; it is typically used only for testing purposes.

The Ascend-Data-Filter attribute is supported in RADIUS Access-Accept and Change of Authorization (CoA) messages.

CoA updates existing filters based on the Ascend-Data-Filter Type field, as shown in the following list:

- If the Type field is 1, IPv4 rules are updated and IPv6 rules are unchanged. The opposite is true if the Type field is 3.
- If both Type 1 and 3 are specified, then all rules are updated.
- If the CoA has no Ascend-Data-Filter rules, then the existing rules are unchanged.

Filter Naming Conventions

Each Ascend-Data-Filter has a unique name, which is assigned by the dynamic firewall process, dfwd. The assigned names are displayed in the results of the **show subscriber extensive** and **show firewall** commands. Ascend-Data-Filters use the following naming convention:

__junos_adf_*session#-interfacename-family-direction*

For example:

__junos_adf_33847-ge/1/0/4.53-init-in

Each Ascend-Data-Filter rule maps to a single term, and the term names are simply **t0**, **t1**, ..., **t*n***. If you configure the **counter** option, the router adds a count action to each term that is created. The counter names are a combination of the the term names with **-cnt** appended. For example **t0-cnt** and **t1-cnt**.

Use of Multiple Sessions with Ascend-Data-Filters on an Interface

An interface can have multiple subscriber sessions, each session using its own Ascend-Data-Filter rules. When an Ascend-Data-Filter is applied to a subscriber session, the rules are created independently of any other filters and are added to the interface filter list. The Ascend-Data-Filter rules for the other sessions on the same interface are also added to the filter list. All packets that are processed for the interface must go through all filters, and the filters are applied according to the precedence you set.

Because the filter list can be a combination of several rules, you must consider how the multiple filters coexist. You must ensure that the filters are designed and applied correctly in order to provide the desired filtering and resulting action. For example, a session might have a filter that accepts traffic from Subscriber-A and discards all other traffic. However, a second session on the same interface might have a filter that accepts traffic from Subscriber-B only and discards other traffic. When the two filters are combined in the filter list, traffic from Subscriber-B is discarded by the first filter, and traffic from Subscriber-A is discarded by the second filter. As a result, no traffic is accepted on the interface because the two filters essentially cancel out each other and discard all traffic.

Optional ADF Filter Requirement for Some Subscribers

When you include either of the predefined variables—**$junos-adf-rule-v4** or **$junos-adf-rule-v6**—in the dynamic profile, by default the RADIUS reply message must include the Ascend-Data-Filter attribute (RADIUS attribute 242) for each subscriber. If the attribute is not included, the router reports an error.

A service provider might apply the same dynamic profile to a mixed pool of subscribers, such that the attribute is included by RADIUS for some of the subscribers and is not included for others. By default, the router returns an error for each of the subscribers without the attribute, consuming system resources. You can configure the dynamic profile to accommodate such a mixture of subscribers by making the attribute requirement optional. To do so, and to suppress attribute error reporting, specify the **not-mandatory** option with the **adf** statement at the **[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter]** hierarchy level. With this configuration, the Ascend-Data-filter is simply not created when the Ascend-Data-Filter attribute is not present.

Related Documentation

- Dynamically Applying Ascend-Data-Filter Policies to Subscriber Sessions on page 61
- Ascend-Data-Filter Attribute Fields on page 12

Ascend-Data-Filter Attribute Fields

Table 3 on page 13 provides information about the fields used in the Ascend-Data-Filter attribute (RADIUS attribute 242) and how the fields map to Junos OS filter functions. The table lists the fields in the order in which they occur in the Ascend-Data-Filter attribute.

Table 3: Ascend-Data-Filter Attribute Fields

Action or Classifier	Format	Value	Junos OS Filter Function
Type	1 byte	• 1 = IPv4 • 3 = IPv6	
Filter or forward	1 byte	• 0 = filter • 1 = forward	• 0 = maps to **discard** action • 1 = maps to **accept** action
Indirection	1 byte	• 0 = egress • 1 = ingress	• 0 = adds egress terms to the output filter • 1= adds ingress terms to the input filter
Spare	1 byte	–	–
Source IP address	IPv4 = 4 bytes IPv6 = 16 bytes	IP address of the source interface	• 0 = no mapping performed • From **source-address** ***address*** entry added to term
Destination IP address	IPv4 = 4 bytes IPv6 = 16 bytes	IP address of the destination interface	• 0 = no mapping performed • From **destination-address** ***address*** entry added to term
Source IP prefix	1 byte	• Type 1 = Number of leading zeros in the wildcard mask • Type 3 = Higher order contiguous bits of the address that make up the network portion of the address	• 0 = no mapping performed • From **source-address** ***prefix*** entry added to term
Destination IP prefix	1 byte	• Type 1 = Number of leading zeros in the wildcard mask • Type 3 = Higher order contiguous bits of the address that make up the network portion of the address	• 0 = no mapping performed • From **destination-address** ***prefix*** entry added to term
Protocol	1 byte	Protocol type	• 0 = no mapping performed • IPv4 = from **protocol** ***number*** added to term • IPv6 = from **next-header** ***number*** added to term

Table 3: Ascend-Data-Filter Attribute Fields *(continued)*

Action or Classifier	Format	Value	Junos OS Filter Function
Established	1 byte	Not implemented	Not implemented
Source port	2 bytes	Port number of the source port	From **source-port** *x* **-** *y* entry added to term
Destination port	2 bytes	Port number of the destination port	From **destination-port** *x* **-** *y* entry added to term
Source port qualifier	1 byte	• 0 = no compare • 1 = less than • 2 = equal to • 3 = greater than • 4 = not equal to	• 0 = no mapping performed • 1 – 3 = mapped to corresponding option • 4 = mapped to **except** match option
Destination port qualifier	1 byte	• 0 = no compare • 1 = less than • 2 = equal to • 3 = greater than • 4 = not equal to	• 0 = no mapping performed • 1 – 3 = mapped to corresponding match option • 4 = mapped to **except** match option
Reserved	2 bytes	Not used	Not used
Marking value	1 byte	• For IPv4 = Type of Service (ToS) • For IPv6 = Differentiated Services Code Point (DSCP)	Not implemented
Marking mask	1 byte	0 = no packet marking	Not implemented

Table 3: Ascend-Data-Filter Attribute Fields *(continued)*

Action or Classifier	Format	Value	Junos OS Filter Function
Traffic class	1–41 bytes	• 0 = no traffic class (required if there is no profile) • First byte specifies the length of the ASCII name of the traffic class • Traffic class must be statically configured • Name can optionally be null terminated, which consumes 1 byte • If a name is given, it must match one of the default forwarding classes (such as best-effort) or the name of a forwarding class configured under the **[edit class-of-service scheduler-maps *map-name*]** stanza.	Maps to the forwarding class name. The action **forwarding-class *name*** is added to term.
Rate-limit profile	1–41 bytes	• 0 = no rate limit (required if there is no profile) • First byte specifies the length of the ASCII, followed by the ASCII name of the profile • Profile must be statically configured • Name can optionally be null terminated, which consumes 1 byte • If a name is given, it must match the name of one of the firewall policers that is configured under the **[edit firewall]** stanza.	Maps to the **policer *policer-name*** action modifier of the same name. The action **policer *name*** is added to term.

Related Documentation

- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11

Firewall Filters and Enhanced Network Services Mode Overview

Under normal conditions, every firewall filter is generated in two different formats -- compiled and term-based. The compiled format is used by the routing engine (RE) kernel, FPCs, and MS-DPs. The term-based format is used by MPCs. Compiled firewall filters are duplicated for each interface or logical interface to which they are applied. Term-based filters, instead of being duplicated, are referenced by each interface or logical interface.

When a combination of MPCs and any other cards populate a chassis, the creation of both firewall filter file formats is necessary. In most networks, the creation of both filter formats and any amount of duplication for compiled firewall filters has no effect on the router. However, in subscriber management networks that include thousands of statically configured subscriber interfaces, creating filters in multiple formats and duplicating those filters for each interface can utilize a large portion of router memory resources. You can use either Enhanced IP Network Services mode or Enhanced Ethernet Network Services mode to improve the scaling and performance specific to routing filters in a subscriber access network that uses statically configured subscriber interfaces.

In configurations where interfaces are created either statically or dynamically and firewall filters are applied dynamically, you must configure the chassis network services to run in enhanced mode. In configurations where interfaces are created statically and firewall filters are applied statically, you must configure chassis network services to run in enhanced mode and also configure each firewall filter for enhanced mode.

Table 4 on page 16 shows the configuration options when determining enhanced network services mode usage.

Table 4: Enhanced Network Services Mode and Firewall Filter Use Case Determination

Interface and Filter Configuration	Chassis Enhanced Mode Required	Firewall Filter Enhanced Mode Required
Dynamically-created interfaces and dynamically-applied filters	Yes	No
Statically-created interfaces and dynamically-applied filters	Yes	No
Statically-created interfaces and statically-applied filters	Yes	Yes

To achieve significant resource savings for the router, combine chassis and filter enhanced mode configuration as follows:

- Install only MPCs in the chassis.

NOTE: **Configuring chassis network services to run one of the enhanced network services modes results in the router enabling only MPCs and MS-DPCs. Because MS-DPCs use compiled firewall filter format, a router chassis that is configured for one of the enhanced network services modes, configuring standard (non-enhanced) firewall filters for use with any MS-DPCs can decrease optimal resource efficiency.**

- When configuring static interfaces on the router, configure chassis network services to run either Enhanced IP Network Services mode or Enhanced Ethernet Network Services mode.
- When statically applying firewall filters to statically-created interfaces, configure any firewall filters for enhanced mode to limit the filter creation to only term-based format.

NOTE: Any firewall filters that are not configured for enhanced mode are created in both compiled and term-based format, even if the chassis is running one of the enhanced network services modes.

Related Documentation

- *Network Services Mode Overview* in the *Junos OS Administration Library for Routing Devices*
- *Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers* in the *Junos OS Administration Library for Routing Devices*
- Configuring a Filter for Use with Enhanced Network Services Mode on page 59

CHAPTER 3

Parameterized Filters

Parameterized Filters Overview

Parameterized filters are configured under a dynamic profile. The filter definition can utilize dynamic-profile variables, allowing their configuration to be customized at session creation time. The user can configure a general baseline filter under a dynamic profile and then provide specific variables of that filter when a dynamic session is activated. These variables can include policing rates, destination addresses, ports, and other items.

In order to provide better scaling, the system analyzes a dynamic profile, and then determines whether the set of variables for one session is the same as for a previous session. Each set of variables is assigned a unique identifier (UID). If a matching filter

already exists, the session creates an interface-specific filter copy of that filter template. If the filter does not already exist, the session reads the configuration and compiles a new filter. This filter is installed as a template with an interface-specific filter copy for the current session pointing at it.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Verifying and Managing Firewall Filter Configuration on page 191
- Unique Identifiers for Firewall Variables in Dynamic Profiles on page 20
- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22
- Dynamic Profile After UID Substitutions for Parameterized Filters on page 24
- Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters on page 25
- Example: Dynamic-Profile Parsing on page 94
- Parameterized Filters Configuration Considerations on page 30
- Parameterized Filter Processing Overview on page 32

Basic Parameterized Filter Syntax

Parameterized filter syntax follows the standard Junos OS filter syntax. When a match condition is met, an action is applied.

Related Documentation

- Basic Classic Filter Syntax on page 10
- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22

Unique Identifiers for Firewall Variables in Dynamic Profiles

The system uses unique identifiers (UIDs) to aid with scaling. The UID enables the system to determine when configuration objects from multiple subscribers are identical and can be shared. In many situations, such as a filter definition, sharing a single filter program among multiple subscribers instead of creating a new program for every subscriber helps to conserve system resources.

Within a dynamic profile a UID is used to name a configuration object. The system assigns the value of the UID (the object's name) based upon all the variables contained within that configuration stanza along with the dynamic profile's name. The assigned UID value consists of the UID name combined with the string_**UID** and a unique number. For instance, the UID **$my-filter** might be given the value **my-filter_UID1022**.

You must first define a UID under the **variable** stanza using the option **uid**. The UID must be defined at the end, after all the variables that are assigned values externally.

```
dynamic-profile test-profile {
  [variables] {
    ... [other variables] ...
```

```
      [my-filter] {
        uid;
      }
    }
  }
```

After a UID has been defined, it can then be used to name an object:

```
dynamic-profile test-profile {
  firewall {
    family inet {
      filter [$my-filter] {
        ... [filter definition that makes use of other variables] ...
      }
    }
  }
}
```

As previously described, the system assigns the value of **$my-filter** depending on the values of the variables used within that filter's definition.

The UID is also used in any other place that the object's name is used. For example, here is an interface stanza to use **$my-filter** as an input filter:

```
dynamic-profile [test-profile] {
  interfaces {
    [$junos-interface-ifd-name]" {
      unit [$junos-interface-unit] {
        family inet {
          filter {
            input [$my-filter];
          }
        }
      }
    }
  }
}
```

You can define multiple configuration objects of the same type (that is, multiple filters) as long as each one uses its own, individual, UID. To ensure that the system selects the correct object when assigning a name, use the **uid-reference** variable.

When the uid-reference is used, it is effectively evaluated twice. First, the value of the uid-reference variable is retrieved. Second, that value is used as the name of a UID and that UID value is retrieved. A uid-reference with a value that is not the name of a UID is considered an error.

A uid-reference is defined similarly to any other variable:

```
dynamic-profile [test-profile] {
  variables {
    [my-filter-selector] {
      uid-reference;
    }
  }
}
```

A uid-reference is used wherever the name of the object is needed. One example is the name of the input filter in the following interface stanza:

```
dynamic-profile [test-profile] {
  interfaces {
    [$junos-interface-ifd-name] {
      unit [$junos-interface-unit] {
        family inet {
          filter {
            input [$my-filter-selector];
          }
        }
      }
    }
  }
}
```

Consider the case where two parameterized filters are defined: **$my-filter-1** and **$my-filter-2**. The **$my-filter-selector** variable might be assigned the value **my-filter-1** or **my-filter-2**, depending upon which filter is appropriate.

Related Documentation

- Parameterized Filter Processing Overview on page 32
- Parameterized Filters Configuration Considerations on page 30

Sample Dynamic-Profile Configuration for Parameterized Filters

In the following sample configuration, the **my-svc-prof** profile provides two different filters: **my-filt-1gw** and **my-filt-2gw**. These filters match on either one or two gateway addresses and apply a policer for that traffic. The name of the filter to apply, the gateway addresses, and the bandwidth for the policer are passed into the service profile from the RADIUS service activation. The uid-reference type supports selection of a particular UID generated object out of multiple objects in the profile. The UID type indicates that a variable is used for UID generation.

```
dynamic-profile {
  [my-svc-prof] {
    variable {
      [my-in-filter] {
        mandatory;
        uid-reference;
      }
      gw1 {
        mandatory;
      }
      gw2 {
        mandatory;
      }
      bw {
        mandatory;
      }
      my-filt-1gw {
        uid;
      }
```

```
    my-filt-2gw {
      uid;
    }
    [my-policer] {
      uid;
    }
  }
  interfaces {
    [$junos-interface-ifd-name] {
      unit [$junos-underlying-interface-unit] {
        family inet {
          filter {
            input [$my-in-filter];
          }
        }
      }
    }
  }
  firewall {
    policer [$my-policer] {
      if-exceeding {
        bandwidth-limit $bw;
        burst-size-limit 15000;
      }
      then discard;
    }
    family inet {
      filter [$my-filt-1gw] {
        interface-specific;
        term t0 {
          from {
            destination-address $gw1;
          }
          then {
            policer [$my-policer];
          }
        }
        term last {
          then {
            count drops;
            discard;
          }
        }
      }
      filter [$my-file-2gw] {
        interface-specific;
        term t0 {
          from {
            destination-address {
              $gw1;
              $gw2;
            }
          }
          then {
            policer [$my-policer];
          }
```

```
        }
        term last {
          then {
            count drops;
            discard;
          }
        }
      }
    }
  }
}
```

Related Documentation

- Dynamic Profile After UID Substitutions for Parameterized Filters on page 24
- Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters on page 25
- Example: Dynamic-Profile Parsing on page 94

Dynamic Profile After UID Substitutions for Parameterized Filters

In the following example, the client session is created on the ge-1/0/0.7 interface and this service is activated:

```
my-svc-prof(my-filt-1gw, 207.17.137.239/32, 0, 5m)
```

A dynamic profile is created by the process. The UIDs assigned by the process are based on the parameters being passed in as well as the sessions previously created.

```
dynamic-profile {
  [my-svc-prof] {
    interfaces {
      ge-1/0/0 {
        unit 7 {
          family inet {
            filter {
              input my-filt-1gw_UID1022;
            }
          }
        }
      }
    }
    firewall {
      policer my-policer_UID1005 {
        if-exceeding {
          bandwidth-limit 5m;
          burst-size-limit 15000;
        }
        then discard;
      }
      family inet {
        filter my-filt-1gw_UID1022 {
          interface-specific;
          term t0 {
```

```
            from {
              destination-address 207.17.137.239/32;
            }
            then {
              policer my-policer_UID1005;
            }
          }
          term last {
            then {
              count drops;
              discard;
            }
          }
        }
        filter my-filt-2gw_UID11018 {
          interface-specific;
          term t0 {
            from {
              destination-address {
                207.17.137.239/32;
                0;
              }
            }
            then {
              policer my-policer_UID1005;
            }
          }
          term last {
            then {
              count drops;
              discard;
            }
          }
        }
      }
    }
  }
}
```

Related Documentation

- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22
- Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters on page 25
- Example: Dynamic-Profile Parsing on page 94

Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters

Table 5 on page 26 compares "Sample Dynamic-Profile Configuration for Parameterized Filters" on page 22 and "Dynamic Profile After UID Substitutions for Parameterized Filters" on page 24 when the following service is activated.

```
my-svc-prof(my-filt-1gw, 207.17.137.239/32, 0, 5m)
```

Table 5: Dynamic Profiles and UID Substitution Comparison

Dynamic Profile Configuration	Result After Substitution	Comment
dynamic-profile {	dynamic-profile {	-
[*my-svc-prof*] {	[*my-svc-prof*] {	-
variable {	-	Define the variables.
[*my-in-filter*] {	-	-
mandatory;	-	-
uid-reference;	-	Assign the name of a UID variable to *my-in-filter.*.
}	-	-
gw1 {	-	-
mandatory;	-	-
}	-	-
gw2 {	-	-
mandatory;	-	-
}	-	-
bw {	-	-
mandatory;	-	-
}	-	-
[*my-filt-1gw*] {	-	-
uid;	-	Type is a UID.
}	-	-
[*my-filt-2gw*] {	-	-
uid;	-	Type is a UID.
}	-	-
[*my-policer*] {	-	-

Table 5: Dynamic Profiles and UID Substitution Comparison *(continued)*

Dynamic Profile Configuration	Result After Substitution	Comment
uid;	-	Type is a UID.
}	-	-
}	-	-
	-	-
interfaces {	interfaces {	-
[*$junos-interface-ifd-name*] {	ge-1/0/0 {	-
unit [*$junos-underlying-interface-unit*] {	unit 7 {	-
family inet {	family inet {	-
filter {	filter {	-
input [*$my-in-filter*];	input my-filt-1gw_UID1022;	Substitute the value of my-filt-1gw for *my-in-filter,* but because my-filt-1gw is a UID reference, substitute the value of *$my-filt-1gw*: my-filt-1gw_UID1022.
}	}	-
}	}	-
}	}	-
}	}	-
}	}	-
		-
firewall {	firewall {	-
policer [*$my-policer*] {	policer my-policer_UID1005 {	Substitute UID name.
if-exceeding {	if-exceeding {	-
bandwidth-limit $bw;	bandwidth-limit 5m;	-
burst-size-limit 15000;	burst-size-limit 15000;	-
}	}	-

Table 5: Dynamic Profiles and UID Substitution Comparison *(continued)*

Dynamic Profile Configuration	Result After Substitution	Comment
then discard;	then discard;	-
}	}	-
family inet {	family inet {	-
filter [*$my-filt-1gw*] {	filter my-filt-1gw_UID1022 {	Substitute UID name
interface-specific;	interface-specific;	-
term t0 {	term t0 {	-
from {	from {	-
destination-address $gw1;	destination-address 207.17.137.239/32;	Substitute $gw1 value.
}	}	-
then {	then {	-
policer [*$my-policer*];	policer my-policer_UID1005;	Substitute UID name.
}	}	-
}	}	-
term last {	term last {	-
then {	then {	-
count drops;	count drops;	-
discard;	discard;	-
}	}	-
}	}	-
}	}	-
filter [*$my-file-2gw*] {	filter my-filt-2gw_UID1018 {	Substitute UID name
interface-specific;	interface-specific;	-
term t0 {	term t0 {	-
from {	from {	-

Table 5: Dynamic Profiles and UID Substitution Comparison *(continued)*

Dynamic Profile Configuration	Result After Substitution	Comment
destination-address {	destination-address {	-
$gw1;	207.17.137.239/32;	Substitute $gw1 value
$gw2;	0;	Substitute $gw2 value
}	}	-
}	}	-
then {	then {	-
policer [*$my-policer*];	policer my-policer_UID1005;	Substitute UID name
}	}	-
}	}	-
term last {	term last {	-
then {	then {	-
count drops;	count drops;	-
discard;	discard;	-
}	}	-
}	}	-
}	}	-
}	}	-
}	}	-
}	}	-
}	}	-

Related Documentation

- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22
- Dynamic Profile After UID Substitutions for Parameterized Filters on page 24
- Example: Dynamic-Profile Parsing on page 94

Parameterized Filters Configuration Considerations

Keep the following considerations in mind when configuring parameterized filters.

Subscriber IP Address

In most deployment scenarios, the interface is based on the subscriber's IP address. Because subscribers may not be unique, they cannot be used in determining similar filters and policers. Do not use the **junos-subscriber-ip-address IP** address as a match candidate. Doing so causes unique filters per subscriber, which inhibits scaling.

Interaction with Static Configuration

Searching for a filter to attach takes place in the following order:

1. Static filter. For example, **firewall family inet filter my-filter**.
2. Fast update filter within the current dynamic profile. For example, **dynamic-profile [*profile-name*] firewall family inet fast-update-filter my-filter**.
3. Parameterized filter within the current dynamic profile. For example, **dynamic-profile [*profile-name*] firewall family inet filter**.

The following static configuration objects may be referenced by a parameterized filter. The search order is first in the static configuration and then in the current dynamic-profile:

- firewall policer
- firewall hierarchical-policer
- three-color policer
- policy-options prefix-list

If an object in the static configuration is being used by an active parameterized filter, you cannot delete that object from the configuration while the subscriber is logged in.

Interface-Specific

All dynamic service filters must be defined as interface-specific.

Service Session Support

Parameterized filters and policers are supported for service activations only, not client sessions.

Filter Naming Conventions

The base filter name is based on the interface and direction (ingress and egress) appended to it. With parameterized filters, the filter-naming process comes from the UID.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Verifying and Managing Firewall Filter Configuration on page 191
- Unique Identifiers for Firewall Variables in Dynamic Profiles on page 20
- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22
- Dynamic Profile After UID Substitutions for Parameterized Filters on page 24
- Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters on page 25
- Example: Dynamic-Profile Parsing on page 94
- Parameterized Filter Processing Overview on page 32

Guidelines for Creating and Applying Parameterized Filters for Subscriber Interfaces

This release supports the dynamic configuration of firewall filters. However, you can also continue to create static firewall filters for interfaces as you do normally, and then dynamically apply those filters to statically created interfaces using dynamic profiles. You can also use dynamic profiles to attach input and output filters through RADIUS.

When creating and applying filters, keep the following in mind:

- This release supports dynamic application of only input and output filters.
- The filters must be interface-specific.
- You can create family-specific **inet** and**inet6** filters.
- You can create interface-specific filters at the **unit** level that apply to any family type (**inet** or **inet6**) configured on the interface.
- You can add or remove both IPv4 and IPv6 filters with the same service activation or deactivation.
- You can remove one filter type without impacting the other type of filter. For example, you can remove IPv6 filters and leave the current IPv4 filters active.
- You can chain up to five input filters and four output filters together.
- If you do not configure and apply a filter, the interface uses the default group filter configuration.
- You cannot modify a firewall filter while subscribers on the same logical interface are bound.

Related Documentation

- Parameterized Filter Processing Overview on page 32

- Parameterized Filters Configuration Considerations on page 30

Parameterized Filter Processing Overview

When creating a parameterized filter, you first define the family address type (**inet** or **inet6**) and then you define one or more terms that specify the filtering criteria and the action to take when a match occurs.

Each term, or rule, consists of the following components:

- Match conditions—Specifies values or fields that the packet must contain. You can define various match conditions, including:
  - IP source address field
  - IP destination address field
  - Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) source port field
  - IP protocol field
  - Internet Control Message Protocol (ICMP) packet type
  - TCP flags
  - interfaces
- Actions—Specifies what to do when a match condition occurs. Possible actions are to accept or discard a packet. In addition, packets can be counted to collect statistical information. If no action is specified for a term, the default action is to accept the packet.

The processing of parameterized filters is the same as classic filters. The order of the terms within a parameterized filter is important. Packets are tested against each term in the order in which the terms are listed in the firewall filter configuration. When a firewall filter contains multiple terms, the router takes a top-down approach and compares a packet against the first term in the firewall filter. If the packet matches the first term, the router executes the action defined by that term to either accept or reject the packet, and no other terms are evaluated. If the router does not find a match between the packet and first term, it then compares the packet to the next term in the firewall filter by using the same match process. If no match occurs between the packet and the second term, the router continues to compare the packet to each successive term defined in the firewall filter until a match is found. If a packet does not match any terms in a firewall filter, the default action is to discard the packet.

You can also specify a precedence (from 0 through 255) for input and output filters within a dynamic profile to force filter processing in a particular order. Setting a lower precedence value for a filter gives it a higher precedence within the dynamic profile. Filters with lower precedence values are applied to interfaces before filters with higher precedence values. A precedence of zero (the default) gives the filter the highest precedence. If no precedence is specified, the filter receives a precedence of zero (highest

precedence). Filters with matching precedence (zero or otherwise) are applied in an unspecified order.

NOTE: Parameterized filters do not support outbound packets that are sourced from the routing engine.

Related Documentation

- Parameterized Filters Configuration Considerations on page 30

Multiple Parameterized Filters

Differing filter match conditions can be achieved by allowing the filter that is being attached to be selected by the unique-identifier--reference capabilities of parameterized filters. If a variable number of terms or varying match conditions are needed, multiple filters are defined. When the service is activated, that activation will select the particular filter that should be applied in the stanza specifying the interface, unit, family and input/output filter:

```
interfaces {
   ge-1/0/0 {
      unit 7 {
         family inet {
            filter {
               input my-filt-1gw-uid1022;
            }
         }
      }
   }
}
```

Related Documentation

- Parameterized Filter Processing Overview on page 32
- Parameterized Filters Configuration Considerations on page 30

IPv4 Parameterized Filter Match Conditions

The following IPv4 match conditions are supported for parameterized filters. Their syntax is the same as the static filter syntax.

address
destination-address
destination-port
destination-port-except
destination-prefix-list
dscp
dscp-except
forwarding-class
forwarding-class-except
icmp-code

icmp-code-except
icmp-type
icmp-type-except
loss-priority
loss-priority-except
packet-length
packet-length-except
port
port-except
precedence
precedence-except
prefix-list
protocol
protocol-except
service-filter-hit
source-address
source-class
source-port
source-port-except
source-prefix-list
ttl
ttl-except

Related Documentation

- *Firewall Filter Match Conditions for IPv4 Traffic*

IPv6 Parameterized Filter Match Conditions

The following IPv6 match conditions are supported for parameterized filters. Their syntax is the same as the static filter syntax.

address
destination-address
destination-port
destination-port-except
destination-prefix-list
forwarding-class
forwarding-class-except
icmp-code
icmp-code-except
icmp-type
icmp-type-except
loss-priority
loss-priority-except
packet-length
packet-length-except
port
port-except
prefix-list

service-filter-hit
source-address
source-class
source-port
source-port-except
source-prefix-list
traffic-class
traffic-class-except

Related Documentation

- *Firewall Filter Match Conditions for IPv6 Traffic*

Parameterized Filter Actions and Modifiers

The following actions and modifiers are supported for parameterized filters. Their syntax is the same as the static filter syntax.

accept
count
discard
forwarding-class
hierarchical-policer
log
loss-priority
next
policer
port-mirror
port-mirror-instance
reject
routing-instance
sample
service-accounting
service-accounting-deferred
service-filter-hit
three-color-policer

Related Documentation

- *Firewall Filter Terminating Actions*
- *Firewall Filter Nonterminating Actions*

Parameterized Filter Policer Actions

The following policer actions are supported for parameterized filters. Their syntax is the same as the existing static policer syntax.

discard
forwarding-class
loss-priority

Related Documentation

- *Firewall Filter Terminating Actions*
- *Firewall Filter Nonterminating Actions*

Hierarchical Policer Overview

Hierarchial policers rate-limit premium traffic separately from the aggregate traffic on an interface as determined by different configured rates. Hierarchical policing uses two token buckets to maintain two rates: an aggregate and a high priority rate, such as 10Mbps and 2Mbps. The traffic is marked differently based on the class of service. Two classes of service are defined for this use: expedited forwarding (EF) and non-expedited forwarding (non-EF). The EF traffic has a user-selectable rate, such as 2Mbps, that is guaranteed before being subject to marking. If there is no EF traffic present, then the non-EF traffic can use up to the 10Mbps rate before being marked. If there is EF traffic present, then the EF traffic is assured up to the 2Mbps (from the 10Mbps) before it becomes subject to marking, but also consumes from the non-EF rate. In this example the EF traffic is guaranteed the 2Mbps and the non-EF traffic has the remaining 8Mbps before being marked.

Hierarchical Policing has the following characteristics:

- Ingress traffic is first classified into premium and non-premium traffic prior to applying a policer.
- The hierarchal policer contains two policers: premium and aggregate. The premium traffic is policed by both the premium policer and aggregate policer. Although premium policer rate-limits the premium traffic, the aggregate policer only decrements the credits but does not drop the packets. The non-premium traffic is rate-limited only by the aggregate policer. Therefore, the premium traffic is assured to have the bandwidth configured for premium and the non-premium traffic is policed to the remaining bandwidth.

Related Documentation

- Hierarchical Policer as Filter Action on page 36

Hierarchical Policer as Filter Action

Hierarchical policer as filter action enables you to have hierarchial policers as one type of filter action. This is useful in provider edge applications using aggregate policing for general traffic and to apply a separate policer for premium traffic on a logical or physical interface. An interface-specific filter can have a hierarchical policer as a filter action whether or not the hierachical policer is a logical interface policer. A non-interface-specific filter can only have a hierarchical policer without using logical interface-specific as a filter action. The following table summarizes where you can use an interface-specific filter.

Interface-specific Filter	Hierarchical Policer Logical-interface-policer	Allowed
no	no	yes

Interface-specific Filter	Hierarchical Policer Logical-interface-policer	Allowed
no	yes	no
yes	no	yes
yes	no	yes

To enable all hierarchical policers of the same name in one filter to share the same policer instance in PFE, use the **filter-specific** statement at the **[edit firewall]** hierarchy level.

Related Documentation

- Hierarchical Policer Overview on page 36

Enhanced Policer Statistics Overview

Enhanced policer statistics enable you to display additional statistics for policers as follows:

- Offered packet statistics for traffic subjected to policing.
- OOS packet statistics for packets that are marked out-of-specification (out-of-spec) by the policer. Changes to all packets that have out-of-spec actions, such as discard, color marking, or forwarding-class, are included this counter.
- Transmitted packet statistics for traffic that is not discarded by the policer. When the policer action is discard, the statistics are the same as the in-spec statistics; when the policer action is non-discard (loss-priority or forwarding-class), the statistics are included in this counter.

This feature is supported on MPC/MIC interfaces on MX Series routers and Multi-Rate Ethernet Enhanced Queuing IP Services DPC with SFP and XFP.

To enable this feature, include the **enhanced-policer** statement at the **[edit chassis]** hierarchy level.

The new **detail** option has been added to the **show firewall**, **show firewall filter** ***filter-name***, and **show policer** commands.

Related Documentation

- *show policer*

Interface-Shared Filters Overview

Interface-shared filters can be defined statically or dynamically, but can only be applied using dynamic profiles, and are supported for both client and service sessions. The same interface-shared instance can be attached to multiple interfaces only if these interfaces reference the same interface-shared filter name and have the same shared-name. The shared-name can either be populated from **$junos-interface-set-name**, where the related client session provides the value, or from a service session variable.

With VLAN subscriber interfaces that use the agent-circuit-identifier information, many subscribers share the same underlying logical interface. Because some of these subscribers are related to each other as part of the same household, you must apply an interface-shared filter to the subscriber logical interfaces that make up the household to be able to filter and police these related subscribers at a household level. All interfaces that share the same interface-shared filter instance share the same set of counters and policer actions.

The base filter name of a parameterized filter is assigned depending upon the profile name and the contents of the filter definition. Therefore, when an interface-shared filter is used with parameterized filters, all service sessions that want to share the same instance of an interface-shared filter must have the exact same parameterized filter and profile. A service session uses a different instance of the interface-shared filter if either the parameterized filter or the profile is different.

Related Documentation

- Example: Interface-Shared Filter Configuration on page 98

CHAPTER 4

Fast Update Filters

Fast Update Filters Overview

The dynamic firewall feature supports classic filters and fast update filters. Fast update filters support subscriber-specific filter values, as opposed to classic filters, which are interface-specific. Fast update filters allow individual filter terms, or rules, to be added or removed from filters without requiring that you recompile the filter after each modification—terms are added and removed when subscriber services are added and removed.

Using the fast update filters feature involves three distinct operations:

1. Creating the filter—You define fast update filters under the **[edit dynamic-profiles *profile-name* firewall family *family*]** hierarchy. The **dynamic-profiles** stanza enables you to use dynamic variables to create subscriber-specific configurations for the filter's match terms. See "Configuring Fast Update Filters" on page 63.
2. Associating the filter with a dynamic profile—You use the **[edit dynamic-profiles *profile-name* interface *interface-name* unit *unit-number* family *family*** hierarchy to associate the filter with a dynamic profile. This is the same procedure used for classic filters. See "Associating Fast Update Filters with Interfaces in a Dynamic Profile" on page 71.
3. Attaching the filter to an interface—When a subscriber logs in, the dynamic profile instantiates the subscriber session and applies the properties of the profile, including the fast update filter, to the session interface. This is the same procedure used for classic filters. Also, similar to classic filters, the name of fast update filters can be provided in a user's RADIUS file.

When a dynamic profile instantiates a subscriber session and applies a fast update filter, the router verifies that the filter is not already present on the session interface. If the filter is not present, the router adds the filter. If the filter is already present on the interface, the router simply adds any new terms that are not in the existing filter. This procedure is reversed when subscriber sessions are deleted. Any terms that were added by a session are then removed when the session is deleted. The filter is deleted when the last subscriber session is deleted.

NOTE: You can optionally specify that a term can be added only once and cannot be modified. See "Match Conditions and Actions in Fast Update Filters" on page 44.

This overview covers:

Fast Update Filter Components

When creating a fast update filter, you define one or more terms that specify the filtering criteria and the action to take when a match occurs.

Each term consists of the following components:

- Match condition—Specifies values or fields that the packet must contain. You can match a maximum of five fields in a fast update filter. A match condition can contain a single value or range. This differs from classic filters, in which terms can have multiple values. However, you can use additional terms to specify multiple ranges. "Fast Update Filter Match Conditions" on page 45 lists the supported match conditions for fast update filters. The order in which the terms appear in a fast update filter is not important, because the router examines the most specific term first. (Classic filters examine the terms in the order in which the terms are listed.)
- Action—Specifies what to do when a packet matches the match condition. If no action is specified for a term, the default action is to accept the packet. "Fast Update Filter Actions and Action Modifiers" on page 46 lists the supported actions for fast update filters.

Terms that are added to the filter during session instantiation must have a unique set of match conditions. Two terms overlap, or conflict, if a packet can match both sets of conditions—as a result, there are two different actions for the packet. You can ensure that terms are unique by using the **$junos-subscriber-ip-address** variable as the **source-address** (for an input filter) or **destination-address** (for an output filter) in the **from** statement. You must then supply the **source-address** or **destination-address** condition, as appropriate, as the first condition in the **match-order** statement.

Related Documentation

- Fast Update Filter Actions and Action Modifiers on page 46
- Fast Update Filter Match Conditions on page 45
- Avoiding Conflicts When Terms Match on page 66

Fast Update Filter Processing

You must use the **match-order** statement to explicitly specify the order in which the router examines filter match conditions. Also, the router examines only those conditions that you include in the **match-order** statement. When a fast update filter contains multiple terms, the router compares a packet against the terms starting with the most specific condition first. When the packet first matches a condition, the router performs the action defined in the term to either accept or reject the packet, and then no other terms are evaluated. If the router does not find a match between the packet and first term, it then compares the packet to the next term in the filter. The router continues to compare the packet to the next specified term until a match is found. If there is no match after all terms have been examined, the router silently drops the packet.

You can specify a precedence (from 0 through 255) for input and output filters within a dynamic profile to force filter processing in a particular order. Setting a lower precedence value for a filter gives it a higher precedence within the dynamic profile. Filters with lower

precedence values are applied to interfaces before filters with higher precedence values. A precedence of zero (the default) gives the filter the highest precedence. If no precedence is specified, the filter receives a precedence of zero (highest precedence). Filters with matching precedence (zero or otherwise) are applied in random order.

Fast Update Filter Names

When a filter is attached to an interface, the router first searches for a classic filter with the specified name, and then uses the classic filter. If no classic filter exists with that name, the router then searches in the dynamic profile for a fast update filter with the specified name, and uses that filter.

If two different dynamic profiles include a fast update filter with the same name, the **match-order** specification of the two filters must be identical. If the two filters are activated on the same interface, the terms are added together.

The router includes the filter name in **show firewall** command results. The router also creates unique names for filter terms and counters for the **show firewall** command.

When a fast update filter is created by the activation of a dynamic profile, the router creates an interface-specific name for the filter. The name uses the following format, which is also used for classic filters:

<filter-name>-<interface-name>.<subunit>-<direction>

For example, an input filter named **httpFilter** on interface **ge-1/0/0.5** is named as follows (**in** indicates an input filter and **out** indicates an output filter):

http-filter-ge-1/0/0.5–in

The router creates unique names for the filter terms and counters by appending the session ID to all term and counter names. Terms that use the **only-at-create** statement have a session-id of 0. Terms and counters use the following format:

<term-name>-<session-id>

<counter-name>-<session-id>

Guidelines for Creating and Applying Fast Update Filters

Fast update filters enable you to create subscriber-specific firewall filters and dynamically apply these filters to statically created interfaces using dynamic profiles. Individual terms can be added to, or removed from, a filter without requiring that the entire filter be recompiled.

When creating and applying fast update filters, keep the following in mind:

- This release supports dynamic application of input and output filters.
- You cannot use the same fast update filter as both an input and output filter in the same dynamic profile attached to an interface.
- Fast update filters must always include terms that permit DHCP traffic to pass. See "Configuring Filters to Permit Expected Traffic" on page 58.

- You can create **family inet** and **inet6** filters.
- You can add or remove both IPv4 and IPv6 filters with the same service activation or deactivation.
- You can remove one filter type without impacting the other type of filter. For example, you can remove IPv6 filters and leave the current IPv4 filters active.
- The **interface-specific** statement is required for all fast update filters.
- The **match-order** statement is required—you must explicitly state the order of the match fields in a fast update filter. See "Configuring the Match Order for Fast Update Filters" on page 64.
- The **match-order** statement uses an implied wildcard for conditions that you specify in the statement. If you specify a condition that is not also configured in the **from** specification of a filter term, the router considers that a wildcard for that condition.
- A filter term can have only a single value or range; however, you can configure multiple terms to specify multiple ranges.
- You can match a maximum of five match conditions in a filter.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- Verifying and Managing Firewall Filter Configuration on page 191

Basic Fast Update Filter Syntax

This section shows the basic fast update filter statement syntax. The first part of this syntax provides the CLI statements to associate an input and output filter with a dynamic profile. The second part of this syntax represents the configured input and output filters associated to the dynamic profile. When a DHCP event occurs, the dynamic profile applies the specified filters to the DHCP client interface on the router.

```
[edit dynamic-profiles profile-name]
interfaces {
   $junos-interface-ifd-name {
      unit $junos-underlying-interface-unit {
         family family {
            filter {
               input filter-name;
               precedence precedence;
               output filter-name;
               precedence precedence;
            }
         }
      }
   }
```

```
}
[edit dynamic-profiles profile-name]
firewall {
  family family {
    fast-update-filter filter-name {
      [desired filter configuration]
    }
    fast-update-filter filter-name {
      [desired filter configuration]
    }
  }
}
```

Related Documentation

- Configuring Fast Update Filters on page 63

Match Conditions and Actions in Fast Update Filters

To create a fast update filter, you use the **term** statement to specify conditions that a packet must have, and to specify the action the router performs when those conditions exist in the packet.

This section covers:

- Match Conditions on page 44
- Actions on page 45
- Adding Terms Only Once on page 45

Match Conditions

Match conditions specify characteristics that a packet must have—if the conditions exist in the packet, the router then performs the specified action. You use the **from** keyword in the **term** statement to specify match conditions for the filter. The packet must match all conditions in the **from** specification for the action to be performed, which also means that their order in the **from** specification is not important.

An individual condition in a **from** specification can contain a single value or range. You can match a maximum of five match conditions in a filter.

"Fast Update Filter Match Conditions" on page 45 lists the match conditions you can use in fast update filters.

NOTE: The router uses an implied wildcard for conditions that you include in the **match-order** statement. If you include a condition that is *not* configured in the **from** specification of a filter term, the router considers that a wildcard for the condition.

For example, if you include the **dscp** condition in the **match-order** statement, but do not configure a **dscp** value in the **from** specification of the filter term, the router performs the action configured in the **then** specification of the filter on all DSCP values.

Actions

Actions and action modifiers specify the operation the router performs when a particular match condition exists in a packet. You use the **then** keyword in the **term** statement to specify the actions to perform on packets whose characteristics match the conditions specified in the preceding **from** specification.

Action modifiers are actions taken in addition to the specified action. You can configure any combination of action modifiers. For the action or action modifier to take effect, all conditions in the **from** specification must match. If you specify **log** as one of the actions in a term, this constitutes a termination action; whether any additional terms in the filter are processed depends on the traffic through the filter. The action modifier operations carry a default **accept** action. For example, if you specify an action modifier and do not specify an action, the specified action modifier is implemented and the packet is accepted.

"Fast Update Filter Actions and Action Modifiers" on page 46 lists the actions and action modifiers you can use in fast update filters.

Adding Terms Only Once

You can optionally specify that a term can be added only when the fast update filter is first created, and cannot be later changed by adding or removing conditions. We recommend that you only use the **only-at-create** option for terms that do not include subscriber-specific data in their match conditions, such as common or default terms (counting the default drop packet, for instance).

Related Documentation

- Configuring Terms for Fast Update Filters on page 65
- Fast Update Filter Match Conditions on page 45
- Fast Update Filter Actions and Action Modifiers on page 46

Fast Update Filter Match Conditions

Table 6: Fast Update Filter Match Conditions

Match Condition	Description
destination-address *prefix*	IP destination address field.
destination-port *number*	TCP or UDP destination port field. Can be a single number, a single range, or one of the standard port synonyms.
dscp *number*	Differentiated services code point. Can be a single number, a single range, or the standard synonyms. IPv4 only.
match-terms *string-of-conditions*	Series of match conditions. Enclose the string within quotation marks and use semicolons to separate entries. For example, **match-terms "protocol tcp; destination-port http";**. Dynamic profile variables are not allowed in the string.

Table 6: Fast Update Filter Match Conditions *(continued)*

Match Condition	Description
protocol *number*	IP protocol field. Can be a single number, a single range, or one of the standard protocol synonyms. IPv4 only.
source-address *prefix*	IP source address field.
source-port *number*	TCP or UDP source port field. Can be a single number, a single range, or one of the standard protocol synonyms.

Related Documentation

- Configuring Fast Update Filters on page 63

Fast Update Filter Actions and Action Modifiers

Table 7: Fast Update Filter Actions and Action Modifiers

Action or Action Modifier	Description
Actions	
accept	Accept the packet.
action-terms *string-of-actions*	A series of multiple actions or action modifiers. Enclose the string within quotation marks and use semicolons to separate entries. For example, **action-terms "log; count http-cnt";**. Dynamic profile variables are not allowed in the string.
discard	Drop the packet silently, without sending an Internet Control Message Protocol (ICMP) message.
ignore-term	Do not add this term to the filter. All match conditions and actions are ignored.
port-mirror	Port mirror packets.
routing-instance *routing-instance*	Forward packets to specified routing instance.
Action Modifiers	
count *counter-name*	Increment the specified counter.
forwarding-class *class*	Classify the packet into one of the following forwarding classes: **as**, **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.
log	Log the packet header information.
loss-priority (high | medium-high | medium-low| low)	Set the loss priority level for packets.
policer *policer-name*	Rate-limit packets based on the specified policer.

Related Documentation

- Configuring Fast Update Filters on page 63

CHAPTER 5

Unicast RPF and Fail Filters

Unicast RPF in Dynamic Profiles for Subscriber Interfaces

Unicast reverse-path forwarding (RPF) provides a way to reduce the effect of denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks on IPv4 and IPv6 interfaces. When you configure unicast RPF on an interface, it checks the packet source address. Packets that pass the check are forwarded. Packets that fail the check are dropped, or if a fail filter is configured, are passed to the filter for further evaluation.

Unicast RPF has two behavioral modes, strict and loose. When you configure unicast RPF in a dynamic profile, strict mode is the default. In strict mode, unicast RPF checks whether the source address of the incoming packet matches a prefix in the routing table, and whether the interface expects to receive a packet with this source address prefix. In loose mode, unicast RPF checks only whether the source address has a match in the routing table. It does not check whether the interface expects to receive a packet from a specific source address.

For both modes, when an incoming packet fails the unicast RPF check, the packet is not accepted on the interface. Instead, unicast RPF counts the packet and sends it to an optional fail filter, if present. The fail filter determines what further action is taken on the packet. In the absence of a fail filter, the packet is silently discarded.

Related Documentation

- Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces on page 75
- For more detailed information about unicast RPF in general, see *Configuring Unicast RPF*

PART 2

Configuration

CHAPTER 6

Configuration Tasks For Filters and Dynamic Profiles

Dynamically Attaching Statically Created Filters for a Specific Interface Family Type

You can dynamically attach statically created filters for either IPv4 (**inet**) or IPv6 (**inet6**) interface types. These filters apply only to interfaces of the specified type.

Before you can attach a statically created filter using a dynamic profile.

1. Create the filters you want to attach.

   See the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices* for detailed information about classic firewall filters and how to create them. See "Configuring Fast Update Filters" on page 63 for information about creating fast update filters.

2. Create a basic dynamic profile.

   See *Configuring a Basic Dynamic Profile*.

To dynamically attach statically created input and output filters:

1. Specify the unit family type you want to use when dynamically attaching the filters.

   a. For IPv4 interfaces, specify the **inet** unit family.

      ```
      [edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1]
      user@host# set family inet
      ```

b. For IPv6 interfaces, specify the **inet6** unit family.

```
[edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1]
user@host# set family inet6
```

2. Specify the input filter in the dynamic profile.

```
[edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1 family inet]
user@host# set filter input static-input-filter
```

3. Specify the output filter in the dynamic profile.

NOTE: The following example specifies an optional precedence value for the output filter.

```
[edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1 family inet]
user@host# set filter output static-output-filter precedence 50
```

Related Documentation

- Classic Filters Overview on page 7
- Fast Update Filters Overview on page 40
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Filters Using RADIUS Variables on page 55
- For information about Junos OS default groups, see the *CLI User Guide*
- For information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Dynamically Attaching Statically Created Filters for Any Interface Type

You can dynamically attach statically created filters for any interface type. These filters apply to any interfaces that are created using the dynamic profile.

NOTE: For an L2TP LNS on MX Series routers, you can attach firewall for static LNS sessions by configuring these at logical interfaces directly on the inline services device (**si-fpc/pic/port**). RADIUS-configured firewall attachments are not supported.

Before you can attach a statically created filter using a dynamic profile.

1. Create the filters you want to attach.

   See the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices* for detailed information about classic firewall filters and how to create them. See "Configuring Fast Update Filters" on page 63 for information about creating fast update filters.

2. Create a basic dynamic profile.

See *Configuring a Basic Dynamic Profile*.

To dynamically attach statically created input and output filters for all interfaces created dynamically using the dynamic profile:

1. Access the dynamic profile, interface, and unit that you want to use when applying the static filters.

   ```
   [edit]
   user@host# edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1
   ```

2. Specify the input filter for the interface unit.

   ```
   [edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1]
   user@host# set filter input static-input-filter
   ```

3. Specify the output filter for the interface unit.

   ```
   [edit dynamic-profiles access-profile interfaces ge-1/1/1 unit 1]
   user@host# set filter output static-output-filter
   ```

Related Documentation

- Classic Filters Overview on page 7
- Fast Update Filters Overview on page 40
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- Dynamically Attaching Filters Using RADIUS Variables on page 55
- For information about Junos OS default groups, see the *CLI User Guide*
- For information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Dynamically Attaching Filters Using RADIUS Variables

You can attach filters to static interfaces by using dynamic profiles. By specifying a variable for the input and output filters, the dynamic profile uses RADIUS VSA attributes for ingress and egress policy.

RADIUS VSA	Attribute Name	Variable
26–10	Ingress-Policy-Name	$junos-input-filter
26–11	Egress-Policy-Name	$junos-output-filter
26–106	IPv6-Ingress-Policy-Name	$junos-input-ipv6-filter
26–107	IPv6-Egress-Policy-Name	$junos-output-ipv6-filter

Before you can attach a filter using RADIUS.

1. Create a basic dynamic profile.

See *Configuring a Basic Dynamic Profile*.

2. Ensure that RADIUS ingress and egress policies are configured appropriately.

   See *Configuring RADIUS Server Parameters for Subscriber Access*.

To dynamically attach IPv4 input and output filters using RADIUS:

1. Specify the dynamic profile you want to attach, the interface, the logical unit number, and family **inet**.

   ```
   [edit]
   user@host# edit dynamic-profiles myProfile interface ge-1/1/1 unit 1 family inet
   ```

2. Specify the IPv4 input filter variable in the dynamic profile.

   ```
   [edit dynamic-profiles myProfile interfaces ge-1/1/1 unit 1 family inet]
   user@host# set filter input $junos-input-filter
   ```

3. Specify the IPv4 output filter variable in the dynamic profile.

   ```
   [edit dynamic-profiles myProfile interfaces ge-1/1/1 unit 1 family inet]
   user@host# set filter output $junos-output-filter
   ```

To dynamically attach IPv6 input and output filters using RADIUS:

1. Specify the dynamic profile you want to attach, the interface, the logical unit number, and family **inet6**.

   ```
   [edit]
   user@host# edit dynamic-profiles myProfile interface ge-1/1/1 unit 1 family inet6
   ```

2. Specify the IPv6 input filter variable in the dynamic profile.

   ```
   [edit dynamic-profiles myProfile interfaces ge-1/1/1 unit 1 family inet6]
   user@host# set filter input $junos-input-ipv6-filter
   ```

3. Specify the IPv6 output filter variable in the dynamic profile.

   ```
   [edit dynamic-profiles myProfile interfaces ge-1/1/1 unit 1 family inet6]
   user@host# set filter output $junos-output-ipv6-filter
   ```

Related Documentation

- Classic Filters Overview on page 7
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- For more information about Junos default groups, see the *CLI User Guide*
- For more information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Defining Dynamic Filter Processing Order

You can force filter processing to occur in a particular order by using the **precedence** statement. You specify a precedence for input and output filters within a dynamic profile at the **[edit dynamic-profiles *profile-name* interfaces (*interface-name* | demux0) unit *logical-unit-number* family *family*]** hierarchy level.

The precedence range is from 0 through 250. Setting a lower precedence value for a filter gives it a higher precedence within the dynamic profile. A precedence of zero (the default) gives the filter the highest precedence. If no precedence is specified, the filter receives a precedence of zero (highest precedence). Filters with matching precedence (zero or otherwise) are applied in random order.

Before you define a precedence for a filter in a dynamic profile.

1. Create the filters you want to attach to the dynamic profile.

   See the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices* for detailed information about firewall filters and how to create them.

2. Create a basic dynamic profile.

   See *Configuring a Basic Dynamic Profile*.

3. Attach the filters to the dynamic profile.

   See "Dynamically Attaching Statically Created Filters for Any Interface Type" on page 54, "Dynamically Attaching Statically Created Filters for a Specific Interface Family Type" on page 53, or "Dynamically Attaching Filters Using RADIUS Variables" on page 55.

To define a precedence for an input and output filter:

1. Specify the input filter precedence in the dynamic profile.

   ```
   [edit dynamic-profiles profile-name interfaces interface-name unit logical-unit-number
     family family]
   user@host# set filter input precedence 50
   ```

2. Specify the output filter precedence in the dynamic profile.

   ```
   [edit dynamic-profiles profile-name interfaces interface-name unit logical-unit-number
     family family]
   user@host# set filter output precedence 5
   ```

Related Documentation

- Classic Filters Overview on page 7
- For information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Configuring Firewall Filter Bypass

You can streamline the filter process, decrease the amount of packet handling for each filter in a chain, and effectively bypass unnecessary filters by using the **service-filter-hit**

match/action combination at the **[edit *firewall family family-name* filter *filter-name term term-name*]** hierarchy level.

To bypass firewall filters using the **service-filter-hit** match/action combination, you configure the **service-filter-hit** action in at least one filter in the chain and configure **service-filter-hit** match condition in any subsequent filters that you want to bypass. All packets must pass through each filter in a chain. However, after the **service-filter-hit** flag is set in a packet, the packet "bypasses" any subsequent filters that contain the **service-filter-hit** match condition and more efficiently passes (accepts) marked packets and accelerating the filter process.

NOTE: When using the service-filter-hit match/action combination, the order in which the filters are applied is important. You can ensure the order in which the filters are processed by specifying a filter precedence value for the interface. See "Defining Dynamic Filter Processing Order" on page 57 for more information about dynamic filter processing.

To bypass filter processing:

1. Specify the **service-filter-hit** action for any filters in a filter chain.

   ```
   [edit firewall family inet filter video term 1]
   user@host# set then service-filter-hit
   ```

   When the match conditions for the filter are met, the **service-filter-hit** action is set to indicate to subsequent filters that further processing is unnecessary.

2. Specify the **service-filter-hit** match condition in any filters with a lower precedence (that is, a higher precedence statement value) that you want to detect **service-filter-hit** actions applied from previous filters in the chain.

   ```
   [edit firewall family inet filter data term 1]
   user@host# set from service-filter-hit
   ```

3. Configure the filter to pass (accept) any packet that has a **service-filter-hit** action applied from any previous filters.

   ```
   [edit firewall family inet filter data term 1]
   user@host# set then accept
   ```

Related Documentation

- Classic Filters Overview on page 7
- Defining Dynamic Filter Processing Order on page 57
- Example: Bypassing Firewall Filters on page 90

Configuring Filters to Permit Expected Traffic

You must explicitly configure your firewall filter to permit expected traffic, such as DHCP traffic, to pass. Otherwise, the expected traffic is denied when the filter is applied to the interface. This requirement applies to both classic and fast update filters.

The following example shows a fast update filter that might be used to accept DHCP traffic. The actual filter you use depends on the expected traffic in your network.

In the example, the term **allow-dhcp** accepts all DHCP traffic from all source addresses. The term also includes the **only-at-create** option to specify that the term is applied only when the filter is first applied. The term **sub-allow-dhcp** includes the Junos OS predefined variable **$junos-subscriber-ip-address**, which permits all subscriber-specific DHCP traffic.

The **match-order** statement configuration lists the conditions from most-specific to least-specific, as recommended in "Configuring the Match Order for Fast Update Filters" on page 64. Because this filter is designed to permit ingress DHCP traffic, the **source-address** condition is listed first.

```
firewall {
    family inet {
        fast-update-filter psf1 {
            interface-specific;
            match-order [ source-address destination-address protocol destination-port ];
            term allow-dhcp {
                only-at-create;
                from {
                    source-address 0.0.0.0/32;
                    destination-address 255.255.255.255/32;
                    destination-port 67;
                    protocol udp;
                }
                then accept;
            }
            term sub-allow-dhcp {
                from {
                    source-address $junos-subscriber-ip-address;
                    destination-address 192.168.1.2/32;
                    destination-port 67;
                    protocol udp;
                }
                then accept;
            }
        }
    }
}
```

Related Documentation

- Configuring the Match Order for Fast Update Filters on page 64
- Configuring Terms for Fast Update Filters on page 65

Configuring a Filter for Use with Enhanced Network Services Mode

For a statically-applied enhanced mode filter to function on statically created interfaces, you must include the **enhanced mode** statement in each filter. However, you do not need to configure the **enhanced mode** statement in filters that are dynamically applied to either static or dynamically-created interfaces.

NOTE: For either static or dynamic interfaces to use enhanced network services mode, you must configure the router chassis network services to use either Enhanced IP Network Services mode or Enhanced Ethernet Network Services mode. By configuring chassis network services to run in one of the enhanced modes, the router enables only MPCs and MS-DPCs in the chassis. See "Firewall Filters and Enhanced Network Services Mode Overview" on page 15 for details.

To configure a stateless firewall filter to use enhanced mode:

1. Create or edit the stateless firewall filter.

   NOTE: You can configure enhanced mode firewall filters for only inet and inet6 filter families.

   For IPv4:

   ```
   [edit]
   user@host# edit firewall family inet filter filter-name
   ```

   For IPv6:

   ```
   [edit]
   user@host# edit firewall family inet6 filter filter-name
   ```

2. Specify the filter as an enhanced mode filter.

   ```
   [edit firewall family inet filter filter-name]
   user@host# set enhanced-mode
   ```

3. Configure or modify any filter terms.

   See any of the filter configuration examples described in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*.

Related Documentation

- *Understanding How to Use Firewall Filters* in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*
- *Network Services Mode Overview* in the *Junos OS Administration Library for Routing Devices*
- Firewall Filters and Enhanced Network Services Mode Overview on page 15
- *Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers* in the *Junos OS Administration Library for Routing Devices*
- Dynamic Firewall Filters Overview on page 4

Dynamically Applying Ascend-Data-Filter Policies to Subscriber Sessions

Subscriber management enables you to use dynamic profiles to dynamically apply policies that are defined in Ascend-Data-Filters (RADIUS attribute 242) to subscriber sessions. The dynamic profiles include a Junos OS predefined variable that maps the rules and actions defined in the Ascend-Data-Filter to Junos OS features. The RADIUS administrator configures the Ascend-Data-Filter on the RADIUS server in a separate operation.

Subscriber management dynamic profiles use the following Junos OS predefined variables to map family-specific Ascend-Data-Filter rules to Junos OS filter functionality:

- **$junos-adf-rule-v4**—Used for IPv4 family **inet**.
- **$junos-adf-rule-v6**—Used for IPv6 family **inet6**.

To configure a dynamic profile to dynamically apply the policy defined by an Ascend-Data-Filter to a subscriber session:

1. Specify the dynamic profile in which you want to include the Ascend-Data-Filter. Specify the interface, the logical unit number, and the family type.

   [edit]
   user@host# **edit dynamic-profiles** *profile-name* **interfaces** *interface-name* **unit** *logical-unit-number* **family** *family*

2. Specify that you want to include an Ascend-Data-Filter in the dynamic profile.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family*]
   user@host# **edit filter adf**

3. Specify the Junos OS predefined variable that maps the Ascend-Data-Filter actions to Junos OS filter functionality. Use the variable that corresponds to the specified family type.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter adf]
   user@host# **set rule ($junos-adf-rule-v4 | $junos-adf-rule-v6)**

   NOTE: You can also statically configure the Ascend-Data-Filter in this step by entering the filter in hexadecimal format, rather than use a predefined variable. You might use a static filter for testing purposes.

4. (Optional) Suppress error-reporting in the event the RADIUS reply messages do not include the Ascend-Data-Filter attribute.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter adf]
   user@host# **set not-mandatory**

5. (Optional) Enable the counter feature. The counter increments each time a packet matches the rule.

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter adf]
user@host# **set counter**

6. (Optional) Specify the input precedence used to establish the order in which filters on the interface are applied. A lower precedence value equals a higher precedence. The precedence relates to other dynamic filters configured on the same interface.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter adf]
   user@host# **set input-precedence** *precedence*

7. (Optional) Specify the output precedence used to establish the order in which filters on the interface are applied. A lower precedence value equals a higher precedence. The precedence relates to other dynamic filters configured on the same interface.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter adf]
   user@host# **set output-precedence** *precedence*

Related Documentation

- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11
- Ascend-Data-Filter Attribute Fields on page 12
- Verifying and Managing Dynamic Ascend-Data-Filter Policy Configuration on page 191
- Example: Configuring Dynamic Ascend-Data-Filter Support for Subscriber Access on page 82
- Example: Configuring Static Ascend-Data-Filter Support for Subscriber Access on page 85

CHAPTER 7

Configuration Tasks for Fast Update Filters

Configuring Fast Update Filters

You configure a fast update filter in a dynamic profile—this enables you to use dynamic variables in the filter configuration. After you configure fast update filters, you then use the **dynamic-profiles** syntax to associate the filter with the subscriber interface.

To configure a fast update filter for subscriber access:

1. Access the dynamic profile you want to use.

   ```
   [edit]
   user@host# edit dynamic-profiles myProfile
   ```

2. Specify that you want to configure a firewall, and specify the family.

   ```
   [edit dynamic-profiles myProfile]
   user@host# edit firewall family inet
   ```

3. Specify that you want to configure a fast update filter and assign a name to the filter.

   ```
   [edit dynamic-profiles myProfile firewall family inet]
   user@host# edit fast-update-filter httpFilter
   ```

4. Specify the **interface-specific** statement. This statement is mandatory.

   ```
   [edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
   user@host# set interface-specific
   ```

5. Configure the match order to use for the filter terms.

   ```
   [edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
   user@host# set match-order [source-address protocol destination-port]
   ```

   See "Configuring the Match Order for Fast Update Filters" on page 64.

6. Specify that you want to configure a term for the filter and assign the name to the term. Configure the match conditions and actions for the term.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# edit term term1

[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter term
  term1]
user@host# set from protocol tcp
user@host# set from source-address $junos-subscriber-ip-address
user@host# set from destination-port http
user@host# set then count http-cnt
```

See "Configuring Terms for Fast Update Filters" on page 65.

Related Documentation

- Configuring the Match Order for Fast Update Filters on page 64
- Configuring Terms for Fast Update Filters on page 65
- Associating Fast Update Filters with Interfaces in a Dynamic Profile on page 71
- Fast Update Filters Overview on page 40
- *Dynamic Profiles Overview*
- For information about firewall filters, see *Guidelines for Configuring Firewall Filters* and *Guidelines for Applying Firewall Filters*in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*.

Configuring the Match Order for Fast Update Filters

You must include the **match-order** statement to explicitly specify the order in which router examines the match conditions. The router examines only those match conditions that you include in the statement. You can match a maximum of five conditions.

NOTE: If the **match-order** statement contains a condition that is not specified in the **from** statement of a term, the router considers that a wildcard for that condition.

If you use the same fast update filter in multiple dynamic profiles, you must configure the same match order for all profiles.

To configure the order in which the router examines the match conditions of a fast update filter:

1. Access the fast update filter:

```
[edit dynamic-profiles myProfile]
user@host# edit firewall family inet fast-update-filter httpFilter
```

2. Specify the mandatory **interface-specific** statement.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
```

```
user@host# set interface-specific
```

3. Configure the match order for the match conditions in the filter. Use brackets to enclose multiple match conditions.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# set match-order [source-address protocol destination-port]
```

Related Documentation

- Configuring Fast Update Filters on page 63
- Configuring Terms for Fast Update Filters on page 65
- Fast Update Filters Overview on page 40
- *Dynamic Profiles Overview*
- Fast Update Filter Match Conditions on page 45
- For information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Configuring Terms for Fast Update Filters

A fast update filter consists of one or more terms. A term is made up of one or more match conditions and the action to take when a packet matches the specified conditions.

To configure a term for a fast update filter:

1. Access the fast update filter.

```
[edit dynamic-profiles myProfile]
user@host# edit firewall family inet fast-update-filter httpFilter
```

2. Create the new term and assign a name to the term.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# set term term1
```

3. Configure the match condition for the term. See "Fast Update Filter Match Conditions" on page 45 for the supported match conditions for fast update filters.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# set from protocol tcp
user@host# set from source-address $junos-subscriber-ip-address
user@host# set from destination-port http
```

4. Configure the action that the router takes when the match conditions are met. See "Fast Update Filter Actions and Action Modifiers" on page 46 for the supported actions for fast update filters.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# set then accept
```

5. (Optional) Configure the action modifiers that you want the router to take when the match conditions are met. See "Fast Update Filter Actions and Action Modifiers" on page 46 for the supported action-modifiers for fast update filters.

```
[edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
user@host# set then count http-cnt
```

6. (Optional) Configure the term to be added only once, when the fast update filter is first created.

   ```
   [edit dynamic-profiles myProfile firewall family inet fast-update-filter httpFilter]
   user@host# set only-at-create
   ```

Related Documentation

- Configuring Fast Update Filters on page 63
- Configuring the Match Order for Fast Update Filters on page 64
- Fast Update Filters Overview on page 40
- Fast Update Filter Match Conditions on page 45
- Fast Update Filter Actions and Action Modifiers on page 46
- For additional information about firewall filter terms, see the following topics in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*
  - *Stateless Firewall Filter Overview*
  - *Stateless Firewall Filter Components*

Avoiding Conflicts When Terms Match

A fast update filter can contain multiple terms, each with a variety of match conditions. However, when you configure multiple terms in a filter, you must ensure that the terms do not overlap, or conflict with each other. Two terms are considered to overlap when it is possible for a packet to match all conditions of both terms. Because each term specifies a different action for matches, the router cannot determine which action to take. When terms overlap, a conflict error occurs and the session fails when the dynamic profile attempts to apply the filter. The error log indicates the overlapping terms.

How the Router Evaluates Terms in a Filter

The router creates a table of match conditions when examining terms. The table, which is similar to a routing table, is based on the conditions included in the **match-order** statement. When the router receives a packet, the router examines the packet's contents in the sequence specified in the **match-order** statement.

For example, using the sample configuration in the following Match-Order Example, the router first examines the packet's **source-address**, then the **destination-address**, and finally the **destination-port**. As shown in the following table, the two terms in the filter do not overlap because each term has a different **destination-port** specification. The router then takes the appropriate filter action for the term that matches the **destination-port** value of the packet.

Term	source-address	destination-address	destination-port	Action
t55	subscriber's address	3.1.1.2/32	http	count t55_cntr accept

Term	source-address	destination-address	destination-port	Action
t999	subscriber's address	3.1.1.2/32	https	count t999_cntr accept

Match-Order Example

```
firewall {
  family inet {
    fast-update-filter psf1 {
      interface-specific;
      match-order [ source-address destination-address destination-port ];
      term t55 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 3.1.1.2/32;
          destination-port http;
        }
        then {
          count t55_cntr;
          accept;
        }
      }
      term t999 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 3.1.1.2/32;
          destination-port https;
        }
        then {
          count t999_cntr;
          accept;
        }
      }
    }
  }
}
```

Using Implied Wildcards

This section shows an example of how you might use an implied wildcard specification in the match configuration. A condition in the **match-order** statement is an implied wildcard when that condition is not configured in the **from** specification of a term in the filter.

NOTE: When you use ranges (for example, a range of values or a wildcard) in terms, the ranges must not overlap—overlapping ranges create a conflict error. However, you can configure a range in one term and an exact match in another term. For example, in the following filter table, the wildcard destination port value in term t3 does not overlap the destination port specifications in terms t55 and t999 because the http and https values are exact matches.

In the Implied Wildcard Example configuration, the router views the **destination-port** condition in the **match-order** statement as an implied wildcard for term **t3**, because there is no **destination-port** value configured in that term. As a result, the wildcard specifies that for term **t3** any **destination-port** value is accepted. The filter table appears as follows:

Term	source-address	destination-address	destination-port	Action
t3	subscriber's address	3.1.1.2/32	any (wildcard)	count t3_cntr accept
t55	subscriber's address	3.1.1.2/32	http	count t55_cntr accept
t999	subscriber's address	3.1.1.2/32	https	count t999_cntr accept

In the following filter configuration, traffic with a destination port of **http** matches term **t55** and traffic with a destination port of **https** matches term **t999**. Traffic with a destination port other than **http** or **https** matches term **t3**, which is the implied wildcard.

Implied Wildcard Example

```
firewall {
  family inet {
    fast-update-filter psf1 {
      interface-specific;
      match-order [ source-address destination-address dscp protocol destination-port
        ];
      term t3 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 3.1.1.2/32;
        }
        then {
          count t3_cntr;
          accept;
        }
      }
      term t55 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 3.1.1.2/32;
          destination-port http;
        }
        then {
          count t55_cntr;
          accept;
        }
      }
      term t999 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 3.1.1.2/32;
```

```
            destination-port https;
          }
          then {
            count t999_cntr;
            accept;
          }
        }
      }
    }
  }
```

Conflict Caused by Overlapping Ranges

This section shows two examples of overlapping ranges in terms. When you use ranges (such as a wildcard or a range of values) in terms, the ranges must not overlap—overlapping ranges create a conflict error and the session fails.

In the following filter configuration, the **destination-port** ranges in the two terms overlap. Ports in the range from 50 through 80 match both term **src0** and term **src1**, which each specify different actions to take.

NOTE: You can configure a range in one term and an exact match in another term. See the section, *Using Implied Wildcards*, for an example that uses a wildcard for a match condition in one term and an exact match for the condition in a second term.

Term	source-address	destination-address	destination-port	Action
src0	subscriber's address	10.1.1.2/32	0–80	count c1_cntr accept
src1	subscriber's address	10.1.1.2/32	50–100	count c2_cntr accept

Overlapping Ranges Example 1

```
firewall {
  family inet {
    fast-update-filter fuf–src {
      interface-specific;
      match-order [ source-address destination-address destination-port ];
      term src0 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 10.1.1.2/32;
          destination-port 0–80;
        }
        then {
          count c1_cntr;
          accept;
        }
      }
```

```
    term src1 {
      from {
        source-address $junos-subscriber-ip-address;
        destination-address 10.1.1.2/32;
        destination-port 50–100;
      }
      then {
        count c2_cntr;
        accept;
      }
    }
  }
```

In this filter configuration, the **protocol** specification in terms **src21** and **src22** use the implied wildcard, which configures a range for each term. Because overlapping ranges are not allowed, a conflict error results.

Term	source-address	destination-address	protocol	destination-port	Action
src20	subscriber's address	10.1.1.2/32	udp	any (wildcard)	count c20_cntr accept
src21	subscriber's address	10.1.1.2/32	any (wildcard)	http	count c21_cntr accept
src21	subscriber's address	10.1.1.2/32	any (wildcard)	https	count c22_cntr accept

Overlapping Ranges Example 2

```
firewall {
  family inet {
    fast-update-filter fuf–src2 {
      interface-specific;
      match-order [ source-address destination-address protocol destination-port ];
      term src20 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 10.1.1.2/32;
          protocol udp;
        }
        then {
          count c20_cntr;
          accept;
        }
      }
      term src21 {
        from {
          source-address $junos-subscriber-ip-address;
          destination-address 10.1.1.2/32;
          destination-port http;
```

```
        }
        then {
            count c21_cntr;
            accept;
        }
    }
    term src22 {
        from {
            source-address $junos-subscriber-ip-address;
            destination-address 10.1.1.2/32;
            destination-port https;
        }
        then {
            count c22_cntr;
            accept;
        }
    }
}
```

Related Documentation

- Configuring Fast Update Filters on page 63
- Configuring Terms for Fast Update Filters on page 65
- Configuring the Match Order for Fast Update Filters on page 64

Associating Fast Update Filters with Interfaces in a Dynamic Profile

After you configure the fast update filter, you reference the filter in the **interfaces** stanza of a dynamic profile. When the dynamic profile instantiates a subscriber session, the router applies the terms of the filter to the interface.

To apply a fast update filter to an interface in a dynamic profile:

1. Access the dynamic profile you want to use.

   ```
   [edit]
   user@host# edit dynamic-profiles myProfile
   ```

2. Specify the interface for the dynamic profile—use the dynamic interface variable.

   ```
   [edit dynamic-profiles myProfile]
   user@host# edit interfaces $junos-interface-ifd-name
   ```

3. Specify the underlying interface—use the unit number variable.

   ```
   [edit dynamic-profiles myProfile interfaces "$junos-interface-ifd-name"]
   user@host# edit unit $junos-underlying-interface-unit
   ```

4. Specify the family. Use **inet** if you are using IPv4 filters or **inet6** for IPv6 filters.

   ```
   [edit dynamic-profiles myProfile interfaces "$junos-interface-ifd-name" unit
     "$junos-underlying-interface-unit"]
   user@host# edit family inet
   ```

5. Specify the filters that you want to apply to the interface.

   ```
   [edit dynamic-profiles myProfile interfaces "$junos-interface-ifd-name" unit
     "$junos-underlying-interface-unit" family inet]
   ```

```
user@host# set filter input httpFilter
user@host# set filter output myOutFilter
```

Related Documentation

- *Dynamic Profiles Overview*
- *Configuring Static Subscriber Interfaces in Dynamic Profiles*
- *Associating Dynamic Profiles with Statically Created Interfaces*
- Fast Update Filters Overview on page 40
- For information about firewall filters, see *Guidelines for Configuring Firewall Filters* and *Guidelines for Applying Firewall Filters*in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*.

CHAPTER 8

Configuration Tasks for Dynamic Service Sets

Associating Service Sets with Interfaces in a Dynamic Profile

After you configure a service set, you use a dynamic profile to dynamically associate the service set with interfaces. You reference the filter in the **interfaces** stanza of a dynamic profile. When the dynamic profile instantiates a subscriber session, the router applies the terms of the filter to the interface.

To apply a service set to an interface in a dynamic profile:

1. Access the dynamic profile you want to use.

   ```
   [edit]
   user@host# edit dynamic-profiles myProfile
   ```

2. Specify the interface for the dynamic profile—use the dynamic interface variable.

   ```
   [edit dynamic-profiles myProfile]
   user@host# edit interfaces $junos-interface-ifd-name
   ```

3. Specify the underlying interface—use the unit number variable.

   ```
   [edit dynamic-profiles myProfile interfaces “$junos-interface-ifd-name”]
   user@host# edit unit $junos-underlying-interface-unit
   ```

4. Specify the family. Dynamic service sets are supported only on **family inet** (IPv4).

   ```
   [edit dynamic-profiles myProfile interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit”]
   user@host# edit family inet
   ```

5. Specify the input and output service sets that you want to apply to the interface.

   ```
   [edit dynamic-profiles myProfile interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit” family inet]
   user@host# set service input service-set inputService_200
   user@host# set service input post-service-filter postService_15
   user@host# set service output service-set outputService_320
   ```

Related Documentation

- Dynamic Service Sets Overview on page 5
- Verifying and Managing Service Sets Information on page 192
- For information about creating service sets, see "Service Set Configuration Guidelines" in the *Junos OS Services Interfaces Library for Routing Devices*.
- For information about statically applying service sets to interfaces, see *Applying Filters and Services to Interfaces* in the *Junos OS Services Interfaces Library for Routing Devices*.

CHAPTER 9

Configuration Tasks for Unicast and Fail Filters

Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces

This topic provides a summary of unicast RPF configuration for subscriber interfaces in dynamic profiles on MX Series routers. Unicast RPF provides a way to reduce the effect of denial-of-service attacks on IPv4 and IPv6 interfaces by checking the source IP address against the routing table. Packets that do not match are silently discarded, unless an optional fail filter is configured. The fail filter performs an additional check and directs some action be taken on certain packets. Typical actions include logging the packets or passing them even though they failed the RPF check.

NOTE: **Although the fail filter is technically optional, for dynamic profiles in a DHCP environment you must configure a filter to pass DHCP packets. By default, the RPF check prevents DHCP packets from being accepted on interfaces protected by the RPF check. The fail filter identifies the DHCP packets and passes them on.**

To configure unicast RPF in dynamic profiles:

1. Enable unicast RPF on one or more interfaces in a dynamic profile.

   See "Configuring Unicast RPF in Dynamic Profiles for Subscriber Interfaces" on page 76.

2. (Optional) Create a fail filter to evaluate failed packets and perform further actions.

   See "Configuring a Fail Filter for Unicast RPF in Dynamic Profiles for Subscriber Interfaces" on page 76.

Related Documentation

- Unicast RPF in Dynamic Profiles for Subscriber Interfaces on page 49
- Example: Configuring Unicast RPF in a Dynamic Profile on MX Series Routers on page 100

Configuring Unicast RPF in Dynamic Profiles for Subscriber Interfaces

This topic describes how to configure unicast RPF for subscriber interfaces in dynamic profiles on MX Series routers.

To configure a unicast RPF with a fail filter in a dynamic profile:

1. Access the dynamic profile.

   ```
   [edit]
   user@host# edit dynamic-profiles profile-name
   ```

2. Access the interface and specify the address family

   ```
   [edit dynamic-profiles profile-name]
   user@host# edit interfaces interface-name unit logical-unit-number family inet
   ```

3. Configure the RPF check and specify the fail filter.

   ```
   [edit dynamic-profiles profile-name interface interface-name unit logical-unit-number
     family inet]
   user@host# set rpf-check fail-filter filter-name
   ```

Related Documentation

- Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces on page 75
- Example: Configuring Unicast RPF in a Dynamic Profile on MX Series Routers on page 100

Configuring a Fail Filter for Unicast RPF in Dynamic Profiles for Subscriber Interfaces

This topic describes how to configure a fail filter at the **[edit firewall]** hierarchy level that can be optionally applied by unicast RPF for subscriber interfaces in dynamic profiles on MX Series routers.

NOTE: In contrast to statically configured fail filters, RPF-check fail filters used in a dynamic profile cannot be specific to a particular interface.

To configure a firewall fail filter:

1. Create the filter.

   ```
   [edit]
   user@host# edit firewall family inet filter filter-name
   ```

2. Specify a term for the filter.

   ```
   [edit firewall family inet filter filter-name]
   user@host# edit term term-name
   ```

3. Configure the match conditions for the filter.

   [edit firewall family inet filter *filter-name* term *term-name*]
   user@host# **set from** ***match-conditions***

4. Configure the actions to be taken for the matching packets.

   [edit firewall family inet filter *filter-name* term *term-name*]
   user@host# **set then** ***actions***

5. (Optional) Repeat Steps 3 and 4 for additional filter terms.

Related Documentation

- Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces on page 75
- Example: Configuring Unicast RPF in a Dynamic Profile on MX Series Routers on page 100

CHAPTER 10

Examples

Examples: Configuring Static Filters

This topic provides some static filter configuration examples.

```
firewall {
    policer p1 {
        if-exceeding {
            bandwidth-limit 5m;
            burst-size-limit 10m;
        }
        then discard;
    }
    family inet {
        filter dfwd {
            interface-specific;
            term 1 {
                from {
                    source-address {
                        192.1.1.0/24;
                    }
                }
                then {
                    count c1;
                    next term;
                }
```

```
    }
    term 2 {
      from {
        source-address {
          192.2.1.0/24;
        }
      }
      then count c2;
    }
    term 3 {
      then accept;
    }
  }
  filter dfwd1 {
    interface-specific;
    term 1 {
      from {
        address {
          192.1.1.0/24;
        }
      }
      then {
        discard;
      }
    }
  }
  filter tos {
    interface-specific;
    term 1 {
      from {
        precedence priority;
      }
      then forwarding-class assured-forwarding;
    }
    term 2 {
      then {
        log;
        accept;
      }
    }
  }
  filter dfwd2 {
    interface-specific;
    term 1 {
      from {
        forwarding-class best-effort;
      }
      then {
        sample;
        forwarding-class expedited-forwarding;
      }
    }
    term 2 {
      then accept;
    }
  }
```

```
    filter nodhcp {
        term dhcpdiscover {
            from {
                protocol udp;
                source-port 68;
                destination-port 67;
            }
            then {
                discard;
            }
        }
        term others {
            then accept;
        }
    }
    filter p1 {
        interface-specific;
        term 1 {
            from {
                precedence priority;
            }
            then {
                policer p1;
                log;
            }
        }
        term 2 {
            then accept;
        }
    }
    filter dscp {
        interface-specific;
        term 1 {
            from {
                dscp af11;
            }
            then log;
        }
        term 2 {
            then accept;
        }
    }
    filter tcm {
        interface-specific;
        term 1 {
            from {
                dscp af11;
            }
            then policer p1;
        }
        term 2 {
            then accept;
        }
    }
}
traceoptions {
```

```
        flag dynamic;
    }
}
```

Related Documentation

- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54
- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53

Example: Configuring Dynamic Ascend-Data-Filter Support for Subscriber Access

This example shows how to configure support for dynamic Ascend-Data-Filter policies.

Requirements

- Ensure that the Ascend-Data-Filter has been configured on the RADIUS server.
- Create the dynamic profile. See *Dynamic Profiles Overview*.
- Configure RADIUS support. See *Configuring RADIUS Server Parameters for Subscriber Access*.

Overview

Ascend-Data-Filters are configured on a RADIUS server, and contain rules that create policies. Subscriber management uses a dynamic profile to obtain the Ascend-Data-Filter attribute (RADIUS attribute 242) from the RADIUS server and apply the policy to a subscriber session.

- Specify the dynamic profile to use to apply the Ascend-Data-Filter policy to the subscriber session.
- Specify the Junos OS predefined variable that maps the Ascend-Data-Filter rules to Junos OS filter functionality.
- Configure optional settings, which include counting the rule usage and setting the precedence order for the filter.

Configuration

Step-by-Step Procedure

To configure dynamic Ascend-Data-Filter support:

1. Specify the dynamic profile in which you want to include the Ascend-Data-Filter, and configure the interface, the logical unit number, and the family type.

```
[edit]
user@host# edit dynamic-profiles adf-profile-v4 interfaces
   $junos-interface-ifd-name unit $junos-underlying-interface-unit family inet
```

2. Specify that you want to include an Ascend-Data-Filter in the dynamic profile and provide the Junos OS predefined variable as the rule that maps the Ascend-Data-Filter actions to Junos OS filter functionality.

```
[edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
  “$junos-underlying-interface-unit” family inet]
user@host# set filter adf rule $junos-adf-rule-v4
```

3. Enable the counter for the rule.

```
[edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
  “$junos-underlying-interface-unit” family inet]
user@host# set filter adf counter
```

4. Specify the precedence for received packets on the interface.

```
[edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
  “$junos-underlying-interface-unit” family inet]
user@host# set filter adf input-precedence 75
```

5. Specify the precedence for transmitted packets on the interface.

```
[edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
  “$junos-underlying-interface-unit” family inet]
user@host# set filter adf output precedence 80
```

Results From configuration mode, confirm your configuration by entering the **show dynamic-profiles** command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show dynamic-profiles
...
adf-profile-v4 {
  interfaces {
    "$junos-interface-ifd-name" {
      unit "$junos-underlying-interface-unit" {
        family inet {
          filter {
            adf {
              rule "$junos-adf-rule-v4";
              counter;
              input-precedence 75;
              output-precedence 80;
              ...
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is working properly, perform these tasks:

- Verifying that Dynamic Ascend-Data-Filter Rules Are Applied to Subscriber Sessions on page 84
- Verifying Dynamic Ascend-Data-Filter Usage on page 85

Verifying that Dynamic Ascend-Data-Filter Rules Are Applied to Subscriber Sessions

Purpose Verify that the Ascend-Data-Filter rules were attached to the subscriber.

Action From operational mode, enter the **show subscribers extensive** command.

```
user@host>show subscribers extensive
  Type: DHCP
  User Name: user1-adf
  IP Address: 192.168.1.10
  IP Netmask: 255.255.255.0
  Logical System: default
  Routing Instance: default
  Interface: ge-1/0/0.0
  Interface type: Static
  Dynamic Profile Name: adf-profile-v4
  MAC Address: 00:10:94:00:00:01
  State: Active
  Radius Accounting ID: 5
  Login Time: 2010-08-12 14:06:27 PDT
  ADF IPv4 Input Filter Name: __junos_adf_5-ge-1/0/0.0-inet-in
          Rule 0: 0101010000000000d87f920000180000000000000000000
              from {
                destination-address 216.127.146.0/24;
              }
              then {
                accept;
              }
          Rule 1: 0100010000000000000000000000060000000001900020000
              from {
                protocol 6;
                destination-port 25;
              }
              then {
                discard;
              }
          Rule 2: 0101010000000000000000000000000000000000000000000
              then {
                accept;
              }
```

Meaning The output shows the information for the dynamic profile, including Ascend-Data-Filter rules. Verify the following information:

- The User Name field indicates the correct subscriber.
- The Dynamic Profile Name field is correct for the subscriber.
- The correct Ascend-Data-Filter rules are applied to the subscriber. The display shows the rules that are configured on the RADIUS server.

Verifying Dynamic Ascend-Data-Filter Usage

Purpose Verify usage of the dynamic Ascend-Data-Filter. Counter statistics are displayed when the **counter** option is configured for the **adf** command in the dynamic profile.

Action From operational mode, enter the **show firewall** command.

```
user@host> show firewall

Filter: __junos_adf_5-ge-1/0/0.0-inet-in
Counters:
Name                                Bytes              Packets
t0-cnt                              32758                   22
t1-cnt                              22199                   15
t2-cnt                              21723                   14
```

Meaning The output shows the name of the filter and lists the counter activity. If the **counter** option is not configured, the output displays only the filter name.

Related Documentation

- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11
- Dynamically Applying Ascend-Data-Filter Policies to Subscriber Sessions on page 61

Example: Configuring Static Ascend-Data-Filter Support for Subscriber Access

This example shows how to configure support for static Ascend-Data-Filter policies. In a static configuration, you manually configure the Ascend-Data-Filter as part of the dynamic profile configuration. This procedure differs from dynamic configuration, in which the Ascend-Data-Filter is defined on the RADIUS server and then subscriber management uses a predefined variable to map the Ascend-Data-Filter rules to Junos OS filter functionality. Because creating a static Ascend-Data-Filter configuration can be labor-intensive, you might typically use this method for testing purposes.

Requirements

- Create the dynamic profile. See *Dynamic Profiles Overview*.
- Configure RADIUS support. See *Configuring RADIUS Server Parameters for Subscriber Access*.

Overview

Ascend-Data-Filters contain rules that create policies. Subscriber management uses a dynamic profile to apply the policy to a subscriber session. You manually configure the Ascend-Data-Filter as part of the dynamic policy.

- Specify the dynamic profile to use to apply the Ascend-Data-Filter policy to the subscriber session.
- Configure the Ascend-Data-Filter.
- Configure optional settings, which include counting the rule usage and setting the precedence for received and transmitted traffic.

Configuration

Step-by-Step Procedure

To configure static Ascend-Data-Filter support:

1. Specify the dynamic profile in which you want to create the Ascend-Data-Filter, and configure the interface, the logical unit number, and the family type.

   ```
   [edit]
   user@host# edit dynamic-profiles adf-profile-v4 interfaces
     $junos-interface-ifd-name unit $junos-underlying-interface-unit family inet
   ```

2. Configure the Ascend-Data-Filter. Enclose the filter values within quotation marks. You can configure multiple Ascend-Data-Filter rules in the same dynamic profile.

   ```
   [edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit” family inet]
   user@host# set filter adf rule “01000100 0A020100 00000000 18000000
     00000000 00000000”
   ```

3. Enable the counter for the rule.

   ```
   [edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit” family inet]
   user@host# set filter adf counter
   ```

4. Specify the precedence for received packets on the interface.

   ```
   [edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit” family inet]
   user@host# set filter adf input-precedence 80
   ```

5. Specify the precedence for transmitted packets on the interface.

   ```
   [edit dynamic-profiles adf-profile-v4 interfaces “$junos-interface-ifd-name” unit
     “$junos-underlying-interface-unit” family inet]
   user@host# set filter adf output precedence 85
   ```

Results

From configuration mode, confirm your configuration by entering the **show dynamic-profiles** command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show dynamic-profiles
```

```
...
adf-profile-v4 {
   interfaces {
      "$junos-interface-ifd-name" {
         unit "$junos-underlying-interface-unit" {
            family inet {
               filter {
                  adf {
                     rule "01000100 0A020100 00000000 18000000 00000000 00000000";
                     counter;
                     input-precedence 80;
                     output-precedence 85;
                  ...
```

If you are done configuring the device, enter **commit** from configuration mode.

Results

The Ascend-Data-Filter rule defined in Step 2 of the procedure configures an input policy that filters all packets from network 10.2.1.0 with wildcard mask 255.255.255.0 to any destination.

Table 8 on page 87 lists the values specified in the Ascend-Data-Filter rule.

Table 8: Ascend-Data-Filter Rule

Action or Classifier	Hex Value	Junos OS Filter Function
Type	01	IPv4
Forward	00	Forward
Indirection	01	Ingress
Spare	00	None
Source IP address	0a020100	10.2.1.0
Destination IP address	00000000	Any
Source IP mask	18	24 (255.255.255.0)
Destination IP mask	00	0 (0.0.0.0)
Protocol	00	None
Established	00	None
Source port	0000	None
Destination port	0000	None
Source port qualifier	00	None

Table 8: Ascend-Data-Filter Rule *(continued)*

Action or Classifier	Hex Value	Junos OS Filter Function
Destination port qualifier	00	None
Reserved	0000	None

Verification

To confirm that the configuration is working properly, perform these tasks:

- Verifying that Static Ascend-Data-Filter Rules are Applied to Subscriber Sessions on page 88
- Verifying Static Ascend-Data-Filter Usage on page 89

Verifying that Static Ascend-Data-Filter Rules are Applied to Subscriber Sessions

Purpose Verify that the Ascend-Data-Filter rules you manually configured were attached to the subscriber.

Action From operational mode, enter the **show subscribers extensive** command.

```
user@host>show subscriber extensive
  Type: DHCP
  User Name: user1-adf
  IP Address: 192.168.1.10
  IP Netmask: 255.255.255.0
  Logical System: default
  Routing Instance: default
  Interface: ge-1/0/0.0
  Interface type: Static
  Dynamic Profile Name: adf-profile-v4
  MAC Address: 00:10:94:00:00:01
  State: Active
  Radius Accounting ID: 5
  Login Time: 2010-08-12 14:06:27 PDT
  ADF IPv4 Input Filter Name: __junos_adf_5-ge-1/0/0.0-inet-in
            Rule 0: 010001000A020100000000001800000000000000000000000
                from {
                  destination-address 10.2.1.0/24;
                }
                then {
                  accept;
                }
```

Meaning The output shows the information for the dynamic profile, including Ascend-Data-Filter rules. Verify the following information:

- The User Name field indicates the correct subscriber.
- The Dynamic Profile Name field is correct for the subscriber.

- The correct static Ascend-Data-Filter rule is applied to the subscriber.

Verifying Static Ascend-Data-Filter Usage

Purpose Verify usage of the static Ascend-Data-Filter. Counter statistics are displayed when the **counter** option is configured for the **adf** command in the dynamic profile.

Action From operational mode, enter the **show firewall** command.

```
user@host> show firewall

Filter: __junos_adf_5-ge-1/0/0.0-inet-in
Counters:
Name                            Bytes              Packets
t0-cnt                          32758                   22
```

Meaning The output shows the name of the filter and the lists counter activity. If the **counter** option is not configured, the output displays only the filter name.

Related Documentation

- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11
- Dynamically Applying Ascend-Data-Filter Policies to Subscriber Sessions on page 61

Example: Configuring Fast Update Filters for Subscriber Access

This example shows you how to configure a fast update filter that is an input filter that counts the HTTP and non-HTTP packets from a subscriber. In the example, you use the firewall stanza to create the filter and the interfaces stanza to attach the filter.

```
[edit dynamic-profiles myProfile]
firewall {
   family inet {
      fast-update-filter httpFilter {
         interface-specific;
         match-order [source-address protocol destination-port];
         term term1 {
            from {
               protocol tcp;
               source-address $junos-subscriber-ip-address;
               destination-port http;
            }
            then {
               count http-cnt;
            }
         }
         term term2 {
            from {
               protocol tcp;
               source-address $junos-subscriber-ip-address;
            }
            then {
               count non-http-cnt;
            }
```

```
        }
      }
    }
  }
  interfaces {
    "$junos-interface-ifd-name" {
      unit "$junos-underlying-interface-unit" {
        family inet {
          filter {
            input httpFilter;
          }
        }
      }
    }
  }
```

Related Documentation

- Configuring Fast Update Filters on page 63

Example: Bypassing Firewall Filters

This example describes how to configure multiple filters using the **service-filter-hit** match/action combination and contains the following sections:

- Before You Begin on page 90
- Filter Bypass Overview on page 90
- Configuring Filter Bypass on page 91

Before You Begin

When using the **service-filter-hit** match/action combination, keep the following in mind:

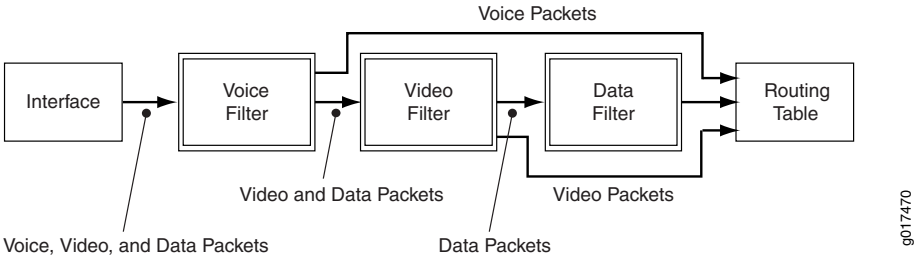
- The order in which the filters are applied is important. You can ensure the order in which the filters are processed by specifying a filter precedence value for the interface. See "Defining Dynamic Filter Processing Order" on page 57 for more information about dynamic filter processing and how to use the **precedence** statement.
- The following example uses policers to further define the match conditions each filter uses. These filters are not described here. To better understand how to configure policers, see "*Statement Hierarchy for Configuring Policers*" in the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*.

Filter Bypass Overview

Packets must pass through each filter in a chain. However, if you create a chain of filters to process different types of packets (for example, voice, video, and data packets), you can streamline the filter process, decreasing the amount of packet handling for each filter in the chain, effectively bypassing unnecessary filters, by using the **service-filter-hit** match/action combination at the **[edit *firewall family family-name* filter *filter-name term term-name*]** hierarchy level.

Figure 1 on page 91 shows the logical processing flow through a chain of three filters (voice, video, and data) where only processing for a specific data type is desired. This configuration example shows an ingress filter flow. Though subsequent ingress filters in a chain can detect whether the **service-filter-hit** action is set, egress filters do not. To bypass egress filters, you must also configure the **service-filter-hit** match/action combination on those filters.

Figure 1: Logical Flow Example for Filter Bypass Processing



Configuring Filter Bypass

CLI Quick Configuration

To quickly configure this example:

```
[edit]
set firewall filter voice term T1 from address 1.1.1.1/32
set firewall filter voice term T1 from source-port 5004-5005
set firewall filter voice term T1 then forwarding-class assured-forwarding service-filter-hit
    accept
set firewall filter voice term default then accept
set firewall filter video term T1 from service-filter-hit
set firewall filter video term T1 then accept
set firewall filter video term T2 from source-address 10.10.10.10/32
set firewall filter video term T2 then policer video-policer service-filter-hit accept
set firewall filter video term default then accept
set firewall filter data term T1 from service-filter-hit
set firewall filter data term T1 then accept
set firewall filter data term T2 then policer data-policer service-filter-hit accept
```

Configuring the Voice Filter

Step-by-Step Procedure

To configure the voice filter for the logical flow in Figure 1 on page 91:

1. Configure the filter to apply the assured forwarding class and set the **service-filter-hit** action for traffic from a specific address and port range (over which voice traffic is expected).

   ```
   [edit]
   set firewall filter voice term T1 from address 1.1.1.1/32
   set firewall filter voice term T1 from source-port 5004-5005
   ```

```
set firewall filter voice term T1 then forwarding-class assured-forwarding
  service-filter-hit accept
```

2. Configure the filter default action to pass (accept) packet traffic from any other address or port range.

```
[edit]
set firewall filter voice term default then accept
```

Configuring the Video Filter

Step-by-Step Procedure

To configure the video filter for the logical flow in Figure 1 on page 91:

1. Configure the filter to pass (accept) incoming packets that are tagged by the **service-filter-hit** action.

```
[edit]
set firewall filter video term T1 from service-filter-hit
set firewall filter video term T1 then accept
```

2. Configure the filter to apply a video policer and set the **service-filter-hit** action for traffic from a specific address (over which video traffic is expected).

```
[edit]
set firewall filter video term T2 from source-address 10.10.10.10/32
set firewall filter video term T2 then policer video-policer service-filter-hit accept
```

3. Configure the filter default action to pass (accept) packet traffic from any other address or port range.

```
[edit]
set firewall filter video term default then accept
```

Configuring the Data Filter

Step-by-Step Procedure

To configure the data filter for the logical flow in Figure 1 on page 91:

1. Configure the filter to pass (accept) incoming packets that are tagged by the **service-filter-hit** action.

```
[edit]
set firewall filter data term T1 from service-filter-hit
set firewall filter data term T1 then accept
```

2. Configure the filter to apply a data policer and set the **service-filter-hit** action for traffic from a specific address (over which video traffic is expected).

```
[edit]
set firewall filter data term T2 then policer data-policer service-filter-hit accept
```

Results

Display the results of the configuration:

```
[edit firewall]
user@host# show
filter voice {
```

```
    term T1 {
      from {
        address {
          1.1.1.1/32;
        }
        source-port 5004-5005;
      }
      then {
        forwarding-class assured-forwarding;
        service-filter-hit;
        accept;
      }
    }
    term default {
      then accept;
    }
  }
  filter video {
    term T1 {
      from {
        service-filter-hit;
      }
      then accept;
    }
    term T2 {
      from {
        source-address {
          10.10.10.10/32;
        }
      }
      then {
        policer video_policer;
        service-filter-hit;
        accept;
      }
    }
    term default {
      then accept;
    }
  }
  filter data {
    term T1 {
      from {
        service-filter-hit;
      }
      then accept;
    }
    term T2 {
      then {
        policer data_policer;
        service-filter-hit;
        accept;
      }
    }
  }
```

Related Documentation

- Classic Filters Overview on page 7
- Defining Dynamic Filter Processing Order on page 57
- *Statement Hierarchy for Configuring Policers*
- Configuring Firewall Filter Bypass on page 57

Example: Dynamic-Profile Parsing

The following example shows the basic dynamic-profile parsing steps for "Dynamic Profile Configuration and UID Substitution Comparison for Parameterized Filters" on page 25. The steps apply to any parameterized filter.

1. Read **dynamic-profiles my-svc-prof interface ge-1/0/0 unit 7 family inet filter input** and get the value **my-filt-1gw_UID1022**. The **my-in-filter** variable received the name of the UID (**my-filt-1gw**) from the first service parameter. The name **my-filt-1gw_UID1022** comes from the value of the **my-filt-1gw UID**.
2. Determine whether a static filter called **my-filt-1gw_UID1022** exists. If so, this is the existing classic filter case and not a parameterized filter.
3. Try to read **dynamic-profile my-svc-prof firewall family inet fast-update-filter my-filt-1gw_UID1022**'. If this exists, this is a fast update filter, not a parameterized filter.
4. Try to read **dynamic-profile my-svc-prof firewall family inet filter my-filt-1gw_UID1022**. If this does not exist, return a "filter not found" error.
5. Search for a template named **my-filt-1gw_UID1022**. If it does not exist:
   a. Read the parameterized filter configuration. This adds the match destination address **207.17.137.239** and the policer **my-policer_UID1005** as the action.
   b. Determine whether **my-policer_UID1005** exists. If it does not, read the **dynamic-profile my-svc-prof firewall policer my-policer_UID1005** configuration and create the **my-policer_UID1005** policer.
   c. Compile the **my-filt-1gw_UID1022** filter.
   d. Install **my-filt-1gw_UID1022** as a filter template.
6. Create and install an interface-specific filter reference named my-filt-1gw_UID1022-ge-1/0/0.7-in with **my-filt-1gw_UID1022** as its template.
7. Attach **my-filt-1gw_UID1022-ge-1/0/0.7-in** to interface **ge-1/0/0.7**.

When subsequent sessions are created with the same parameters, the system returns the same **my-filt-1gw_UID1022** filter name. In this case, Step 5 finds the existing filter template and proceeds directly to Step 6.

Related Documentation

- Sample Dynamic-Profile Configuration for Parameterized Filters on page 22
- Dynamic Profile After UID Substitutions for Parameterized Filters on page 24

Example: Configuring Hierarchical Policers as Filter Actions

This example shows how to configure a hierarchical policer and apply the policer to ingress Layer 3 traffic at a logical interface on the MX-series platform.

Requirements

Before you begin, be sure that your environment meets the following requirements:

- Supported on MX Series routers.

Overview

In this example, you configure a hierarchical policer as a filter action.

Configuration

Example: Hierarchical Policer as Filter Action

Step-by-Step Procedure

You can have hierarchical policers as one type of filter action. To configure a firewall filter:

1. Configure the family address type for a firewall filter:

   ```
   [edit firewall]
   user@host# set family inet
   ```

2. Specify the filter name:

   ```
   [edit firewall family inet]
   user@host# set filter inet-filter
   ```

3. Specify the term name:

   ```
   [edit firewall family inet filter inet-filter]
   user@host# set term t1
   ```

4. In each firewall filter term, specify the match conditions to use to match components of a packet:

   ```
   [edit firewall family inet filter inet-filter term t1]
   user@host# set from precedence critical-ecp immediate priority
   user@host# set from protocol tcp
   ```

5. In each firewall filter term, specify the actions to take if the packet matches all the condition in that term:

```
[edit firewall family inet filter inet-filter term t1]
user@host# set then hierarchical-policer HP1
```

6. (Optional) Enable all hierarchical policers in one filter to share the same policer instance in PFE:

```
[edit firewall family inet filter inet-filter term t1]
user@host# set then hierarchical-policer HP1 filter-specific
```

Results Confirm the configuration by entering the **show firewall** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter inet-filter {
    interface-specific;
    term t1 {
      from {
        precedence [ critical-ecp immediate priority ];
        protocol tcp;
      }
      then hierarchical-policer HP1;
    }
    term t2 {
      from {
        precedence [ internet-control routine ];
        protocol tcp;
      }
      then hierarchical-policer HP2;
    }
  }
}
family inet6 {
  filter inet6-filter {
    interface-specific;
    term t1 {
      from {
        next-header [ tcp udp ];
        forwarding-class [ assured-forwarding expedited-forwarding ];
      }
      then hierarchical-policer HP1;
    }
    term t2 {
      from {
        next-header [ tcp udp icmpv6 ospf rsvp ];
        forwarding-class [ network-control best-effort ];
      }
      then hierarchical-policer HP2;
    }
  }
}
```

```
}
```

Example: Defining the Interface:

Step-by-Step Procedure

To define the interface:

1. Enable configuration of the physical interface:

   ```
   [edit]
   user@host# edit interfaces ge-1/2/0 unit 0
   ```

2. Configure the family address:

   ```
   [edit interfaces ge-1/2/0 unit 0]
   user@host# set family inet address 10.100.16.2/24
   ```

3. Specify the filter name:

   ```
   [edit interfaces ge-1/2/0 unit 0 family inet]
   user@host# set filter inet-filter
   user@host# set address 10.100.16.2/24
   ```

Results

Confirm the configuration by entering the **show interfaces** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
interfaces {
  ge-1/2/0 {
    unit 0 {
      family inet {
        filter {
          input inet-filter;
        }
        address 10.100.16.2/24;
      }
      family inet6 {
        input-hierarchal-policer shared_HP;
        address 1A23:120B::7634:AD01:4D/120;
      }
    }
  }
  ge-1/2/1 {
    unit 0 {
      family inet {
        input-hierarchal-policer shared_HP;
        address 10.100.16.2/24;
      }
    }
  }
}
```

Verification

Confirm that the configuration is working properly.

- Displaying Packets for the Firewall on page 98

Displaying Packets for the Firewall

Purpose Verify the number of packets evaluated by the policer. Premium policer counters are not supported.

Action Use the **show firewall** operational mode command. The command output displays the number of packets.

```
[edit]
user@host# show firewall
```

```
Filter: __default_bpdu_filter__

Filter: utp_4550-ge-1/0/0.100-in
Counters:
Name                                                Bytes              Packets
c_ef-ge-1/0/0.0-i                                 1696750                15425
c_other-ge-1/0/0.0-i                                    0                    0
Policers:
Name                                                Packets
hp_abc-filter-ge-1/0/0.0-i                             7509
```

Related Documentation

- Hierarchical Policer Overview on page 36
- Hierarchical Policer as Filter Action on page 36
- filter-specific on page 149

Example: Interface-Shared Filter Configuration

Before you can attach an interface-shared filter using a dynamic profile.

- Create a basic dynamic profile.

  See *Configuring a Basic Dynamic Profile*.

To configure an interface-shared filter using a dynamic profile that is used to implement agent-circuit-identifier VLAN household filtering:

1. Access the dynamic profile you want to use.

   ```
   [edit]
   user@host# edit dynamic-profiles profile-name
   ```

2. Specify the interfaces.

   ```
   [edit dynamic-profiles profile-name]
   user@host# edit interfaces interface-name
   ```

3. Specify the unit.

[edit dynamic-profiles *profile-name* interfaces *interface-name*]
user@host# **edit unit $junos-interface-unit**

4. Specify the family.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit
   "$junos-interface-unit"]
   user@host# **edit family *family-name***

5. Specify the input filter and the filter terms for the interface unit.

   [edit dynamic-profiles *profile-name* interfaces *interface-name* unit
   "$junos-interface-unit" family *family-name*]
   user@host# **edit input $junos-input-filter shared-name $junos-interface-set-name precedence *precedence-number***

6. Specify the output filter and the filter terms for the interface unit.

   [edit dynamic-profiles *profile-name* interfaces*interface-name* unit
   "$junos-interface-unit" family *family-name*]
   user@host# **edit input $junos-output-filter shared-name $junos-interface-set-name precedence *precedence-number***

7. Specify that you want to configure a firewall, and specify the family.

   [edit dynamic-profiles *profile-name*]
   user@host# **edit firewall family *family-name***

8. Specify the filter.

   [edit dynamic-profiles *profile-name* firewall family *family-name*
   user@host# **edit** filter ***filter-name***

9. Specify the interface-shared filter.

   [edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*]
   user@host# **set interface-shared**

In the following example using an interface-shared filter, you configure a dynamic profile that is used to implement agent-circuit-identifier VLAN household filtering. If **$junos-input-filter** is FILTER1 and **$junos-interface-set-name** is ACI1, then a filter with the name FILTER1-ACI1-in is created and attached to the demux0 unit. When a subsequent login from the same household occurs, it is in the same VLAN. If **$junos-input-filter** is also FILTER1, the next demux0 interface also has the FILTER1-ACI1-in filter attached. A low value precedence was used with the interface-shared filter. If you want to have the interface-shared filter applied first, then you must give a higher precedence to any other filters that are attached to the same interfaces.

```
[edit]
  dynamic-profile {
  client-profile {
    interfaces {
      demux0 {
        unit $junos-interface-unit {
          family inet {
            filter {
              input $junos-input-filter shared-name $junos-interface-set-name precedence
                10;
            }
```

```
            }
          }
        }
      }
    }
    }
    firewall {
      family inet {
        filter FILTER1 {
          interface-shared;
          term... # the filter's terms
        }
      }
    }
```

Related Documentation

- Dynamically Attaching Statically Created Filters for a Specific Interface Family Type on page 53
- Dynamically Attaching Filters Using RADIUS Variables on page 55
- For information about firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Example: Configuring Unicast RPF in a Dynamic Profile on MX Series Routers

This example shows how to help defend the router ingress interfaces against denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks by configuring unicast reverse-path forwarding (RPF) on a customer-edge interface to filter incoming traffic. Unicast RPF verifies the unicast source address of each packet that arrives on an ingress interface where unicast RPF is enabled. Packets that fail verification are silently discarded unless a fail filter performs some other action on them.

Requirements

This example uses the following software and hardware components:

- Junos OS Release 12.3
- An MX Series 3D Universal Edge router

Before you begin:

- Configure the dynamic profile that you intend to use to apply the RPF check.

  See *Configuring a Basic Dynamic Profile*.

Overview

Large amounts of unauthorized traffic—such as attempts to flood a network with fake service requests in a denial-of-service (DoS) attack—can consume network resources and deny service to legitimate users. One way to help prevent DoS and distributed denial-of-service (DDoS) attacks is to verify that incoming traffic originates from legitimate network sources.

Unicast RPF helps ensure that a traffic source is legitimate (authorized) by comparing the source address of each packet that arrives on an interface to the forwarding-table entry for its source address. If the router uses the same interface that the packet arrived on to reply to the packet's source, this verifies that the packet originated from an authorized source, and the router forwards the packet. If the router does not use the same interface that the packet arrived on to reply to the packet's source, the packet might have originated from an unauthorized source, and the router discards the packet, or passes it to a fail filter.

The fail filter enables you to set criteria for packets you want to be passed in spite of failing the RPF check, such as DHCP packets, which are dropped by default.

On MX Series routers, you can configure unicast RPF in a dynamic profile to apply the configuration to one or more subscriber interfaces. See *Configuring Unicast RPF* for more information about the behavior and limitations of unicast RPF on MX Series routers.

In this example, you configure the router to protect against potential DoS and DDoS attacks from the Internet perpetrated through IPv4 packets arriving on dynamically created VLAN demux interfaces. The dynamic profile, vlan-demux-prof, establishes that VLAN demux interfaces are automatically created for subscribers. Unicast RPF is enabled on the dynamic interfaces by the rpf-check term.

By default, unicast RPF prevents Dynamic Host Configuration Protocol (DHCP) packets from being accepted on interfaces to which it applies. When DHCP packets are discarded, no new subscribers can be created by the dynamic profile. To enable interfaces to accept DHCP packets, you must apply a fail filter that properly sorts through the packets that fail the check and identifies the DHCP packets. In this example, you configure the **allow-dhcp** term in the filter **rpf-pass-dhcp**. This term matches, counts, and accepts IPv4 packets that are destined for the DHCP port and any address. The **default term** drops all other packets that fail the RPF check.

This example does not show all possible configuration choices.

Configuration

To enable unicast RPF with a fail filter in a dynamic profile, perform these tasks:

- Configuring the Dynamic Profile to Apply RPF Checking to Dynamic VLAN Demux Interfaces on page 102
- Configuring the RPF-Check Fail Filter on page 103

Configuring the Dynamic Profile to Apply RPF Checking to Dynamic VLAN Demux Interfaces

CLI Quick Configuration

To quickly configure the dynamic profile to apply unicast RPF to dynamically created VLAN demux interfaces, copy the following commands, paste them in a text file, remove any line breaks, and then copy and paste the commands into the CLI.

```
edit dynamic-profiles vlan-demux-prof interfaces demux0
edit unit $junos-interface-unit
set demux-options underlying-interface $junos-interface-ifd-name
set vlan-id $junos-vlan-id
edit family inet
set unnumbered-address lo0.0
set rpf-check fail-filter rpf-pass-dhcp
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode*.

To configure unicast RPF on the router:

1. Create a dynamic profile.

   ```
   [edit]
   user@host# edit dynamic-profiles vlan-demux-prof
   ```

2. Specify that the dynamic VLAN profile use the demux interface.

   ```
   [edit dynamic-profiles vlan-demux-prof]
   user@host# edit interfaces demux0
   ```

3. Specify that the dynamic profile applies the demux interface unit value to the dynamic VLANs.

   ```
   [edit dynamic-profiles vlan-demux-prof interfaces demux0]
   user@host# edit unit $junos-interface-unit
   ```

4. Specify the logical underlying interface for the dynamic VLANs.

   ```
   [edit dynamic-profiles vlan-demux-prof interfaces demux0 unit $junos-interface-unit]
   user@host# set demux-options underlying-interface $junos-interface-ifd-name
   ```

5. Configure the variable that results in dynamically created VLAN IDs.

   ```
   [edit dynamic-profiles vlan-demux-prof interfaces demux0 unit $junos-interface-unit]
   user@host# set vlan-id $junos-vlan-id
   ```

6. Configure the IPv4 address family for the demux interfaces.

   ```
   [edit dynamic-profiles vlan-demux-prof interfaces demux0 unit $junos-interface-unit]
   ```

```
user@host# edit family inet
```

7. Configure the unnumbered address for the family.

```
[edit dynamic-profiles vlan-demux-prof interfaces demux0 unit $junos-interface-unit
  family inet]
user@host# set unnumbered-address lo0.0
```

8. Configure unicast RPF and specify the fail filter that is applied to incoming packets that fail the check.

```
[edit dynamic-profiles vlan-demux-prof interfaces demux0 unit $junos-interface-unit
  family inet]
user@host# set fail-filter fail-filter rpf-pass-dhcp
```

Configuring the RPF-Check Fail Filter

CLI Quick Configuration

To quickly configure the unicast RPF-check fail filter, copy the following commands, paste them in a text file, remove any line breaks, and then copy and paste the commands into the CLI.

```
edit firewall family inet filter rpf-pass-dhcp
edit term allow-dhcp
set from destination-port dhcp
set from destination-address 255.255.255.255/32
set then count rpf-dhcp-traffic
set then accept
up
edit term default
set then discard
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode*.

To configure the RPF-check fail filter:

1. Create the fail filter.

```
[edit firewall]
user@host# edit family inet filter rpf-pass-dhcp
```

2. Define the filter term that identifies DHCP packets based on the DHCP destination port, then counts and passes the packets.

```
[edit firewall family inet filter rpf-pass-dhcp]
user@host# edit term allow-dhcp
user@host# set from destination-port dhcp
user@host# set from destination-address 255.255.255.255/32
user@host# set then count rpf-dhcp-traffic
user@host# set then accept
```

3. Define the filter term that drops all other failed packets.

```
[edit firewall filter rpf-pass-dhcp]
user@host# edit term default
user@host# set then discard
```

Results From configuration mode, confirm the unicast RPF configuration by entering the **show dynamic-profiles** command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show dynamic-profiles
vlan-demux-prof {
  interfaces {
    demux0 {
      unit "$junos-interface-unit" {
        vlan-id "$junos-vlan-id";
        demux-options {
          underlying-interface "$junos-interface-ifd-name";
        }
        family inet {
          unnumbered-address lo0.0;
          rpf-check {
            fail-filter rpf-pass-dhcp;
          }
        }
      }
    }
  }
}
```

From configuration mode, confirm the fail filter configuration by entering the **show firewall** command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show firewall
family inet {
  filter rpf-pass-dhcp {
    term allow-dhcp {
      from {
        destination-address {
          255.255.255.255/32;
        }
        destination-port dhcp;
      }
      then {
        count rpf-dhcp-traffic;
        accept;
      }
    }
    term default {
      then {
        discard;
      }
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is correct, perform these tasks:

- Verifying That Unicast RPF Is Enabled on the Router on page 105

Verifying That Unicast RPF Is Enabled on the Router

Purpose Verify that unicast RPF is enabled.

Action Verify that unicast RPF is enabled by using the **show subscribers extensive** command.

```
user@host> show subscribers extensive
Type: VLAN
  Logical System: default
  Routing Instance: default
  Interface: ae0.1073741824
  Interface type: Dynamic
  Dynamic Profile Name: vlan-demux-prof
  State: Active
  Session ID: 9
  VLAN Id: 100
  Login Time: 2011-08-26 08:17:00 PDT
  IPv4 rpf-check Fail Filter Name: rpf-pass-dhcp
```

Meaning The IPv4 rpf-check Fail Filter Name field displays **rpf-pass-dhcp**, the name of the fail filter applied by the dynamic profile for IPv4 packets failing the RPF check.

Related Documentation

- Unicast RPF in Dynamic Profiles for Subscriber Interfaces on page 49
- Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces on page 75
- *Configuring a Basic Dynamic Profile*

CHAPTER 11

Configuration Statements

[edit dynamic-profiles] Hierarchy Level

```
dynamic-profiles {
  profile-name {
    class-of-service {
      interfaces {
        interface-name {
          unit logical-unit-number {
            classifiers {
              type (classifier-name | default);
            }
            output-traffic-control-profile (profile-name | $junos-cos-traffic-control-profile);
            rewrite-rules {
              dscp (rewrite-name | default);
              dscp-ipv6 (rewrite-name | default);
              ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
              inet-precedence (rewrite-name | default);
              }
            }
          }
        }
      }
      scheduler-maps {
        map-name {
          forwarding-class class-name scheduler scheduler-name;
        }
      }
      schedulers {
        (scheduler-name) {
          buffer-size (percent percentage | remainder | temporal microseconds |
            $junos-cos-scheduler-bs);
          drop-profile-map loss-priority (any | low | medium-low | medium-high | high)
            protocol (any | non-tcp | tcp) drop-profile (profile-name | predefined-variable);
          excess-priority (low | high | $junos-cos-scheduler-excess-priority);
          excess-rate (percent percentage  | percent $junos-cos-scheduler-excess-rate);
          overhead-accounting (shaping-mode) <bytes (byte-value>;
          priority (priority-level | $junos-cos-scheduler-priority);
          shaping-rate (rate | predefined-variable);
```

```
        transmit-rate (rate | percent percentage | remainder | percent percentage
          $junos-cos-scheduler-tx) <exact | rate-limit>;
      }
    }
    traffic-control-profiles profile-name {
      delay-buffer-rate (percent percentage | rate);
      excess-rate (percent percentage | proportion value | percent
        $junos-cos-excess-rate);
      guaranteed-rate (percent percentage | rate);
      overhead-accounting (shaping-mode) <bytes (byte-value>;
      scheduler-map map-name;
      shaping-rate (percent percentage | rate | predefined-variable);
    }
  }
  firewall {
    family family {
      fast-update-filter filter-name {
        interface-specific;
      match-order [match-order];
        term term-name {
          from {
            match-conditions;
          }
          then {
            action;
            action-modifiers;
          }
        only-at-create;
      filter filter-name {
      interface-specific;
        term term-name {
          from {
            match-conditions;
          }
          then {
            action;
            action-modifiers;
          }
        }
    policer policer-name {
      filter-specific;
      if-exceeding {
        (bandwidth-limit bps | bandwidth-percent percentage);
        burst-size-limit bytes;
      }
      logical-bandwidth-policer;
      logical-interface-policer;
      physical-interface-policer;
      then {
        policer-action;
      }
    }
    hierarchical-policer policer-name {
      aggregate {
        if-exceeding {
          bandwidth-limit-limit bps;
```

```
            burst-size-limit bytes;
          }
          then {
            policer-action;
          }
        }
        premium {
          if-exceeding {
            bandwidth-limit bps;
            burst-size-limit bytes;
          }
          then {
            policer-action;
          }
        }
      }
      three-color-policer policer-name {
        action {
          loss-priority high then discard;
        }
        logical-interface-policer;
        single-rate {
          (color-aware | color-blind);
          committed-burst-size bytes;
          committed-information-rate bps;
          excess-burst-size bytes;
        }
        two-rate {
          (color-aware | color-blind);
          committed-burst-size bytes;
          committed-information-rate bps;
          peak-burst-size bytes;
          peak-information-rate bps;
          }
        }
      }
}
policy-options {
  prefix-listname {
    ip-addresses;
  }
}
interfaces {
  interface-name {
    unit logical-unit-number {
      family family {
        access-concentrator name;
        address address;
        duplicate-protection;
        dynamic-profile profile-name;
        filter {
          adf {
            counter;
            input-precedence precedence;
            not-mandatory;
            output-precedence precedence;
```

```
            rule rule-value;
          }
          input filter-name {
            precedence precedence;
            shared-name filter-shared-name;
          }
          output filter-name {
            precedence precedence;
            shared-name filter-shared-name;
          }
        }
        max-sessions number;
        max-sessions-vsa-ignore;
        rpf-check {
          fail-filter filter-name;
          mode loose;
        }
        service {
          input {
            service-set service-set-name {
              service-filter filter-name;
            }
            post-service-filter filter-name;
          }
          output {
            service-set service-set-name {
              service-filter filter-name;
            }
          }
        }
        service-name-table table-name;
        short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
          maximum-seconds>;
        unnumbered-address interface-name <preferred-source-address address>;
      }
      ppp-options {
        chap;
        pap;
      }
      vlan-id number;
    }
    vlan-tagging;
  }
  interface-set interface-set-name {
    interface interface-name {
      unit logical-unit-number;
    }
  }
  demux0 {
    unit logical-unit-number {
      demux-options {
        underlying-interface interface-name
      }
      demux-source {
        source-prefix;
      }
```

```
      family family {
        access-concentrator name;
        address address;
        duplicate-protection;
        dynamic-profile profile-name;
        filter {
          input filter-name;
          output filter-name;
        }
        mac-validate (loose | strict):
        max-sessions number;
        max-sessions-vsa-ignore;
        service-name-table table-name;
        short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
          maximum-seconds>;
        unnumbered-address interface-name <preferred-source-address address>;
      }
    }
  }
  pp0 {
    unit logical-unit-number {
      keepalives interval seconds;
      no-keepalives;
      pppoe-options {
        underlying-interface interface-name;
        server;
      }
      ppp-options {
        authentication [ authentication-protocols ];
        chap {
          challenge-length minimum minimum-length maximum maximum-length;
        }
        pap;
      }
      family inet {
        unnumbered-address interface-name;
        address address;
        service {
          input {
            service-set service-set-name {
              service-filter filter-name;
            }
            post-service-filter filter-name;
          }
          output {
            service-set service-set-name {
              service-filter filter-name;
            }
          }
        }
        filter {
          input filter-name {
            precedence precedence;
          }
          output filter-name {
            precedence precedence;
```

```
            }
          }
        }
      }
    }
}
protocols {
    igmp {
        interface interface-name {
            accounting;
            disable;
            group-policy;
            immediate-leave
            no-accounting;
            promiscuous-mode;
            ssm-map ssm-map-name;
            static {
                group group {
                    source source;
                }
            }
            version version;
        }
        mld {
            interface interface-name {
                disable;
                (accounting | no-accounting);
                group-policy;
                immediate-leave;
                oif-map;
                passive;
                ssm-map ssm-map-name;
                static {
                    group multicast-group-address {
                        exclude;
                        group-count number;
                        group-increment increment;
                        source ip-address {
                            source-count number;
                            source-increment increment;
                        }
                    }
                }
                version version;
            }
        }
        router-advertisement {
            interface interface-name {
                current-hop-limit number;
                default-lifetime seconds;
                (managed-configuration | no-managed-configuration);
                max-advertisement-interval seconds;
                min-advertisement-interval seconds;
                (other-stateful-configuration | no-other-stateful-configuration);
                prefix prefix {
                    (autonomous | no-autonomous);
```

```
                (on-link | no-on-link);
                preferred-lifetime seconds;
                valid-lifetime seconds;
              }
              reachable-time milliseconds;
              retransmit-timer milliseconds;
            }
          }
        }
      }
    }
    routing-instances routing-instance-name {
      interface interface-name;
      routing-options {
        access {
          route prefix {
            next-hop next-hop;
            metric route-cost;
            preference route-distance;
            tag route-tag;
          }
        }
        access-internal {
          route subscriber-ip-address {
            qualified-next-hop underlying-interface {
              mac-address address;
            }
          }
        }
        multicast {
          interface interface-name {
            no-qos-adjust;
          }
        }
      }
      rib routing-table-name {
        access {
          route prefix {
            next-hop next-hop;
            metric route-cost;
            preference route-distance;
            tag route-tag;
          }
        }
        access-internal {
          route subscriber-ip-address {
            qualified-next-hop underlying-interface {
              mac-address address;
            }
          }
        }
      }
    }
    routing-options {
      access {
        route prefix {
```

```
            next-hop next-hop;
            metric route-cost;
            preference route-distance;
            tag route-tag;
          }
        }
        access-internal {
          route subscriber-ip-address {
            qualified-next-hop underlying-interface {
              mac-address address;
            }
          }
        }
        multicast {
          interface interface-name {
            no-qos-adjust;
          }
        }
      }
      variables {
        variable-name {
          default-value default-value;
          equals expression;
          mandatory;
          radius {
            vendor-id id {
              attribute attribute-number;
              tag tag-number;
            }
            redirect-url
          }
          uid;
          uid-reference;
        }
      }
    }
```

Related Documentation

- *Dynamic Profiles Overview*
- *CoS for Subscriber Access Overview*
- *Configuring a Basic Dynamic Profile*
- *Configuring Static Hierarchical Scheduling and Queuing in a Dynamic Profile for Subscriber Access*
- *Two-Color Policer Configuration Overview*
- *Three-Color Policer Configuration Overview*
- *Hierarchical Policer Configuration Overview*
- *Guidelines for Applying Traffic Policers*

action

Syntax

```
action {
  loss-priority high then discard;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *name*]

Release Information

Statement introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Discard traffic on a logical interface using tricolor marking policing.

NOTE: This statement is supported only on IQ2 interfaces.

The remaining statement is explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Basic Single-Rate Three-Color Policers*
- *Basic Two-Rate Three-Color Policers*
- *Two-Color and Three-Color Logical Interface Policers*
- *Two-Color and Three-Color Physical Interface Policers*
- *Two-Color and Three-Color Policers at Layer 2*
- loss-priority high then discard on page 162

adf (Dynamic Firewalls)

Syntax

```
adf {
  counter;
  input-precedence precedence;
  not-mandatory;
  output-precedence precedence;
  rule rule-value;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter]

Release Information

Statement introduced in Junos OS Release 10.4.
Option **not-mandatory** introduced in Junos OS Release 12.2.

Description

Configure an Ascend-Data-Filter that the dynamic profile applies to a subscriber session.

Options

counter—Enable a counter that increments each time the Ascend-Data-Filter rule is used. Typically used for testing purposes.

not-mandatory—Suppress router from reporting an error when the RADIUS reply message does not include the $junos-adf-rule-v4 or $junos-adf-rule-v6 variable that is configured for the Ascend-Data-Filter in the dynamic profile. In this circumstance, the Ascend-Data-Filter is not created.

precedence—Precedence value that sets the order in which dynamic service filters are applied on the interface. The lower the precedence value, the higher the precedence that is given. The precedence setting is used in conjunction with the precedence settings of all dynamic service filters configured (not only Ascend-Data-Filters) on the same interface to establish the order. For example, the order also includes any configured **input *filter-name* precedence *precedence*** and **output *filter-name* precedence *precedence*** statements.

Range: 0 through 255

Default: 0

rule-value—Ascend-Data-Filter rule. You can specify either a Junos predefined variable that maps the Ascend-Data-Filter actions to Junos filter functionality or you can manually configure the Ascend-Data-Filter rule. The router supports two predefined variables depending on family type: **$junos-adf-rule-v4** for family **inet** and **$junos-adf-rule-v6** for family **inet6**.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Basic Classic Filter Syntax on page 10

- For general information about configuring firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*.

aggregate (Hierarchical Policer)

Syntax

```
aggregate {
  if-exceeding {
    bandwidth-limit bandwidth;
    burst-size-limit burst;
  }
  then {
    discard;
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* firewall hierarchical-policer *name*],
[edit firewall hierarchical-policer]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... hierarchical-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description On M40e, M120, and M320 edge routers with Flexible PIC Concentrator (FPC) input as FFPC and FPC output as SFPC, and on MX Series, T320, T640, and T1600 edge routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, configure an aggregate hierarchical policer.

The remaining statements are explained separately.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Hierarchical Policer Configuration Overview*
- *Hierarchical Policers*
- *bandwidth-limit (Hierarchical Policer)*
- burst-size-limit (Hierarchical Policer) on page 122
- hierarchical-policer on page 151
- if-exceeding (Hierarchical Policer) on page 153
- premium on page 174

bandwidth-limit (Policer)

Syntax bandwidth-limit *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description For a single-rate two-color policer, configure the bandwidth limit as a number of bits per second. Single-rate two-color policing uses the single token bucket algorithm to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority (PLP) and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: This statement specifies the bandwidth limit as an absolute number of bits per second. Alternatively, for single-rate two-color policers only, you can use the **bandwidth-percent** *percentage* statement to specify the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options *bps*—You can specify the number of bits per second either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: (M Series, MX Series, and T Series routers) 8000 through 100,000,000,000

Default: None.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Two-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Single Token Bucket Algorithm*
- *Determining Proper Burst Size for Traffic Policers*
- bandwidth-percent on page 120
- burst-size-limit (Policer) on page 123

bandwidth-percent

Syntax bandwidth-percent *percentage*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description For a single-rate two-color policer, configure the bandwidth limit as a percentage value. Single-rate two-color policing uses the *single token bucket algorithm* to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: This statement specifies the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate. Alternatively, you can use the **bandwidth-limit** *bps* statement to specify the bandwidth limit as an absolute number of bits per second.

The function of the bandwidth limit is extended by the burst size (configured using the **burst-size-limit** *bytes* statement) to allow bursts of traffic up to a limit based on the overall traffic load:

- When a single-rate two-color policer is applied to the input or output traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified by this statement.
- During periods of relatively low traffic (traffic that arrives at or departs from the interface at overall rates below the token arrival rate), unused tokens accumulate in the bucket, but only up to the configured token bucket depth.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower

priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options

percentage—Traffic rate as a percentage of either the physical interface media rate or the logical interface configured shaping rate. You can configure a shaping rate on a logical interface by using class-of-service statement.

NOTE: You cannot rate-limit based on bandwidth percentage for aggregate, tunnel, and software interfaces. The bandwidth percentage policer cannot be used for forwarding table filters. Bandwidth percentage policers can only be used for interface-specific filters.

Range: 0 through 100

Default: None.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Two-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Single Token Bucket Algorithm*
- *Determining Proper Burst Size for Traffic Policers*
- *Bandwidth Policers*
- bandwidth-limit (Policer) on page 118
- burst-size-limit (Policer) on page 123

burst-size-limit (Hierarchical Policer)

Syntax burst-size-limit *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall hierarchical-policer aggregate if-exceeding],
[editdynamic-profiles *profile-name* firewall hierarchical-policer premium if-exceeding],
[edit firewall hierarchical-policer aggregate if-exceeding],
[edit firewall hierarchical-policer premium if-exceeding]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... if exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description For M40e, M120, and M320 (with FFPC and SFPC) edge routers and T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs,T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, configure the burst-size limit for premium or aggregate traffic in a hierarchical policer.

Options *bytes*—Burst-size limit in bytes. The minimum recommended value is the maximum transmission unit (MTU) of the IP packets being policed. You can specify the value either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 2,147,450,880

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Hierarchical Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Single Token Bucket Algorithm*
- *Determining Proper Burst Size for Traffic Policers*
- *Hierarchical Policers*
- aggregate (Hierarchical Policer) on page 117
- *bandwidth-limit (Hierarchical Policer)*
- premium (Hierarchical Policer) on page 174

burst-size-limit (Policer)

Syntax burst-size-limit *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description For a single-rate two-color policer, configure the burst size as a number of bytes. The burst size allows for short periods of traffic bursting (back-to-back traffic at average rates that exceed the configured bandwidth limit). Single-rate two-color policing uses the *single token bucket algorithm* to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

The burst size extends the function of the bandwidth limit (configured using either the **bandwidth-limit** *bps* statement or the **bandwidth-percent** *percentage* statement) to allow bursts of traffic up to a limit based on the overall traffic load:

- When a single-rate two-color policer is applied to the input or output traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified by this statement.
- During periods of relatively low traffic (traffic that arrives at or departs from the interface at overall rates below the token arrival rate), unused tokens accumulate in the bucket, but only up to the configured token bucket depth.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Table 9 on page 124 summarizes the relationship between the **bandwidth-limit** and the token arrival rate. This information is useful in calculating the minimum **burst-size-limit**.

Table 9: Bandwidth Limits and Token Rates

Bandwidth Limit	Token Rate
0-333 Mbps	low (262 µs)
334-666 Mbps	high (8.2 µs)
667-1333 Mbps	low
1334 Mbps and above	high

The burst-size limit enforced is based on the burst-size limit you configure. For a rate-limited logical interface, the Packet Forwarding Engine calculates the optimum burst-size-limit values and then applies the value closest to the burst-size-limit value specified in the policer configuration.

On MX Series routers, the burst-size limit is not as freely configurable as it is on other platforms. Junos OS does not support an unlimited combination of policer bandwidth and burst-size limits on MX Series routers. For a single-rate two-color policer on an MX Series router, the minimum supported burst-size limit is equivalent to the amount of traffic allowed by the policer bandwidth limit in a time span of 1 millisecond. For example, for a policer configured with a **bandwidth-limit** value of 1 Gbps, the minimum supported value for **burst-size-limit** on an MX Series router is 125 KB. If you configure a value that is smaller than the minimum, Junos OS overrides the configuration and applies the actual minimum.

Options

bytes—Burst-size limit in bytes. The minimum recommended value is the maximum transmission unit (MTU) of the IP packets being policed. You can specify the value either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000

Default: None

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Two-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Single Token Bucket Algorithm*
- *Determining Proper Burst Size for Traffic Policers*
- bandwidth-limit (Policer) on page 118
- bandwidth-percent on page 120

color-aware

Syntax color-aware;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.

Description For a three-color policer, configure the way preclassified packets are metered. In color-aware mode, the local router can assign a higher packet loss priority, but cannot assign a lower packet loss priority.

For example, suppose an upstream router assigned medium-high packet loss priority to a packet because the packet exceeded the committed information rate on the upstream router interface.

- If the local router applies color-aware policing to the packet, the router *cannot* change the packet loss priority to low, even if the packet conforms to the configured committed information route on the local router interface.
- If the local router applies color-blind policing to the packet, the router *can* change the packet loss priority to low if the packet conforms to the configured committed information route on the local router interface.

NOTE: **A color-aware policer cannot be applied to Layer 2 traffic.**

Default If you omit the **color-aware** statement, the default behavior is color-aware mode.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Color Modes for Three-Color Policers*
- color-blind on page 127

color-blind

Syntax color-blind;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.

Description For a three-color policer, configure the way preclassified packets are metered. In color-blind mode, the local router ignores the preclassification of packets and can assign a higher or lower packet loss priority.

For example, suppose an upstream router assigned medium-high packet loss priority to a packet because the packet exceeded the committed information rate on the upstream router interface.

- If the local router applies color-aware policing to the packet, the router *cannot* change the packet loss priority to low, even if the packet conforms to the configured committed information route on the local router interface.

NOTE: A color-aware policer cannot be applied to Layer 2 traffic.

- If the local router applies color-blind policing to the packet, the router *can* change the packet loss priority to low if the packet conforms to the configured committed information route on the local router interface.

Default If you omit the **color-blind** statement, the default behavior is color-aware mode.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Color Modes for Three-Color Policers*
- color-aware on page 126

committed-burst-size

Syntax committed-burst-size *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.

Description For a three-color policer, configure the committed burst size (CBS) as a number of bytes.

NOTE: When you include the **committed-burst-size** statement in the configuration, you must also include the **committed-information-rate** statement at the same hierarchy level.

In three-color policing, a committed information rate (CIR) defines the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions. A flow of traffic at an average rate that conforms to the CIR is categorized green.

During periods of average traffic rates below the CIR, any unused bandwidth capacity accumulates up to a maximum amount defined by the CBS. Short periods of bursting traffic (back-to-back traffic at averages rates that exceed the CIR) are also categorized as green provided that unused bandwidth capacity is available.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policers use a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red, based on the **excess-burst-size** statement included in the policer configuration.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits. Nonconforming traffic is categorized as yellow or red based on the **peak-information-rate** and **peak-burst-rate** statements included in the policer configuration.

Options *bytes*—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Dual Token Bucket Algorithms*
- *Determining Proper Burst Size for Traffic Policers*
- committed-information-rate on page 130
- excess-burst-size on page 140
- peak-burst-size on page 165
- peak-information-rate on page 167

committed-information-rate

Syntax committed-information-rate *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.

Description For a three-color policer, configure the committed information rate as a number of bits per second. The committed information rate (CIR) is the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions.

NOTE: When you include the **committed-information-rate** statement in the configuration, you must also include the **committed-burst-size** statement at the same hierarchy level.

In three-color policing, a CIR defines the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions. A flow of traffic at an average rate that conforms to the CIR is categorized green.

During periods of average traffic rates below the CIR, any unused bandwidth capacity accumulates up to a maximum amount defined by the committed burst size (CBS). Short periods of bursting traffic (back-to-back traffic at averages rates that exceed the CIR) are also categorized as green provided that unused bandwidth capacity is available.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policers use a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red, based on the **excess-burst-size** statement included in the policer configuration.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits. Nonconforming traffic is categorized as yellow or red based on the **peak-information-rate** and **peak-burst-rate** statements included in the policer configuration.

Options *bps*—Number of bits per second. You can specify a value in bits per second either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bps

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Dual Token Bucket Algorithms*
- *Determining Proper Burst Size for Traffic Policers*
- committed-burst-size on page 128
- excess-burst-size on page 140
- peak-burst-size on page 165
- peak-information-rate on page 167

dynamic-profiles

Syntax

```
dynamic-profiles {
  profile-name {
    class-of-service {
      interfaces {
        interface-name ;
        }
        unit logical-unit-number {
          classifiers {
            type (classifier-name | default);
          }
          output-traffic-control-profile (profile-name | $junos-cos-traffic-control-profile);
          rewrite-rules {
            dscp (rewrite-name | default);
            dscp-ipv6 (rewrite-name | default);
            ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
            inet-precedence (rewrite-name | default);
            }
          }
        }
      }
      scheduler-maps {
        map-name {
          forwarding-class class-name scheduler scheduler-name;
        }
      }
      schedulers {
        (scheduler-name) {
          buffer-size (seconds | percent percentage | remainder | temporal microseconds);
          drop-profile-map loss-priority (any | low | medium-low | medium-high | high)
            protocol (any | non-tcp | tcp) drop-profile profile-name;
          excess-priority (low | high | $junos-cos-scheduler-excess-priority);
          excess-rate (percent percentage | percent $junos-cos-scheduler-excess-rate);
          overhead-accounting (shaping-mode) <bytes (byte-value>;
          priority priority-level;
          shaping-rate (rate | predefined-variable);
          transmit-rate (percent percentage | rate | remainder) <exact | rate-limit>;
        }
      }
      traffic-control-profiles profile-name {
        delay-buffer-rate (percent percentage | rate | $junos-cos-delay-buffer-rate);
        excess-rate (percent percentage | proportion value | percent $junos-cos-excess-rate);
        guaranteed-rate (percent percentage | rate | $junos-cos-guaranteed-rate);
        overhead-accounting (shaping-mode) <bytes (byte-value>;
        scheduler-map map-name;
        shaping-rate (rate | predefined-variable);
      }
    }
    firewall {
      family family {
        fast-update-filter filter-name {
          interface-specific;
          match-order [match-order];
```

```
    term term-name {
      from {
        match-conditions;
      }
      then {
        action;
        action-modifiers;
      }
      only-at-create;
    }
  }
  firewall {
    family family {
      fast-update-filter filter-name {
        interface-specific;
      match-order [match-order];
        term term-name {
          from {
            match-conditions;
          }
          then {
            action;
            action-modifiers;
          }
        only-at-create;
      filter filter-name {
      interface-specific;
        term term-name {
          from {
            match-conditions;
          }
          then {
            action;
            action-modifiers;
          }
        }
      policer policer-name {
        filter-specific;
        if-exceeding {
          (bandwidth-limit bps | bandwidth-percent percentage);
          burst-size-limit bytes;
        }
        logical-bandwidth-policer;
        logical-interface-policer;
        physical-interface-policer;
        then {
          policer-action;
        }
      }
      hierarchical-policer policer-name {
        aggregate {
          if-exceeding {
            bandwidth-limit-limit bps;
            burst-size-limit bytes;
          }
          then {
```

```
                policer-action;
              }
            }
            premium {
              if-exceeding {
                bandwidth-limit bps;
                burst-size-limit bytes;
              }
              then {
                policer-action;
              }
            }
          }
          three-color-policer policer-name {
            action {
              loss-priority high then discard;
            }
            logical-interface-policer;
            single-rate {
              (color-aware | color-blind);
              committed-burst-size bytes;
              committed-information-rate bps;
              excess-burst-size bytes;
            }
            two-rate {
              (color-aware | color-blind);
              committed-burst-size bytes;
              committed-information-rate bps;
              peak-burst-size bytes;
              peak-information-rate bps;
              }
            }
          }
        }
        policy-options {
          prefix-listname {
            ip-addresses;
          }
        }
      }
    }
    interfaces interface-name {
      interface-set interface-set-name {
        interface interface-name {
          unit logical unit number {
            advisory-options {
              downstream-rate rate;
              upstream-rate rate;
            }
          }
        }
      }
      unit logical-unit-number {
        auto-configure {
          agent-circuit-identifier {
            dynamic-profile profile-name;
```

```
    }
  }
  encapsulation (atm-ccc-cell-relay | atm-ccc-vc-mux | atm-cisco-nlpid |
    atm-tcc-vc-mux | atm-mlppp-llc | atm-nlpid | atm-ppp-llc | atm-ppp-vc-mux |
    atm-snap | atm-tcc-snap | atm-vc-mux | ether-over-atm-llc |
    ether-vpls-over-atm-llc | ether-vpls-over-fr | ether-vpls-over-ppp | ethernet |
    frame-relay-ccc | frame-relay-ppp | frame-relay-tcc | frame-relay-ether-type |
    frame-relay-ether-type-tcc | multilink-frame-relay-end-to-end | multilink-ppp |
    ppp-over-ether | ppp-over-ether-over-atm-llc | vlan-bridge | vlan-ccc | vlan-vci-ccc
    | vlan-tcc | vlan-vpls);
  family family {
    address address;
    filter {
      adf {
        counter;
        input-precedence precedence;
        not-mandatory;
        output-precedence precedence;
        rule rule-value;
      }
      input filter-name (
        precedence precedence;
      }
      output filter-name {
        precedence precedence;
      }
    }
    rpf-check {
      fail-filter filter-name;
      mode loose;
    }
    service {
      input {
        service-set service-set-name {
          service-filter filter-name;
        }
        post-service-filter filter-name;
      }
      input-vlan-map {
        inner-tag-protocol-id tpid;
        inner-vlan-id number;
        (push | swap);
        tag-protocol-id tpid;
        vlan-id number;
      }
      output {
        service-set service-set-name {
          service-filter filter-name;
        }
      }
      output-vlan-map {
        inner-tag-protocol-id tpid;
        inner-vlan-id number;
        (pop | swap);
        tag-protocol-id tpid;
        vlan-id number;
```

```
            }
          }
          unnumbered-address interface-name <preferred-source-address address>;
        }
        ppp-options {
          chap;
          pap;
        }
        vlan-id number;
        vlan-tags outer [tpid].vlan-id [inner [tpid].vlan-id];
      }
    }
    interfaces {
      demux0 {...}
    }
    interfaces {
      pp0 {...}
    }
    protocols {
      igmp {
        interface interface-name {
          accounting;
          disable;
          group-policy;
          immediate-leave
          no-accounting;
          promiscuous-mode;
          ssm-map ssm-map-name;
          static {
            group group {
              source source;
            }
          }
          version version;
        }
        mld {
          interfaceinterface-name {
            disable;
            (accounting | no-accounting);
            group-policy;
            immediate-leave;
            oif-map;
            passive;
            ssm-map ssm-map-name;
            static {
              group multicast-group-address {
                exclude;
                group-count number;
                group-increment increment;
                source ip-address {
                  source-count number;
                  source-increment increment;
                }
              }
            }
            version version;
```

```
        }
      }
      router-advertisement {
        interface interface-name {
          current-hop-limit number;
          default-lifetime seconds;
          (managed-configuration | no-managed-configuration);
          max-advertisement-interval seconds;
          min-advertisement-interval seconds;
          (other-stateful-configuration | no-other-stateful-configuration);
          prefix prefix;
          reachable-time milliseconds;
          retransmit-timer milliseconds;
        }
      }
    }
  }
  routing-instances routing-instance-name {
    interface interface-name;
    routing-options {
      access {
        route prefix {
          next-hop next-hop;
          metric route-cost;
          preference route-distance;
          tag route-tag;
        }
      }
      access-internal {
        route subscriber-ip-address {
          qualified-next-hop underlying-interface {
            mac-address address;
          }
        }
      }
      multicast {
        interface interface-name {
          no-qos-adjust;
        }
      }
    }
    rib routing-table-name {
      access {
        route prefix {
          next-hop next-hop;
          metric route-cost;
          preference route-distance;
          tag route-tag;
        }
      }
      access-internal {
        route subscriber-ip-address {
          qualified-next-hop underlying-interface {
            mac-address address;
          }
        }
```

```
        }
      }
    }
    routing-options {
      access {
        route prefix {
          next-hop next-hop;
          metric route-cost;
          preference route-distance;
          tag route-tag;
        }
      }
      access-internal {
        route subscriber-ip-address {
          qualified-next-hop underlying-interface {
            mac-address address;
          }
        }
      }
      multicast {
        interface interface-name {
          no-qos-adjust;
        }
      }
    }
    variables {
      variable-name {
        default-value default-value;
        equals expression;
        mandatory;
        radius {
          vendor-id id {
            attribute attribute-number;
            tag tag-number;
          }
        }
        uid;
        uid-reference;
      }
    }
  }
}
```

Hierarchy Level [edit]

Release Information Statement introduced in Junos OS Release 9.2.
Support at the **filter, policer, hierarchical-policer, three-color-policer, and policy options** hierarchy levels introduced in Junos OS Release 11.4.

Description Create dynamic profiles for use with DHCP or PPP client access.

Options ***profile-name***—Name of the dynamic profile; string of up to 80 alphanumeric characters.

The remaining statements are explained separately.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Configuring a Basic Dynamic Profile*
- *Configuring Dynamic VLANs Based on Agent Circuit Identifier Information*
- *Dynamic Profiles Overview*

enhanced-policer

Syntax
enhanced-policer

Hierarchy Level
[edit chassis]

Release Information
Statement introduced in Junos OS Release 12.3.

Description
Display the collection of policer detailed statistics. An FPC restart is required after changing this configuration.

When you commit a configuration that contains the **enhanced-policer** statement at the **[edit chassis]** hierarchy level, a warning message is displayed stating that all the FPCs in the router need to be rebooted for the configuration changes to become effective. At this point, you must confirm that you want to proceed with the reboot of the FPCs. If you do not reboot the FPCs, the FPCs return all 0s (zeros) when you perform a query for the retrieval of detailed statistics—for example, when you issue the **show firewall detail** command.

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- *Router Chassis Configuration Statements*

excess-burst-size

Syntax excess-burst-size *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit firewall three-color-policer *policer-name* single-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** hierarchy level introduced in Junos Release OS 11.4.

Description For a single-rate three-color policer, configure the excess burst size (EBS) as a number of bytes. The EBS allows for moderate periods of bursting traffic that exceeds both the committed information rate (CIR) and the committed burst size (CBS).

NOTE: When you include the **excess-burst-size** statement in the configuration, you must also include the **committed-burst-size** and **committed-information-rate** statements at the same hierarchy level.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policing uses a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red based on the **excess-burst-size** statement included in the policer configuration.

During periods of traffic that conforms to the CIR, any unused portion of the guaranteed bandwidth capacity accumulates in the first token bucket, up to the maximum number of bytes defined by the CBS. If any accumulated bandwidth capacity overflows the first bucket, the excess accumulates in a second token bucket, up to the maximum number of bytes defined by the EBS.

A nonconforming traffic flow is categorized yellow if its size conforms to bandwidth capacity accumulated in the first token bucket. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.

A nonconforming traffic flow is categorized red if its size exceeds the bandwidth capacity accumulated in the second token bucket. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options *bytes*—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Dual Token Bucket Algorithms*
- *Determining Proper Burst Size for Traffic Policers*
- committed-burst-size on page 128
- committed-information-rate on page 130

fail-filter (Dynamic Profiles)

Syntax fail-filter *filter-name*;

Hierarchy Level [edit dynamic-profiles *profile-name* interfaces demux0 unit *logical-unit-number* family *family* rpf-check],
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* rpf-check]

Release Information Statement introduced in Junos OS Release 12.3.

Description Specify a filter that evaluates packets that fail a unicast RPF check. The filter determines what action to take with the failed packets. If the fail filter is not configured, the failed packets are silently discarded.

Options ***filter-name***—Name of the filter that evaluates packets that fail the RPF check.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Configuring Unicast RPF*
- Configuring a Fail Filter for Unicast RPF in Dynamic Profiles for Subscriber Interfaces on page 76

family (Dynamic Standard Interface)

Syntax

```
family family {
  access-concentrator name;
  address address;
  duplicate-protection;
  dynamic-profile profile-name;
  filter {
    adf {
      counter;
      input-precedence precedence;
      not-mandatory;
      output-precedence precedence;
      rule rule-value;
    }
    input filter-name {
      precedence precedence;
    }
    output filter-name {
      precedence precedence;
    }
  }
  mac-validate (loose | strict);
  max-sessions number;
  max-sessions-vsa-ignore;
  rpf-check {
    fail-filter filter-name;
    mode loose;
  }
  service {
    input {
      service-set service-set-name {
        service-filter filter-name;
      }
      post-service-filter filter-name;
    }
    output {
      service-set service-set-name {
        service-filter filter-name;
      }
    }
  }
  service-name-table table-name
  short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
    maximum-seconds>;
  unnumbered-address interface-name <preferred-source-address address>;
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 9.2.
Option **pppoe** introduced in Junos OS Release 11.2.

Description Configure protocol family information for the logical interface.

NOTE: Not all subordinate stanzas are available to every protocol family.

Options

family—Protocol family:

- **inet**—IP version 4 suite
- **inet6**—IP version 6 suite
- **pppoe**—(MX Series routers with MPCs only) Point-to-Point Protocol over Ethernet
- **vpls**—Virtual private LAN service

The remaining statements are explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- For general information about configuring static interfaces, see the *Junos OS Network Interfaces Library for Routing Devices*.
- "Configuring the Protocol Family," in the *Junos OS Network Interfaces Library for Routing Devices*.

family (Dynamic Firewalls)

Syntax

```
family family {
  fast-update-filter filter-name {
    interface-specific;
    match-order [match-order];
    term term-name {
      from {
        match-conditions;
      }
      then {
        action;
        action-modifiers;
      }
      only-at-create;
    }
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* firewall]

Release Information Statement introduced in Junos OS Release 9.6.

Description Configure protocol family information for firewall filters in a dynamic profile.

Options *family*—Protocol family:

- **inet**—Internet Protocol version 4 suite
- **inet6**—Internet Protocol version 6 suite

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63

fast-update-filter (Dynamic Firewalls)

Syntax

```
fast-update-filter filter-name {
  interface-specific;
  match-order [match-order];
  term term-name {
    from {
      match-conditions;
    }
    then {
      action;
      action-modifiers;
    }
    only-at-create;
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* firewall family *family*]

Release Information Statement introduced in Junos OS Release 9.6.

Description Configure fast update firewall filters in a dynamic profile.

Options *filter-name*—Name that identifies the filter. The name can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

The statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63

filter (Configuring)

Syntax

```
filter filter-name {
    accounting-profile name;
    enhanced-mode;
    interface-shared;
    interface-specific;
    physical-interface-filter;
    term term-name {
        ... term configuration ...
    }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall family *family-name*],
[edit firewall family *family-name*],
[edit logical-systems *logical-system-name* firewall family *family-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
physical-interface-filter statement introduced in Junos OS Release 9.6.
Support at the **[edit dynamic-profiles ... family family-name]** hierarchy level introduced in Junos OS Release 11.4.
Support for the **interface-shared**> statement introduced in Junos OS Release 12.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure firewall filters.

Options

filter-name—Name that identifies the filter. This must be a non-reserved string of not more than 64 characters. To include spaces in the name, enclose it in quotation marks (" "). In Junos OS Release 9.0 and later, you can no longer use special characters within the name of a firewall filter. Firewall filter names are restricted from having the form **__.*__** (beginning and ending with underscores) or **__.*** (beginning with an underscore.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Guidelines for Configuring Firewall Filters*
- *Guidelines for Applying Firewall Filters*
- *Configuring Multifield Classifiers*
- *Using Multifield Classifiers to Set PLP*
- *simple-filter (Configuring)*

filter (Dynamic Firewalls)

Syntax

```
filter {
  adf {
    counter;
    input-precedence precedence;
    not-mandatory;
    output-precedence precedence;
    rule rule-value;
  }
  input filter-name {
    precedence precedence;
    shared-name filter-shared-name;
  }
  output filter-name {
    precedence precedence;
    shared-name filter-shared-name;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family*],
[edit dynamic-profiles *profile-name* interfaces demux0 unit *logical-unit-number* family *family*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family*]

Release Information

Statement introduced in Junos OS Release 9.2.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family*]** hierarchy level introduced in Junos OS Release 10.1.
shared-name statement added in Junos OS Release 12.2.

Description

Apply a dynamic filter to an interface. You can configure filters for either **family inet** or **family inet6**, and the filters can be classic filters, fast update filters, or (for the **adf** statement) Ascend-Data-Filters. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

Options

input *filter-name*—Name of one filter to evaluate when packets are received on the interface.

output *filter-name*—Name of one filter to evaluate when packets are transmitted on the interface.

The remaining statements are explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- For general information about configuring firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*
- Dynamic Firewall Filters Overview on page 4

- Classic Filters Overview on page 7
- Basic Classic Filter Syntax on page 10

filter (Dynamic Interface Unit)

Syntax

```
filter {
  input filter-name;
  output filter-name;
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number*],
[edit dynamic-profiles *profile-name* interfaces demux0 unit *logical-unit-number*],

Release Information Statement introduced in Junos OS Release 9.2.

Description Apply a dynamic filter to an interface, regardless of its family type.

Options

input *filter-name*—Name of one filter to evaluate when packets are received on the interface.

output *filter-name*—Name of one filter to evaluate when packets are transmitted on the interface.

The remaining statement is explained separately.

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- For general information about configuring firewall filters, see the *Routing Policy Feature Guide for Routing Devices*
- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Basic Classic Filter Syntax on page 10
- Dynamically Attaching Statically Created Filters for Any Interface Type on page 54

filter-specific

Syntax filter-specific;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall family inet prefix-action *name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall family inet prefix-action *name*]

Release Information Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.

Description Set the prefix-specific action or policer to operate in *filter-specific* mode, meaning that a single policer and counter are shared by all filter terms that reference the prefix-specific action or policer. By default, the prefix-specific action or policer operates in *term-specific* mode, meaning that a separate policer and counter are used for each filter term that references the prefix-specific action or policer.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Filter-Specific Policer Overview*
- *Prefix-Specific Counting and Policing Overview*
- *Filter-Specific Counter and Policer Set Overview*

firewall (Dynamic Firewalls)

Syntax

```
firewall {
  family family {
    fast-update-filter filter-name {
      interface-specific;
      match-order [match-order];
      term term-name {
        from {
          match-conditions;
        }
        then {
          action;
          action-modifiers;
        }
        only-at-create;
      }
    }
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name*]

Release Information Statement introduced in Junos OS Release 9.6.

Description Configure firewall filters in a dynamic profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63

hierarchical-policer

Syntax

```
hierarchical-policer hierarchical-policer-name {
  aggregate {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
  premium {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... firewall]** hierarchy level introduced in Junos OS Release 11.4.

Description

On M40e, M120, and M320 edge routers with Flexible PIC Concentrator (FPC) input as FFPC and FPC output as SFPC, and on MX Series, T320, T640, and T1600 edge routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, specify a hierarchical policer.

Options

hierarchical-policer-name—Name that identifies the policer. The name can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Hierarchical Policer Configuration Overview*
- *Hierarchical Policers*
- aggregate (Hierarchical Policer) on page 117
- *bandwidth-limit (Hierarchical Policer)*
- burst-size-limit (Hierarchical Policer) on page 122

- if-exceeding (Hierarchical Policer) on page 153
- premium (Hierarchical Policer) on page 174

if-exceeding (Policer)

Syntax

```
if-exceeding {
  (bandwidth-limit bps | bandwidth-percent number);
  burst-size-limit bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure rate limits for a single-rate two-color policer.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Two-Color Policer Configuration Overview*
- *Hierarchical Policer Configuration Overview*
- *Basic Single-Rate Two-Color Policers*
- *Bandwidth Policers*
- *Filter-Specific Counters and Policers*
- *Prefix-Specific Counting and Policing Actions*
- *Multifield Classification*
- *Policer Overhead to Account for Rate Shaping in the Traffic Manager*
- *Hierarchical Policers*

if-exceeding (Hierarchical Policer)

Syntax

```
if-exceeding {
    bandwidth-limit bps;
    burst-size-limit bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer aggregate],
[edit dynamic-profiles *profile-name* firewall hierarchical-policer premium],
[edit firewall hierarchical-policer aggregate],
[edit firewall hierarchical-policer premium]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... aggregate]** and **[edit dynamic-profiles ... premium]** hierarchy level introduced in Junos OS Release 11.4.

Description

For M40e, M120, and M320 (with FFPC and SFPC) edge routers and T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, specify bandwidth and burst limits for a premium or aggregate component of a hierarchical policer.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Hierarchical Policer Configuration Overview*
- *Hierarchical Policers*
- aggregate (Hierarchical Policer) on page 117
- *bandwidth-limit (Hierarchical Policer)*
- burst-size-limit (Hierarchical Policer) on page 122
- hierarchical-policer on page 151
- premium (Hierarchical Policer) on page 174

input (Dynamic Service Sets)

Syntax

```
input {
  service-set service-set-name {
    service-filter filter-name;
  }
  post-service-filter filter-name;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family* service]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service]** hierarchy level introduced in Junos OS Release 10.1.

Description

Define the input service sets and filters to be applied to traffic by a dynamic profile. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

The remaining statements are explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

interface-shared

Syntax

```
interface-shared;
```

Hierarchy Level

[edit firewall family *family-name* filter *filter-name*],
[edit dynamic-profiles *profile-name*firewall family *family-name* filter *filter-name*]

Release Information

Statement introduced in Junos OS Release 12.2.

Description

Set the interface-shared attribute for a firewall filter.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Basic Classic Filter Syntax on page 10

interface-specific (Dynamic Firewalls)

Syntax interface-specific;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall family *family* fast-update-filter *filter-name*]

Release Information Statement introduced in Junos OS Release 9.6.

Description Configure interface-specific names for firewall counters that are based on fast update filters.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63

interfaces (Static and Dynamic Subscribers)

Syntax

```
interfaces {
  interface-name {
    unit logical-unit-number {
      auto-configure {
        agent-circuit-identifier {
          dynamic-profile profile-name;
        }
      }
      family family {
        access-concentrator name;
        address address;
        duplicate-protection;
        dynamic-profile profile-name;
        filter {
          adf {
            counter;
            input-precedence precedence;
            not-mandatory;
            output-precedence precedence;
            rule rule-value;
          }
          input filter-name (
            precedence precedence;
            shared-name filter-shared-name;
          }
          output filter-name {
            precedence precedence;shared-name filter-shared-name;
          }
        }
        max-sessions number;
        max-sessions-vsa-ignore;
        rpf-check {
          mode loose;
        }
        service {
          input {
            service-set service-set-name {
              service-filter filter-name;
            }
            post-service-filter filter-name;
          }
          output {
            service-set service-set-name {
              service-filter filter-name;
            }
          }
        }
        service-name-table table-name
        short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
          maximum-seconds>;
        unnumbered-address interface-name <preferred-source-address address>;
      }
```

```
      filter {
        input filter-name;
        shared-name filter-shared-name;
        output filter-name;
        shared-name filter-shared-name;
      }
      ppp-options {
        chap;
        pap;
      }
      proxy-arp;
      vlan-id;
      vlan-tags outer [tpid].vlan-id [inner [tpid].vlan-id];
    }
    vlan-tagging;
  }
  interface-set interface-set-name {
    interface interface-name {
      unit logical unit number {
        advisory-options {
          downstream-rate rate;
          upstream-rate rate;
        }
      }
    }
    pppoe-underlying-options {
      max-sessions number;
    }
  }
  demux0 {
    unit logical-unit-number {
      demux-options {
        underlying-interface interface-name
      }
      family family {
        access-concentrator name;
        address address;
        duplicate-protection;
        dynamic-profile profile-name;
        demux-source {
          source-prefix;
        }
        filter{
          input filter-name (
            precedence precedence;
            shared-name filter-shared-name;
          }
          output filter-name {
            precedence precedence;
            shared-name filter-shared-name;
          }
        }
        mac-validate (loose | strict):
        max-sessions number;
        max-sessions-vsa-ignore;
        rpf-check {
```

```
            fail-filter filter-name;
            mode loose;
          }
          service-name-table table-name
          short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
            maximum-seconds>;
          unnumbered-address interface-name <preferred-source-address address>;
        }
        filter {
          input filter-name;
          output filter-name;
        }
        vlan-id number;
        vlan-tags outer [tpid].vlan-id [inner [tpid].vlan-id];
      }
    }
    pp0 {
      unit logical-unit-number {
        keepalives interval seconds;
        no-keepalives;
        pppoe-options {
          underlying-interface interface-name;
          server;
        }
        ppp-options {
          authentication [ authentication-protocols ];
          chap {
            challenge-length minimum minimum-length maximum maximum-length;
          }
          pap;
        }
        family inet {
          unnumbered-address interface-name;
          address address;
          service {
            input {
              service-set service-set-name {
                service-filter filter-name;
              }
              post-service-filter filter-name;
            }
            output {
              service-set service-set-name {
                service-filter filter-name;
              }
            }
          }
          filter {
            input filter-name {
              precedence precedence;
              shared-name filter-shared-name;
            }
            output filter-name {
              precedence precedence;
              shared-name filter-shared-name;
            }
```

```
        }
      }
    }
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name*]

Release Information Statement introduced in Junos OS Release 9.2.

Description Define interfaces for dynamic profiles.

Options ***interface-name***—The interface variable (**$junos-interface-ifd-name**). The interface variable is dynamically replaced with the interface the DHCP client accesses when connecting to the router.

NOTE: Though we do not recommend it, you can also enter the specific name of the interface you want to assign to the dynamic profile.

The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring Static Subscriber Interfaces in Dynamic Profiles*
- *Configuring Dynamic Subscriber Interfaces Using IP Demux Interfaces in Dynamic Profiles*
- *Configuring Dynamic PPPoE Subscriber Interfaces Using Dynamic Profiles*
- *Configuring Dynamic VLANs Based on Agent Circuit Identifier Information*
- *Subscriber Interface Overview*
- *Relationship Between Subscribers and Interfaces in an Access Network*
- For general information about configuring static interfaces, see the *Junos OS Network Interfaces Library for Routing Devices*
- For information about static IP demux interfaces, see the *Junos OS Network Interfaces Library for Routing Devices*

logical-bandwidth-policer

Syntax logical-bandwidth-policer;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*]

Release Information Statement introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.

Description For a policer with a bandwidth limit configured as a percentage (using the bandwidth-percent statement), specify that the percentage be based on the shaping rate defined on the logical interface, rather than on the media rate of the physical interface.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Bandwidth Policers*
- *Configuring Logical Bandwidth Policers*
- bandwidth-percent on page 120 statement
- **interface-specific** statement

logical-interface-policer

Syntax logical-interface-policer;

Hierarchy Level
[edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall atm-policer*atm-policer-name*]
[edit firewall policer *policer-name*],
[edit firewall policer *policer-template-name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *name*]

Release Information
Statement introduced before Junos OS Release 7.4.
Support at the **[edit firewall three-color-policer *policer-name*]** hierarchy level introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** and **[edit dynamic-profiles ... three-color-policer *name*]** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Configure a logical interface policer.

NOTE: Starting in Junos OS Release 12.2R2, on T Series Core Routers only, you can configure an MPLS LSP policer for a specific LSP to be shared across different protocol family types. You must include the **logical-interface-policer** statement to do so.

Required Privilege Level
firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Two-Color and Three-Color Logical Interface Policers*
- *Traffic Policer Types*
- *Configuring Tricolor Marking Policers*
- action on page 115
- *Configuring Gigabit Ethernet Two-Color and Tricolor Policers*
- *action*

loss-priority high then discard (Three-Color Policer)

Syntax loss-priority high then discard;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* action],
[edit firewall three-color-policer *policer-name* action],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name* action]

Release Information Statement introduced before Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... action]** hierarchy level introduced in Junos OS Release 11.4.

Description For packets with high loss priority, discard the packets. The loss priority setting is implicit and is not configurable. Include this statement if you do not want the local router to forward packets that have high packet loss priority.

For single-rate three-color policers, the Junos OS assigns high loss priority to packets that exceed the committed information rate and the excess burst size.

For two-rate three-color policers, the Junos OS assigns high loss priority to packets that exceed the peak information rate and the peak burst size.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Basic Single-Rate Three-Color Policers*
- *Basic Two-Rate Three-Color Policers*
- *Two-Color and Three-Color Logical Interface Policers*
- *Two-Color and Three-Color Physical Interface Policers*
- *Two-Color and Three-Color Policers at Layer 2*
- action on page 115

match-order (Dynamic Firewalls)

Syntax match-order [*match-order*];

Hierarchy Level [edit **dynamic-profiles** ***profile-name*** **firewall** family *family* fast-update-filter *filter-name*]

Release Information Statement introduced in Junos OS Release 9.6.

Description Specify the match conditions and the order in which the conditions are examined. Enclose a string of multiple conditions in brackets. The router examines only the conditions you specify, and examines them in the specified order.

Options ***match-order***—One or more of the following conditions. "Fast Update Filter Match Conditions" on page 45 describes the match conditions.

- destination-address
- destination-port
- dscp (IPv4 only)
- protocol (IPv4 only)
- source-address
- source-port

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63
- Configuring the Match Order for Fast Update Filters on page 64
- Fast Update Filter Match Conditions on page 45

output (Dynamic Service Sets)

Syntax

```
service-set service-set-name {
    service-filter filter-name;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family* service]

Release Information

Statement introduced in Junos OS Release 9.5.
Support of the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service]** hierarchy level introduced in Junos OS Release 10.1.

Description

Define the output service sets and filters to be applied to traffic by a dynamic profile. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

The remaining statement is explained separately.

Options

service-set-name—Name of the service set.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

peak-burst-size

Syntax peak-burst-size *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... two-rate]** hierarchy level introduced in Junos OS Release 11.4.

Description For a two-rate three-color policer, configure the peak burst size (PBS) as a number of bytes. The PBS defines the maximum number of bytes of unused peak bandwidth capacity that can be accumulated. The accumulated bandwidth allows for moderate periods of bursting traffic that exceeds the peak information rate (PIR) and the committed burst size (CBS).

NOTE: When you include the **peak-burst-size** statement in the configuration, you must also include the **committed-burst-size** and **peak-information-rate** statements at the same hierarchy level.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits.

- A traffic flow is categorized green if it conforms to both the committed information rate (CIR) and the CBS-bounded accumulation of available committed bandwidth capacity.
- A traffic flow is categorized yellow if exceeds the CIR and CBS but conforms to the PIR. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.
- A traffic flow is categorized red if exceeds the PIR and the PBS-bounded accumulation of available peak bandwidth capacity. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options ***bytes***—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*

- *Policer Color-Marking and Actions*
- *Dual Token Bucket Algorithms*
- *Determining Proper Burst Size for Traffic Policers*
- committed-burst-size on page 128
- committed-information-rate on page 130
- excess-burst-size on page 140
- peak-information-rate on page 167

peak-information-rate

Syntax peak-information-rate *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... two-rate]** hierarchy level introduced in Junos OS Release 11.4.

Description For a two-rate three-color policer, configure the peak information rate (PIR) as a number of bits per second. The PIR is the maximum rate for traffic arriving at or departing from the interface under peak line conditions. Traffic that exceeds the committed information rate (CIR) and the committed burst size (CBS) is metered to the PIR.

NOTE: When you include the **peak-information-rate** statement in the configuration, you must also include the **committed-information-rate** and **peak-burst-size** statements at the same hierarchy level.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits.

- A traffic flow is categorized green if it conforms to both the CIR and the CBS-bounded accumulation of available committed bandwidth capacity.
- A traffic flow is categorized yellow if exceeds the CIR and CBS but conforms to the PIR. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.
- A traffic flow is categorized red if exceeds the PIR and the PBS-bounded accumulation of available peak bandwidth capacity. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options *bps*—Number of bits per second. You can specify a value in bits per second either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bps

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- *Policer Bandwidth and Burst-Size Limits*
- *Policer Color-Marking and Actions*
- *Dual Token Bucket Algorithms*

- *Determining Proper Burst Size for Traffic Policers*
- committed-burst-size on page 128
- committed-information-rate on page 130
- excess-burst-size on page 140
- peak-burst-size on page 165

physical-interface-policer

Syntax physical-interface-policer;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-system *logical-system-name* firewall policer *policer-name*],
[edit logical-system *logical-system-name* three-color-policer *policer-name*],
[edit routing-instances *routing-instance-name* firewall policer *policer-name*],
[edit routing-instances *routing-instance-name* firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall three-color-policer *policer-name*]

Release Information Statement introduced in Junos OS Release 9.6.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos Release OS 11.4.

Description Configure an aggregate policer for a physical interface.

A physical interface policer can be a two-color or three-color policer. When you apply physical interface policer, to different protocol families on the same logical interface, the protocol families share the same policer instance. This means that rate limiting is performed aggregately for the protocol families for which the policer is applied. This feature enables you to use a single policer instance to perform aggregate policing for different protocol families on the same physical interface. If you want a policer instance to be associated with a protocol family, the corresponding physical interface filter needs to be applied to that protocol family. The policer is not automatically applied to all protocol families configured on the physical interface.

In contrast, with logical interface policers there are multiple separate policer instances.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- *Two-Color and Three-Color Physical Interface Policers*
- *physical-interface-filter*

policer (Configuring)

Syntax

```
policer policer-name {
  filter-specific;
  if-exceeding {
    bandwidth-limit bps;
    bandwidth-percent number;
    burst-size-limit bytes;
  }
  logical-bandwidth-policer;
  logical-interface-policer;
  physical-interface-policer;
  shared-bandwidth-policer;
  then {
    policer-action;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
The **out-of-profile** policer action added in Junos OS Release 8.1.
The **logical-bandwidth-policer** statement added in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
The **physical-interface-policer** statement introduced in Junos OS Release 9.6.
The **shared-bandwidth-policer** statement added in Junos OS Release 11.2
Support at the **[edit dynamic-profiles ... firewall]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure policer rate limits and actions. When included at the **[edit firewall]** hierarchy level, the **policer** statement creates a template, and you do not have to configure a policer individually for every firewall filter or interface. To activate a policer, you must include the **policer-action** modifier in the **then** statement in a firewall filter term or on an interface.

Options

policer-action—One or more actions to take:

- **discard**—Discard traffic that exceeds the rate limits.
- **forwarding-class** *class-name*—Specify the particular forwarding class.
- **loss-priority**—Set the packet loss priority (PLP) to **low**, **medium-low**, **medium-high**, or **high**.
- **out-of-profile**—On J Series routers with strict priority queuing, prevent starvation of other queues by rate limiting the data stream entering the strict priority queue, marking the packets that exceed the rate limit as out-of-profile, and dropping the out-of-profile packets if the physical interface is congested.

policer-name—Name that identifies the policer. The name can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" "). Policer names cannot begin with an underscore in the form **__.***.

then—Actions to take on matching packets.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Bandwidth Policer Overview*
- *Configuring Firewall Filters and Policers for VPLS*
- *Configuring Multifield Classifiers*
- *Logical Interface (Aggregate) Policer Overview*
- *Physical Interface Policer Overview*
- *Statement Hierarchy for Configuring Policers*
- *Single-Rate Two-Color Policer Overview*
- *Using Multifield Classifiers to Set PLP*
- filter (Configuring) on page 146
- *priority (Schedulers)*

policy-options

Syntax

```
policy-options {
  as-path name regular-expression;
  as-path-group group-name;
  community name {
    invert-match;
    members [ community-ids ];
  }
  condition condition-name {
    if-route-exists address table table-name;
  }
  damping name {
    disable;
    half-life minutes;
    max-suppress minutes;
    reuse number;
    suppress number;
  }
  policy-statement policy-name {
    term term-name {
      from {
        family;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
        policy subroutine-policy-name;
      }
      then actions;
      default-action (accept | reject);
    }
  }
  prefix-list name {
    ip-addresses;
  }
}
```

Hierarchy Level

[edit],
[edit dynamic],
[edit dynamic-profiles *profile-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure routing policy.

The statements are explained separately.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Example: Using Routing Policy in an ISP Network*

post-service-filter (Dynamic Service Sets)

Syntax
post-service-filter *filter-name*;

Hierarchy Level
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service input],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family* service input]

Release Information
Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service input]** hierarchy level introduced in Junos OS Release 10.1.

Description
Define the filter to be applied to traffic after service processing. The filter is applied only if a service set is configured and selected. You can configure a postservice filter on the input side of the interface only. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

Options
filter-name—Identifier for the post-service filter.

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

precedence

Syntax precedence *precedence*;

Hierarchy Level
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter input *filter-name*],
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* filter output *filter-name*],
[edit dynamic-profiles *profile-name* interfaces demux0 unit *logical-unit-number* family *family* filter input *filter-name*],
[edit dynamic-profiles *profile-name* interfaces demux0 unit *logical-unit-number* family *family* filter output *filter-name*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family* filter input *filter-name*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family* filter output *filter-name*]

Release Information Statement introduced in Junos OS Release 9.3.
The **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family inet filter input *filter-name*]** hierarchy level and **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family inet filter output *filter-name*]** hierarchy level introduced in Junos OS Release 10.1.

Description Apply a precedence to a dynamic filter. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

Options *precedence*—Precedence value for the filter. The lower the precedence value, the higher the precedence.

Range: 0 through 250

Default: 0

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- For general information about configuring firewall filters, see the *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*
- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Fast Update Filters Overview on page 40
- Basic Classic Filter Syntax on page 10
- Basic Fast Update Filter Syntax on page 43

premium (Hierarchical Policer)

Syntax

```
premium {
  if-exceeding {
    bandwidth-limit bandwidth;
    burst-size-limit burst;
  }
  then {
    discard;
  }
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* firewall hierarchical-policer],
[edit firewall hierarchical-policer]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... hierarchical-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description On M40e, M120, and M320 edge routers with FPC input as FFPC and FPC output as SFPC, and on MX Series, T320, T640, and T1600 edge routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, specify a premium level for a hierarchical policer.

Options Options are described separately.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Applying Policers*
- *Junos OS Class of Service Library for Routing Devices*
- *Hierarchical Policer Configuration Overview*
- *Hierarchical Policers*
- aggregate (Hierarchical Policer) on page 117
- *bandwidth-limit (Hierarchical Policer)*
- burst-size-limit (Hierarchical Policer) on page 122
- hierarchical-policer on page 151
- if-exceeding (Hierarchical Policer) on page 153

rpf-check (Dynamic Profiles)

Syntax

```
rpf-check {
  fail-filter filter-name;
  mode loose;
}
```

Hierarchy Level [edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family*]

Release Information Statement introduced in Junos OS Release 9.6.

Description Check whether traffic is arriving on an expected path. You can include this statement with the **inet** protocol family only.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Configuring Unicast RPF*
- Configuring Unicast RPF and Fail Filters in Dynamic Profiles for Subscriber Interfaces on page 75

service (Dynamic Service Sets)

Syntax

```
service {
  input {
    service-set service-set-name {
      service-filter filter-name;
    }
    post-service-filter filter-name;
  }
  output {
    service-set service-set-name {
      service-filter filter-name;
    }
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos–interface–unit" family *family*]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family*]** hierarchy level introduced in Junos OS Release 10.1.

Description

Define the service sets and filters to be applied to an interface. This statement is not supported for **family inet6**.

The remaining statements are explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

service-filter (Dynamic Service Sets)

Syntax service-filter *filter-name*;

Hierarchy Level
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service input service-set *service-set-name*],
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service output service-set *service-set-name*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service input service-set *service-set-name*],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service output service-set *service-set-name*]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service input service-set *service-set-name*]** and **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service output service-set *service-set-name*]** hierarchy levels introduced in Junos OS Release 10.1.

Description Define the filter to be applied to traffic before it is accepted for service processing. Configuration of a service filter is optional; if you include the **service-set** statement without a **service-filter** definition, the router software assumes that the match condition is true and selects the service set for processing automatically. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

Options ***filter-name***—Identifies the filter to be applied in service processing.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

service-set (Dynamic Service Sets)

Syntax

```
service-set service-set-name {
    service-filter filter-name;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service input],
[edit dynamic-profiles *profile-name* interfaces *interface-name* unit *logical-unit-number* family *family* service output],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service input],
[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service output]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service input]** and **[edit dynamic-profiles *profile-name* interfaces pp0 unit "$junos-interface-unit" family *family* service output]** hierarchy levels introduced in Junos OS Release 10.1.

Description

Define one or more service sets in a dynamic profile. Service sets are applied to an interface. If you define multiple service sets, the router software evaluates the filters in the order in which they appear in the configuration. Only the Internet Protocol version 4 (IPv4) protocol family is currently supported for dynamic PPPoE logical interfaces.

Options

service-set-name—Name of the service set.

The remaining statement is explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73

interface-shared

Syntax interface-shared;

Hierarchy Level [edit firewall family *family-name* filter *filter-name*],
[edit dynamic-profiles *profile-name*firewall family *family-name* filter *filter-name*]

Release Information Statement introduced in Junos OS Release 12.2.

Description Set the interface-shared attribute for a firewall filter.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Dynamic Firewall Filters Overview on page 4
- Classic Filters Overview on page 7
- Basic Classic Filter Syntax on page 10

single-rate

Syntax

```
single-rate {
    (color-aware | color-blind);
    committed-information-rate bps;
    committed-burst-size bytes;
    excess-burst-size bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure a single-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), and excess burst size (EBS).

Packets that conform to the CIR or the CBS are assigned low loss priority (green). Packets that exceed the CIR and the CBS but are within the EBS are assigned medium-high loss priority (yellow). Packets that exceed the EBS are assigned high loss priority (red).

Green and yellow packets are always forwarded; this action is not configurable. You can configure red packets to be discarded. By default, red packets are forwarded.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- color-aware on page 126
- color-blind on page 127
- two-rate on page 184

term

Syntax

```
term term-name {
  from {
    match-conditions;
  }
  then {
    action;
    action-modifiers;
  }
  only-at-create;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall family *family-name* fast-update-filter *filter-name*],
[edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*]

Release Information

Statement introduced in Junos OS Release 9.6.
Support at the **[edit dynamic-profiles ... filter *filter-name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

Define terms for fast update filters.

Options

action—(Optional) An action to take if conditions match. If you do not specify an action, the packets that match the conditions in the **from** statement are accepted.

action-modifiers—(Optional) One or more actions to perform on a packet.

from—(Optional) Match packet fields to values. If not included, all packets are considered to match and the actions and action modifiers in the **then** statement are taken.

match-conditions—One or more conditions to make a match.

only-at-create—(Optional) Specify that the term is added only when the fast update filter is first created. No subsequent changes can be made to the term in the filter. Use this option only for terms that do not include subscriber-specific data in their match conditions, such as common or default terms (for example, counting the default drop packets).

term-name—Name that identifies the term. The name can contain letters, numbers, and hyphens (-), and can be up to 64 characters long. To include spaces in the name, enclose it in quotation marks (" ").

then—(Optional) Actions to take on matching packets. If not included and a packet matches all the conditions in the **from** statement, the packet is accepted.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Fast Update Filters on page 63
- Configuring Terms for Fast Update Filters on page 65

three-color-policer (Configuring)

Syntax

```
three-color-policer policer-name {
  action {
    loss-priority high then discard;
  }
  filter-specific;
  logical-interface-policer;
  physical-interface-policer;
  shared-bandwidth-policer;
  single-rate {
    (color-aware | color-blind);
    committed-burst-size bytes;
    committed-information-rate bps;
    excess-burst-size bytes;
  }
  two-rate {
    (color-aware | color-blind);
    committed-burst-size bytes;
    committed-information-rate bps;
    peak-burst-size bytes;
    peak-information-rate bps;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
The **action** and **single-rate** statements added in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... firewall]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure a three-color policer.

Options

policer-name—Name of the three-color policer. Reference this name when you apply the policer to an interface.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Statement Hierarchy for Configuring Policers*
- *Configuring Tricolor Marking Policers*
- *Three-Color Policer Configuration Guidelines*
- *Basic Single-Rate Three-Color Policers*

- *Basic Two-Rate Three-Color Policers*
- *Two-Color and Three-Color Logical Interface Policers*
- *Two-Color and Three-Color Physical Interface Policers*
- *Two-Color and Three-Color Policers at Layer 2*

two-rate

Syntax

```
two-rate {
  (color-aware | color-blind);
  committed-information-rate bps;
  committed-burst-size bytes;
  peak-information-rate bps;
  peak-burst-size bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer *name*** hierarchy levels introduced in Junos OS Release 11.4.

Description

Configure a two-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), peak information rate (PIR), and peak burst size (PBS).

Packets that conform to the CIR or the CBS are assigned low loss priority (green). Packets that exceed the CIR and the CBS but are within the PIR or the PBS are assigned medium-high loss priority (yellow). Packets that exceed the PIR and the PBS are assigned high loss priority (red).

Green and yellow packets are always forwarded; this action is not configurable. You can configure red packets to be discarded. By default, red packets are forwarded.

The remaining statements are explained separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Three-Color Policer Configuration Overview*
- color-aware on page 126
- color-blind on page 127
- single-rate on page 180

unit (Dynamic Profiles Standard Interface)

Syntax

```
unit logical-unit-number {
  auto-configure {
    agent-circuit-identifier {
      dynamic-profile profile-name;
    }
  }
  dial-options {
    ipsec-interface-id name;
    l2tp-interface-id name;
    (shared | dedicated);
  }
  encapsulation (atm-ccc-cell-relay | atm-ccc-vc-mux | atm-cisco-nlpid | atm-tcc-vc-mux
    | atm-mlppp-llc | atm-nlpid | atm-ppp-llc | atm-ppp-vc-mux | atm-snap | atm-tcc-snap
    | atm-vc-mux | ether-over-atm-llc | ether-vpls-over-atm-llc | ether-vpls-over-fr |
    ether-vpls-over-ppp | ethernet | frame-relay-ccc | frame-relay-ppp | frame-relay-tcc |
    frame-relay-ether-type | frame-relay-ether-type-tcc | multilink-frame-relay-end-to-end
    | multilink-ppp | ppp-over-ether | ppp-over-ether-over-atm-llc | vlan-bridge | vlan-ccc |
    vlan-vci-ccc | vlan-tcc | vlan-vpls);
  family family {
    access-concentrator name;
    address address;
    duplicate-protection;
    dynamic-profile profile-name;
    filter {
      adf {
        counter;
        input-precedence precedence;
        not-mandatory;
        output-precedence precedence;
        rule rule-value;
      }
      input filter-name (
        precedence precedence;
      }
      output filter-name {
        precedence precedence;
      }
    }
    max-sessions number;
    max-sessions-vsa-ignore;
    rpf-check {
      fail-filter filter-name;
      mode loose;
    }
    service {
      input {
        service-set service-set-name {
          service-filter filter-name;
        }
        post-service-filter filter-name;
      }
      input-vlan-map {
```

```
            inner-tag-protocol-id tpid;
            inner-vlan-id number;
            (push | swap);
            tag-protocol-id tpid;
            vlan-id number;
          }
          output {
            service-set service-set-name {
              service-filter filter-name;
            }
          }
          output-vlan-map {
            inner-tag-protocol-id tpid;
            inner-vlan-id number;
            (pop | swap);
            tag-protocol-id tpid;
            vlan-id number;
          }
        }
        service-name-table table-name
        short-cycle-protection <lockout-time-min minimum-seconds lockout-time-max
          maximum-seconds>;
        unnumbered-address interface-name <preferred-source-address address>;
        filter {
          input filter-name;
          output filter-name;
        }
        keepalives {
          interval seconds;
          }
        ppp-options {
          chap;
          pap;
        }
        vlan-id number;
        vlan-tags outer [tpid].vlan-id [inner [tpid].vlan-id];
      }
    }
```

Hierarchy Level [edit dynamic-profiles *profile-name* interfaces *interface-name*]

Release Information Statement introduced in Junos OS Release 9.2.

Description Configure a logical interface on the physical device. You must configure a logical interface to be able to use the physical device.

Options

logical-unit-number—The specific unit number of the interface you want to assign to the dynamic profile, or one of the following Junos OS predefined variables:

- **$junos-underlying-interface-unit**—For static VLANs, the unit number variable. The static unit number variable is dynamically replaced with the client unit number when the client session begins. The client unit number is specified by the DHCP when it accesses the subscriber network.
- **$junos-interface-unit**—The unit number variable on a dynamic underlying VLAN interface for which you want to enable the creation of dynamic VLAN subscriber interfaces based on agent circuit identifier information.

The remaining statements are explained separately.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Configuring Dynamic Underlying VLAN Interfaces to Use Agent Circuit Identifier Information*
- *Configuring Static Underlying VLAN Interfaces to Use Agent Circuit Identifier Information*
- *Agent Circuit Identifier-Based Dynamic VLANs Components Overview*

PART 3

Administration

CHAPTER 12

Verifying and Managing Filter and Service Set Configurations

Verifying and Managing Firewall Filter Configuration

Purpose View or manage information for firewall filters:

NOTE: The router creates unique names for fast update filters and for filter terms and counters. See *Naming Fast Update Filters* in "Fast Update Filters Overview" on page 40 for information.

Action

- To display statistics for firewall filters:

  user@host> **show firewall**

- To display firewall log information:

  user@host> **show firewall log**

- To clear filter counters:

  user@host> **clear firewall all**

Related Documentation

- Classic Filters Overview on page 7
- Fast Update Filters Overview on page 40
- *Junos OS Operational Mode Commands*

Verifying and Managing Dynamic Ascend-Data-Filter Policy Configuration

Purpose View or manage information for Ascend-Data-Filters.

Action

- To display statistics for Ascend-Data-Filters:

user@host> show firewall

- To display firewall log information:

  user@host> show subscribers **extensive**

- To clear filter counters:

  user@host> clear firewall **all**

Related Documentation

- Ascend-Data-Filter Policies for Subscriber Management Overview on page 11
- Dynamically Applying Ascend-Data-Filter Policies to Subscriber Sessions on page 61

Verifying and Managing Service Sets Information

Purpose View information for service sets:

Action

- To display summary information for service sets:

  user@host> show services service-sets summary

- To display interface-specific information for service sets:

  user@host>show services service-sets summary **interface** ***interface-name***

Related Documentation

- Dynamic Service Sets Overview on page 5
- Associating Service Sets with Interfaces in a Dynamic Profile on page 73
- *Junos OS Operational Mode Commands*

CHAPTER 13

Monitoring Commands

clear firewall

Syntax clear firewall (all | counter *counter-name* | filter *filter-name* | log (all | *logical-system-name*) | logical-system *logical-system-name*)

Syntax (EX Series Switches) clear firewall (all | counter *counter-name* | filter *filter-name* | log (all | *logical-system-name*) | policer counter (all | counter-id *counter-index*))

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
logical-system option introduced in Junos OS Release 9.3.
log option introduced before Junos OS Release 11.4.

Description Clear statistics about configured firewall filters.

When you clear the counters of a filter, this impacts not only the counters shown by the CLI, but also the ones tracked by SNMP2.

NOTE: The clear firewall command cannot be used to clear the Routing Engine filter counters on a backup Routing Engine that is enabled for graceful Routing Engine switchover (GRES).

If you clear statistics for firewall filters that are applied to Trio-based DPCs and that also use the **prefix-action** action on matched packets, wait at least 5 seconds before you enter the **show firewall prefix-action-stats** command. A 5-second pause between issuing the **clear firewall** and **show firewall prefix-action-stats** commands avoids a possible timeout of the **show firewall prefix-action-stats** command.

Options

all—Clear the packet and byte counts for all filters. On EX Series switches, this option also clears the packet counts for all policer counters.

counter *counter-name*—Clear the packet and byte counts for a filter counter that has been configured with the counter firewall filter action.

filter *filter-name*—Clear the packet and byte counts for the specified firewall filter.

log (all | *logical-system-name*)—Clear log entries for IPv4 firewall filters that have **then log** as an action. Use **log all** to clear all log entries or **log *logical-system-name*** to clear log entries for the specified logical system.

logical-system *logical-system-name*—Clear the packet and byte counts for the specified logical system.

policer counter (all | counter-id *counter-index*)—(EX8200 switches only) Clear all policer counters using the **policer counter all** command, or clear a specific policer counter using the **policer counter counter-id *counter-index*** command. The value of *counter-index* can be 0, 1, or 2.

Required Privilege Level clear

Related Documentation
- show firewall on page 196

List of Sample Output

Sample Output

clear firewall all

```
user@host> clear firewall all
```

clear firewall (counter counter-name)

```
user@host> clear firewall counter port-filter-counter
```

clear firewall (filter filter-name)

```
user@host> clear firewall filter ingress-port-filter
```

clear firewall (policer counter all) (EX8200 Switch)

```
user@switch> clear firewall policer counter all
```

clear firewall (policer counter counter-id counter-index) (EX8200 Switch)

```
user@switch> clear firewall policer counter counter-id 0
```

show firewall

Syntax

show firewall
<counter *counter-name*>
<filter *filter-name*>
<log>
<logical-system (all | *logical-system-name*)>
<terse>

Syntax (EX Series Switches)

show firewall
<counter *counter-name*>
<detail>
<filter *filter-name*>
<log <(detail | interface *interface-name*)>>
<policer counters <(detail | counter-id *counter-index* <detail>)>>
<terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
logical-system option introduced in Junos OS Release 9.3.
terse option introduced in Junos OS Release 9.4.
policer counters option introduced in Junos OS Release 12.2 for EX Series switches.
detail option introduced in Junos OS Release 12.3.

Description

Display statistics about configured firewall filters.

Options

none—(Optional) Display statistics about all configured firewall filters and counters. For EX Series switches, this command also displays statistics about all configured policers.

counter *counter-name*—(Optional) Name of a filter counter.

detail—(EX Series switches only) (Optional) Display firewall filter statistics with enhanced policer.

filter *filter-name*—(Optional) Name of a configured filter.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

log—(Optional) Display log entries for firewall filters.

log <(detail | interface *interface-name*)>—(EX Series switches only) (Optional) Display detailed log entries of firewall activity or log information about a specific interface.

policer counters <(detail | counter-id *counter-index* <detail>)>—(EX8200 switches only) (Optional) Display policer counter statistics in brief or in detail.

terse—(Optional) Display firewall filter names only.

Required Privilege Level

view

Related Documentation

- clear firewall on page 194
- show firewall log on page 203
- *Verifying That Firewall Filters Are Operational*
- *Verifying That Policers Are Operational*

List of Sample Output

Output Fields

Table 10 on page 197 lists the output fields for the **show firewall** command. Output fields are listed in the approximate order in which they appear.

Table 10: show firewall Output Fields

Field Name	Field Description
Filter	Name of a filter that has been configured with the **filter** statement at the **[edit firewall]** hierarchy level. Except on EX Series switches: • When an interface-specific filter is displayed, the name of the filter is followed by the full interface name and by either **-i** for an input filter or **-o** for an output filter. • When dynamic filters are displayed, the name of the filter is followed by the full interface name and by either **-in** for an input filter or **-out** for an output filter. When a logical system–specific filter is displayed, the name of the filter is prefixed with two underscore (__) characters and the name of the logical system (for example, **__ls1/filter1**). • When a service filter is displayed that uses a service set, the separator between the service-set name and the service-filter name is a semicolon (:).
Counters	Display filter counter information: • **Name**—Name of a filter counter that has been configured with the **counter** firewall filter action. • **Bytes**—Number of bytes that match the filter term under which the **counter** action is specified. • **Packets**—Number of packets that matched the filter term under which the **counter** action is specified. NOTE: On M and T series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per Packet Forwarding Engine (PFE) basis. See *show pfe statistics ip* for sample output.

Table 10: show firewall Output Fields *(continued)*

Field Name	Field Description
Policers	Display policer information: • **Name**—Name of policer. • **Bytes**—(For two-color policers on MX Series routers and EX Series switches, and for hierarchical policers on MS-DPC, MIC, and MPC interfaces on MX Series routers) Number of bytes that match the filter term under which the policer action is specified. This is only the number out-of-specification (out-of-spec) byte counts, not all the bytes in all packets policed by the policer. For other combinations of policer type, device, and line card type, this field is blank. • **Packets**—Number of packets that matched the filter term under which the policer action is specified. This is only the number of out-of-specification (out-of-spec) packet counts, not all packets policed by the policer.
Policer Counter Index	(EX8200 switch only) Global management counter ID. The counter ID value (*counter-index*) can be 0, 1, or 2.
Green	(EX8200 switch only) Number of packets within the limits. The number of packets is smaller than the committed information rate (CIR).
Yellow	(EX8200 switch only) Number of packets partially within the limits. The number of packets is greater than the CIR, but the burst size is within the excess burst size (EBS) limit.
Discard	(EX8200 switch only) Number of discarded packets.
Bytes	(EX8200 switch only) Number of green, yellow, red, or discarded packets in bytes.
Packets	(EX8200 switch only) Number of green, yellow, red, or discarded packets.
Filter name	(EX8200 switch only) Name of the filter with a term associated to a policer.
Term name	(EX8200 switch only) Name of the term associated with a policer.
Policer name	(EX8200 switch only) Name of the policer that is associated with a global management counter.

Sample Output

show firewall filter (MX Series Router and EX Series Switch)

```
user@host> show firewall filter test
Filter: test
Counters:
Name                                Bytes              Packets
Counter-1                               0                    0
Counter-2                               0                    0
Policers:
Name                                Bytes              Packets
Policer-1                            2770                   70
```

show firewall filter (non MX Series Router and EX Series Switch)

```
user@host> show firewall filter test
Filter: test
Counters:
Name                                Bytes              Packets
Counter-1                               0                    0
Counter-2                               0                    0
Policers:
Name                                Bytes              Packets
Policer-1                                                   70
```

show firewall filter (Hierarchical Policer, MX Series with MPC)

```
user@host> show firewall filter
FL_V4_PHY-HP-EF-AWARE-Gold=400k-MCAST=200k-Total=1M-ds-10/0/0:2:1-i


Filter: FL_V4_PHY-HP-EF-AWARE-Gold=400k-MCAST=200k-Total=1M-ds-10/0/0:2:1-i
Counters:
Name                                                  Bytes              Packets
AF1x_counter-ds-10/0/0:2:1-i                              0                    0
AF2x_counter-ds-10/0/0:2:1-i                    25529445976             24500428
AF3x_counter-ds-10/0/0:2:1-i                        2182022                39482
AF4x_counter-ds-10/0/0:2:1-i                              0                    0
BE_counter-ds-10/0/0:2:1-i                                0                    0
EF_counter-ds-10/0/0:2:1-i                      14817044120             12265765
STD_counter-ds-10/0/0:2:1-i                               0                    0
Policers:
Name                                                  Bytes              Packets
POL_CE-PE_M=200k-filter-ds-10/0/0:2:1-i            5948099658              5708349
POL_CE-PE_G=400K_R=1M-filter-ds-10/0/0:2:1-i   ??????????              3572794
???????????                                    ??????????              ???????
```

show firewall filter (Dynamic Input Filter)

```
user@host> show firewall filter dfwd-ge-5/0/0.1-in
Filter: dfwd-ge-5/0/0.1-in
Counters:
Name                                                  Bytes              Packets
c1-ge-5/0/0.1-in                                          0                    0
```

show firewall (Logical Systems)

```
user@host> show firewall
```

```
Filter: __lr1/test
Counters:
Name                                                Bytes              Packets
icmp                                                  420                    5
Filter: __default_bpdu_filter__
Filter: __lr1/inet_filter1
Counters:
Name                                                Bytes              Packets
inet_tcp_count                                          0                    0
inet_udp_count                                          0                    0
Filter: __lr1/inet_filter2
Counters:
Name                                                Bytes              Packets
inet_icmp_count                                         0                    0
inet_pim_count                                          0                    0
Filter: __lr2/inet_filter1
Counters:
Name                                                Bytes              Packets
inet_tcp_count                                          0                    0
inet_udp_count                                          0                    0
```

show firewall (counter counter-name)

```
user@host> show firewall counter icmp-counter
Filter: ingress-port-voip-class-filter
Counters:
Name                                                Bytes              Packets
icmp-counter                                            0                    0
```

show firewall log

```
user@host> show firewall log
Log :

Time      Filter    Action Interface     Protocol        Src Addr
          Dest Addr
08:00:53  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:52  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
08:00:51  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
08:00:50  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
08:00:49  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
08:00:48  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
08:00:47  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
         192.168.3.4
```

show firewall policer counters (EX8200 Switch)

```
user@switch> show firewall policer counters
Policer Counter Index 0:
                          Bytes            Packets
Green:                       73              15914
Yellow:                       9               1962
Discard:                    119              25942
```

```
Policer Counter Index 1:
                            Bytes            Packets
Green:                          0                  0
Yellow:                         0                  0
Discard:                        0                  0

Policer Counter Index 2:
                            Bytes            Packets
Green:                          0                  0
Yellow:                         0                  0
Discard:                        0                  0
```

show firewall policer counters (detail) (EX8200 Switch)

```
user@switch> show firewall policer counters detail
Policer Counter Index 0:
                           Bytes            Packets
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942

Filter name                Term name           Policer name
myfilter                   polcr-term-1        myfilter-polcr-1
inet-filter-ae             ae-snmp             policer-1
inet-filter-ae             ae-ssh              policer-2

Policer Counter Index 1:
                           Bytes            Packets
Green:                          0                  0
Yellow:                         0                  0
Discard:                        0                  0

Filter name                Term name           Policer name

Policer Counter Index 2:
                           Bytes            Packets
Green:                          0                  0
Yellow:                         0                  0
Discard:                        0                  0

Filter name                Term name           Policer name
```

show firewall policer counters (counter-id counter-index) (EX8200 Switch)

```
user@switch> show firewall policer counters counter-id 0
Policer Counter Index 0:
                           Bytes            Packets
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942
```

show firewall policer counters (counter-id counter-index detail) (EX8200 Switch)

```
user@switch> show firewall policer counters counter-id 0 detail
Policer Counter Index 0:
                           Bytes            Packets
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942

Filter name                Term name           Policer name
```

```
myfilter                  polcr-term-1        myfilter-polcr-1
inet-filter-ae            ae-snmp             policer-1
inet-filter-ae            ae-ssh              policer-2
```

show firewall detail

```
user@host> show firewall detail
Filter: __default_bpdu_filter__

Filter: foo
Counters:
Name                                                Bytes              Packets
c1                                              17652140               160474
Policers:
Name                                                Bytes              Packets
P1-t1
    OOS                                                 0                18286
    Offered                                             0 18446744073709376546
    Transmitted                                         0 18446744073709358260
```

show firewall log

Syntax

show firewall log
<detail>
<interface *interface-name*>
<logical-system (*logical-system-name* | all)>

Syntax (EX Series Switches)

show firewall log
<detail>
<interface *interface-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
logical-system option introduced in Junos OS Release 9.3.

Description

Display log information about firewall filters.

Options

none—Display log information about firewall filters.

detail—(Optional) Display detailed information.

interface *interface-name*—(Optional) Display log information about a specific interface.

logical-system (*logical-system-name* | all)—(Optional) Perform this operation on all logical systems or on a particular system.

Required Privilege Level

view

List of Sample Output

show firewall log on page 204
show firewall log detail on page 204

Output Fields

Table 11 on page 203 lists the output fields for the **show firewall log** command. Output fields are listed in the approximate order in which they appear.

Table 11: show firewall log Output Fields

Field Name	Field Description
Time of Log	Time that the event occurred.
Filter	• Displays the name of a configured firewall filter or service filter only if the packet hit the filter's **log** action in a kernel filter (in the control plane). For any traffic that reaches the Routing Engine, the packets hit the **log** action in the kernel. • For all other logged packets (packet hit the filter's **log** action in the Packet Forwarding Engine), this field displays **pfe** instead of a configured filter name.

Table 11: show firewall log Output Fields *(continued)*

Field Name	Field Description
Filter Action	Filter action: • **A**—Accept • **D**—Discard • **R**—Reject
Name of Interface	• Displays a physical interface name if the packet arrived at a port on a line card. • Displays **local** if the packet was generated by the device's internal Ethernet interface, **em1** or **fxp1**, which connects the Routing Engine with the router's packet-forwarding components.
Name of protocol	Packet's protocol name: **egp**, **gre**, **icmp**, **ipip**, **ospf**, **pim**, **rsvp**, **tcp**, or **udp**.
Packet length	Length of the packet.
Source address	Packet's source address.
Destination address	Packet's destination address and port.

Sample Output

show firewall log

```
user@host>show firewall log
Time      Filter    Action Interface     Protocol  Src Addr      Dest Addr

13:10:12  pfe       D      rlsq0.902     ICMP      180.1.177.2   180.1.177.1

13:10:11  pfe       D      rlsq0.902     ICMP      180.1.177.2   180.1.177.1
```

show firewall log detail

```
user@host> show firewall log detail
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0Name of protocol: TCP, Packet Length: 50824, Source address:
172.17.22.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 1020, Source address: 172.17.22.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 172.17.22.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 172.17.22.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
```

```
Name of protocol: TCP, Packet Length: 49245, Source address: 172.17.22.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 172.17.22.108:829,
Destination address: 192.168.70.66:513
....
```

show firewall templates-in-use

Syntax show firewall templates-in-use

Release Information Command introduced in Junos OS Release 12.3.

Description Display the names of configured filter templates that are currently in use by dynamic subscribers and the number of times each template is referenced.

Required Privilege Level view

Related Documentation

- clear firewall on page 194
- show firewall log on page 203
- *Verifying That Firewall Filters Are Operational*

List of Sample Output show firewall templates-in-use on page 207

Output Fields Table 12 on page 206 lists the output fields for the **show firewall templates-in-use** command. Output fields are listed in the approximate order in which they appear.

Table 12: show firewall templates-in-use Output Fields

Field Name	Field Description
Filter Template	Name of a filter that has been configured using the **filter** statement at either the **[edit firewall]** or **[edit dynamic-profiles *profile-name* firewall]** hierarchy and is being used as a template for dynamic subscriber filtering.
Reference Count	Number of times the filter has been referenced by subscribers accessing the network.

Sample Output

show firewall templates-in-use

```
user@host> show firewall templates-in-use
                                  Dynamic Subscribers Reference Counts
Filter Template
                    Reference Count
---------------
                    ---------------
egressFilter
                          10
ingressFilter
                          10
dfilter
                           5
dfilter-pol
                           5
```

show services service-sets summary

Syntax show services service-sets summary
<interface *interface-name*>

Release Information Command introduced before Junos OS Release 7.4.

Description Display service set summary information.

Options **none**—Display service set summary information for all adaptive services interfaces.

interface *interface-name*—(Optional) Display service set summary information for a particular interface. On M Series and T Series routers, ***interface-name*** can be **ms-*fpc/pic/port***, **sp-*fpc/pic/port***, or **rsp*number***. On J Series routers, ***interface-name*** is **sp-*pim*/0/*port***.

Required Privilege Level view

List of Sample Output show services service-sets summary on page 208
show services service-sets summary interface on page 209

Output Fields Table 13 on page 208 lists the output fields for the **show services service-sets summary** command. Output fields are listed in the approximate order in which they appear.

Table 13: show services service-sets summary Output Fields

Field Name	Field Description
Interface	Name of an adaptive services interface
Service type	Type of adaptive service, such as stateful firewall (SFW), Network Address Translation (NAT), intrusion detection service (IDS), Layer 2 Tunneling Protocol (L2TP), Compressed Real-Time Transport Protocol (CRTP), or IP Security (IPsec)
Service sets configured	Total number of service sets configured on the PIC that use internal service set IDs and do not consume external service sets, including CRTP and L2TP
Bytes used	Bytes used by a particular service or all services
Policy bytes used	Policy bytes used by a particular service or all services
CPU utilization	Percentage of the CPU resources being used

Sample Output

show services service-sets summary

```
user@host> show services service-sets summary
```

```
    Service sets                                                  CPU
Interface    configured            Bytes used    Policy bytes used  utilization
ms-4/0/0              1   14821556 ( 4.53 %)     855124 ( 0.40 %)        N/A
 ms-4/1/0             1   14691700 ( 4.49 %)     855068 ( 0.40 %)        N/A
```

show services service-sets summary interface

```
user@host> show services service-sets summary interface sp-1/3/0
Interface: sp-1/3/0
                 Service sets                          CPU
  Service type   configured             Bytes used   utilization
  SFW/NAT/IDS               1          54 ( 0.00 %)       N/A
  L2TP                      1          58 ( 0.00 %)       N/A
  CRTP                      1          58 ( 0.00 %)       N/A
  System                    0      920831 ( 0.44 %)       N/A
  Idle                      0           0 ( 0.00 %)       N/A
  Total                     3      921001 ( 0.44 %)       N/A
```

show subscribers

Syntax

show subscribers
<detail | extensive | terse>
<aci-interface-set-name *aci-interface-set-name*>
<address *address*>
<agent-circuit-identifier *agent-circuit-identifier-substring*>
<client-type *client-type*>
<count>
<interface *interface*>
<logical-system *logical-system*>
<mac-address *mac-address*>
<physical-interface *physical-interface-name*>
<profile-name *profile-name*>
<routing-instance *routing-instance*>
<stacked-vlan-id *stacked-vlan-id*>
<subscriber-state *subscriber-state*>
<user-name *user-name*>
<vci *vci-identifier*>
<vpi *vpi-identifier*>
<vlan-id *vlan-id*>

Release Information

Command introduced in Junos OS Release 9.3.
Command introduced in Junos OS Release 9.3 for EX Series switches.
client-type, **mac-address**, **subscriber-state**, and **extensive** options introduced in Junos OS Release 10.2.
count option usage with other options introduced in Junos OS Release 10.2.
Command introduced in Junos OS Release 11.1 for the QFX Series.
Options **aci-interface-set-name** and **agent-circuit-identifier** introduced in Junos OS Release 12.2.
The **physical-interface** and **user-name** options introduced in Junos OS Release 12.3.
Options **vci** and **vpi** introduced in Junos OS Release 12.3R3 and supported in later 12.3R*x* releases.
Options **vci** and **vpi** supported in Junos OS Release 13.2 and later releases. (Not supported in Junos OS Release 13.1.)

Description

Display information for active subscribers.

Options

detail | extensive | terse—(Optional) Display the specified level of output.

aci-interface-set-name—(Optional) Display all dynamic subscriber sessions that use the specified agent circuit identifier (ACI) interface set. Use the ACI interface set name generated by the router, such as aci-1003-ge-1/0/0.4001, and not the actual ACI value found in the DHCP or PPPoE control packets.

address—(Optional) Display subscribers whose IP address matches the specified address. You must specify the IPv4 or IPv6 address prefix without a netmask (for example, 192.168.17.1). If you specify the IP address as a prefix with a netmask (for example, 192.168.17.1/32), the router displays a message that the IP address is invalid, and rejects the command.

agent-circuit-identifier-substring—(Optional) Display all dynamic subscriber sessions whose ACI value matches the specified substring.

client-type—(Optional) Display subscribers whose client type matches the specified client type (DHCP, L2TP, PPP, PPPOE, VLAN, or static).

count—(Optional) Display the count of total subscribers and active subscribers for any specified option. You can use the **count** option alone or with the **address**, **client-type**, **interface**, **logical-system**, **mac-address**, **profile-name**, **routing-instance**, **stacked-vlan-id**, **subscriber-state**, or **vlan-id** options.

id—(Optional) Display a specific subscriber session whose session id matches the specified subscriber ID. You can display subscriber IDs by using the **show subscribers extensive** or the **show subscribers interface extensive** commands.

interface—(Optional) Display subscribers whose interface matches the specified interface.

logical-system—(Optional) Display subscribers whose logical system matches the specified logical system.

mac-address—(Optional) Display subscribers whose MAC address matches the specified MAC address.

physical-interface-name—(M120, M320, and MX Series routers only) (Optional) Display subscribers whose physical interface matches the specified physical interface.

profile-name—(Optional) Display subscribers whose dynamic profile matches the specified profile name.

routing-instance—(Optional) Display subscribers whose routing instance matches the specified routing instance.

subscriber-state—(Optional) Display subscribers whose subscriber state matches the specified subscriber state (ACTIVE, CONFIGURED, INIT, TERMINATED, or TERMINATING).

user-name—(M120, M320, and MX Series routers only) (Optional) Display subscribers whose username matches the specified subscriber name.

vci-identifier—(MX Series routers with MPCs and ATM MICs with SFP only) (Optional) Display active ATM subscribers whose ATM virtual circuit identifier (VCI) matches the specified VCI identifier. The range of values is **0** through **255**.

vpi-identifier—(MX Series routers with MPCs and ATM MICs with SFP only) (Optional) Display active ATM subscribers whose ATM virtual path identifier (VPI) matches the specified VPI identifier. The range of values is **0** through **65535**.

vlan-id—(Optional) Display subscribers whose VLAN ID matches the specified VLAN ID.

stacked-vlan-id—(Optional) Display subscribers whose stacked VLAN ID matches the specified stacked VLAN ID.

NOTE: Due to display limitations, logical system and routing instance output values are truncated when necessary.

Required Privilege Level view

Related Documentation

- show subscribers summary on page 228
- *Verifying and Managing Agent Circuit Identifier-Based Dynamic VLAN Configuration*

List of Sample Output

show subscribers vlan-id detail on page 227
show subscribers vpi vci extensive (PPPoE-over-ATM Subscriber Session) on page 227

Output Fields Table 14 on page 213 lists the output fields for the **show subscribers** command. Output fields are listed in the approximate order in which they appear.

Table 14: show subscribers Output Fields

Field Name	Field Description
Interface	Interface associated with the subscriber. The router or switch displays subscribers whose interface matches or begins with the specified interface. The * character indicates a continuation of addresses for the same session.
IP Address/VLAN ID	Subscriber IP address or VLAN ID associated with the subscriber in the form *tpid.vlan-id* No IP address or VLAN ID is assigned to an L2TP tunnel-switched session. For these subscriber sessions the value is **Tunnel-switched**.
User Name	Name of subscriber.
LS:RI	Logical system and routing instance associated with the subscriber.
Type	Subscriber client type (DHCP, L2TP, PPP, PPPoE, STATIC-INTERFACE, VLAN).
IP Address	Subscriber IPv4 address.
IP Netmask	Subscriber IP netmask.
Primary DNS Address	IP address of primary DNS server.
Secondary DNS Address	IP address of secondary DNS server.
Primary WINS Address	IP address of primary WINS server.
Secondary WINS Address	IP address of secondary WINS server.
IPv6 Address	Subscriber IPv6 address, or multiple addresses.
IPv6 Prefix	Subscriber IPv6 prefix. If you are using DHCPv6 prefix delegation, this is the delegated prefix.
IPv6 User Prefix	IPv6 prefix obtained through ND/RA.
IPv6 Address Pool	Subscriber IPv6 address pool. The IPv6 address pool is used to allocate IPv6 prefixes to the DHCPv6 clients.
IPv6 Network Prefix Length	Length of the network portion of the IPv6 address.
IPv6 Prefix Length	Length of the subscriber IPv6 prefix.

Table 14: show subscribers Output Fields *(continued)*

Field Name	Field Description
Logical System	Logical system associated with the subscriber.
Routing Instance	Routing instance associated with the subscriber.
Interface Type	Whether the subscriber interface is **Static** or **Dynamic**.
Interface Set	Internally generated name of the dynamic ACI interface set used by the subscriber session.
Interface Set Type	Interface type of the ACI interface set: **Dynamic**. This is the only ACI interface set type currently supported.
Interface Set Session ID	Identifier of the dynamic ACI interface set entry in the session database.
Underlying Interface	Name of the underlying interface for the subscriber session.
Dynamic Profile Name	Dynamic profile used for the subscriber.
Dynamic Profile Version	Version number of the dynamic profile used for the subscriber.
MAC Address	MAC address associated with the subscriber.
State	Current state of the subscriber session (**Init**, **Configured**, **Active**, **Terminating**, **Tunneled**).
L2TP State	Current state of the L2TP session, **Tunneled** or **Tunnel-switched**. When the value is **Tunnel-switched**, two entries are displayed for the subscriber; the first entry is at the LNS interface on the LTS and the second entry is at the LAC interface on the LTS.
Tunnel switch Profile Name	Name of the L2TP tunnel switch profile that initiates tunnel switching.
Local IP Address	IP address of the local gateway (LAC).
Remote IP Address	IP address of the remote peer (LNS).
VLAN Id	VLAN ID associated with the subscriber in the form *tpid.vlan-id*.
Stacked VLAN Id	Stacked VLAN ID associated with the subscriber in the form *tpid.vlan-id*.
RADIUS Accounting ID	RADIUS accounting ID associated with the subscriber.
Agent Circuit ID	Option 82 agent circuit ID associated with the subscriber. The ID is displayed as an ASCII string unless the value has nonprintable characters, in which case it is displayed in hexadecimal format.
Agent Remote ID	Option 82 agent remote ID associated with the subscriber. The ID is displayed as an ASCII string unless the value has nonprintable characters, in which case it is displayed in hexadecimal format.
DHCP Relay IP Address	IP address used by the DHCP relay agent.

Table 14: show subscribers Output Fields *(continued)*

Field Name	Field Description
ATM VPI	(MX Series routers with MPCs and ATM MICs with SFP only) ATM virtual path identifier (VPI) on the subscriber's physical interface.
ATM VCI	(MX Series routers with MPCs and ATM MICs with SFP only) ATM virtual circuit identifier (VCI) for each VPI configured on the subscriber interface.
Login Time	Date and time at which the subscriber logged in.
Effective shaping-rate	Actual downstream traffic shaping rate for the subscriber, in kilobits per second.
IPv4 rpf-check Fail Filter Name	Name of the filter applied by the dynamic profile to IPv4 packets that fail the RPF check.
IPv6 rpf-check Fail Filter Name	Name of the filter applied by the dynamic profile to IPv6 packets that fail the RPF check.
DHCP Options	**len** = number of hex values in the message. The hex values specify the type, length, value (TLV) for DHCP options, as defined in RFC 2132.
Session ID	ID number for a subscriber service session.
Underlying Session ID	For DHCPv6 subscribers on a PPPoE network, displays the session ID of the underlying PPPoE interface.
Service Sessions	Number of service sessions (that is, a service activated using RADIUS CoA) associated with the subscribers.
Service Session Name	Service session profile name.
Session Timeout (seconds)	Number of seconds of access provided to the subscriber before the session is automatically terminated.
Idle Timeout (seconds)	Number of seconds subscriber can be idle before the session is automatically terminated.
IPv6 Delegated Address Pool	Name of the pool used for DHCPv6 prefix delegation.
IPv6 Delegated Network Prefix Length	Length of the prefix configured for the IPv6 delegated address pool.
IPv6 Interface Address	Address assigned by the Framed-Ipv6-Prefix AAA attribute.
IPv6 Framed Interface Id	Interface ID assigned by the Framed-Interface-Id AAA attribute.
ADF IPv4 Input Filter Name	Name assigned to the Ascend-Data-Filter (ADF) interface IPv4 input filter (client or service session). The filter name is followed by the rules (in hexadecimal format) associated with the ADF filter and the decoded rule in Junos OS filter style.

Table 14: show subscribers Output Fields *(continued)*

Field Name	Field Description
ADF IPv4 Output Filter Name	Name assigned to the Ascend-Data-Filter (ADF) interface IPv4 output filter (client or service session). The filter name is followed by the rules (in hexadecimal format) associated with the ADF filter and the decoded rule in Junos OS filter style.
ADF IPv6 Input Filter Name	Name assigned to the Ascend-Data-Filter (ADF) interface IPv6 input filter (client or service session). The filter name is followed by the rules (in hexadecimal format) associated with the ADF filter and the decoded rule in Junos OS filter style.
ADF IPv6 Output Filter Name	Name assigned to the Ascend-Data-Filter (ADF) interface IPv6 output filter (client or service session). The filter name is followed by the rules (in hexadecimal format) associated with the ADF filter and the decoded rule in Junos OS filter style.
IPv4 Input Filter Name	Name assigned to the IPv4 input filter (client or service session).
IPv4 Output Filter Name	Name assigned to the IPv4 output filter (client or service session).
IPv6 Input Filter Name	Name assigned to the IPv6 input filter (client or service session).
IPv6 Output Filter Name	Name assigned to the IPv6 output filter (client or service session).
IFL Input Filter Name	Name assigned to the logical interface input filter (client or service session).
IFL Output Filter Name	Name assigned to the logical interface output filter (client or service session).

Sample Output

show subscribers (IPv4)

```
user@host> show subscribers
Interface               IP Address/VLAN ID   User Name         LS:RI
ge-1/3/0.1073741824     100                                    default:default
demux0.1073741824       100.0.0.10           WHOLESALER-CLIENT default:default
demux0.1073741825       101.0.0.3            RETAILER1-CLIENT  test1:retailer1
demux0.1073741826       102.0.0.3            RETAILER2-CLIENT  test1:retailer2
```

show subscribers (IPv6)

```
user@host> show subscribers
Interface          IP Address/VLAN ID   User Name           LS:RI
ge-1/0/0.0         2001::c0:0:0:0/74    WHOLESALER-CLIENT   default:default
     *             2002::1/128          subscriber-25       default:default
```

show subscribers (IPv4 and IPv6 Dual Stack)

```
user@host> show subscribers
Interface           IP Address/VLAN ID         User Name
LS:RI
demux0.1073741834   0x8100.1002 0x8100.1
default:default
demux0.1073741835   0x8100.1001 0x8100.1
default:default
pp0.1073741836      61.1.1.1                   dualstackuser1@ISP1.com
```

```
default:ASP-1
*                   2041:1:1::/48
*                   2061:1:1:1::/64
pp0.1073741837      23.1.1.3                dualstackuser2@ISP1.com
default:ASP-1
*                   2001:1:2:5::/64
```

show subscribers (LNS on MX Series Routers)

```
user@host> show subscribers
Interface          IP Address/VLAN ID   User Name         LS:RI
si-4/0/0.1         192.168.4.1          xyz@example.com   default:default
```

show subscribers (L2TP Switched Tunnels)

```
user@host> show subscribers
Interface           IP Address/VLAN ID    User Name             LS:RI
si-2/1/0.1073741842 Tunnel-switched       ap@lts.com        default:default

si-2/1/0.1073741843 Tunnel-switched       ap@lts.com        default:default
```

show subscribers client-type dhcp detail

```
user@host> show subscribers client-type dhcp detail
Type: DHCP
IP Address: 100.20.9.7
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: demux0.1073744127
Interface type: Dynamic
Dynamic Profile Name: dhcp-demux-prof
MAC Address: 00:10:95:00:00:98
State: Active
Radius Accounting ID: jnpr :2304
Login Time: 2009-08-25 14:43:52 PDT

Type: DHCP
IP Address: 100.20.10.7
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: demux0.1073744383
Interface type: Dynamic
Dynamic Profile Name: dhcp-demux-prof
MAC Address: 00:10:94:00:01:f3
State: Active
Radius Accounting ID: jnpr :2560
Login Time: 2009-08-25 14:43:56 PDT
```

show subscribers count

```
user@host> show subscribers count
Total Subscribers: 188, Active Subscribers: 188
```

show subscribers address detail (IPv6)

```
user@host> show subscribers address 100.16.12.137 detail
```

```
Type: PPPoE
User Name: pppoeTerV6User1Svc
IP Address: 100.16.12.137
IP Netmask: 255.0.0.0
IPv6 User Prefix: 1016:0:0:c88::/64
Logical System: default
Routing Instance: default
Interface: pp0.1073745151
Interface type: Dynamic
Underlying Interface: demux0.8201
Dynamic Profile Name: pppoe-client-profile
MAC Address: 00:0d:02:01:00:01
Session Timeout (seconds): 31622400
Idle Timeout (seconds): 86400
State: Active
Radius Accounting ID: jnpr demux0.8201:6544
Session ID: 6544
Agent Circuit ID: ifl3720
Agent Remote ID: ifl3720
Login Time: 2012-05-21 13:37:27 PDT
Service Sessions: 1
```

show subscribers detail (IPv4)

```
user@host> show subscribers detail
Type: DHCP
IP Address: 100.20.9.7
IP Netmask: 255.255.0.0
Primary DNS Address: 192.168.17.1
Secondary DNS Address: 192.168.17.2
Primary WINS Address: 192.168.22.1
Secondary WINS Address: 192.168.22.2
Logical System: default
Routing Instance: default
Interface: demux0.1073744127
Interface type: Dynamic
Dynamic Profile Name: dhcp-demux-prof
MAC Address: 00:10:95:00:00:98
State: Active
Radius Accounting ID: jnpr :2304
Session Timeout (seconds): 3600
Idle Timeout (seconds): 600
Login Time: 2009-08-25 14:43:52 PDT
DHCP Options: len 52
35 01 01 39 02 02 40 3d 07 01 00 10 94 00 00 08 33 04 00 00
00 3c 0c 15 63 6c 69 65 6e 74 5f 50 6f 72 74 20 2f 2f 36 2f
33 2d 37 2d 30 37 05 01 06 0f 21 2c
Service Sessions: 2
```

show subscribers detail (IPv6)

```
user@host> show subscribers detail
Type: DHCP
User Name: pd-user1
IPv6 Prefix: 2002:db2:ffff:1::/64
Logical System: default
Routing Instance: default
Interface: ge-3/1/3.2
Interface type: Static
MAC Address: 00:51:ff:ff:00:03
State: Active
```

```
Radius Accounting ID: 1
Session ID: 1
Login Time: 2011-08-25 12:12:26 PDT
DHCP Options: len 42
00 08 00 02 00 00 00 01 00 0a 00 03 00 01 00 51 ff ff 00 03
00 06 00 02 00 19 00 19 00 0c 00 00 00 00 00 00 00 00 00 00
00 00
```

show subscribers detail (IPv6 Static Demux Interface)

```
user@host> show subscribers detail
Type: STATIC-INTERFACE
User Name: demux0.1@jnpr.net
IPv6 Prefix: 1:2:3:4:5:6:7:aa/128
Logical System: default
Routing Instance: default
Interface: demux0.1
Interface type: Static
Dynamic Profile Name: junos-default-profile
State: Active
Radius Accounting ID: 185
Login Time: 2010-05-18 14:33:56 EDT
```

show subscribers detail (L2TP LNS Subscribers on MX Series Routers)

```
user@host> show subscribers detail
Type: L2TP
User Name: user1@jnpr.net
IP Address: 10.1.32.58
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: si-5/2/0.1073749824
Interface type: Dynamic
Dynamic Profile Name: dyn-lns-profile2
Dynamic Profile Version: 1
State: Active
Radius Accounting ID: 8001
Session ID: 8001
Login Time: 2011-04-25 20:27:50 IST
```

show subscribers detail (L2TP Switched Tunnels)

```
user@host> show subscribers detail
Type: L2TP
User Name: ap@example.com
Logical System: default
Routing Instance: default
Interface: si-2/1/0.1073741842
Interface type: Dynamic
Dynamic Profile Name: dyn-lts-profile
State: Active
L2TP State: Tunnel-switched
Tunnel switch Profile Name: ce-lts-profile
Local IP Address: 10.50.1.1
Remote IP Address: 192.168.20.3
Radius Accounting ID: 21
Session ID: 21
Login Time: 2013-01-18 03:01:11 PST

Type: L2TP
User Name: ap@example.com
```

```
Logical System: default
Routing Instance: default
Interface: si-2/1/0.1073741843
Interface type: Dynamic
Dynamic Profile Name: dyn-lts-profile
State: Active
L2TP State: Tunnel-switched
Tunnel switch Profile Name: ce-lts-profile
Local IP Address: 10.30.1.1
Remote IP Address: 172.20.1.10
Session ID: 22
Login Time: 2013-01-18 03:01:14 PST
```

show subscribers detail (Tunneled Subscriber)

```
user@host> show subscribers detail
Type: PPPoE
User Name: user1@example.com
Logical System: default
Routing Instance: default
Interface: pp0.1
State: Active, Tunneled
Radius Accounting ID: 512
```

show subscribers detail (IPv4 and IPv6 Dual Stack)

```
user@host> show subscribers detail
Type: VLAN
Logical System: default
Routing Instance: default
Interface: demux0.1073741824
Interface type: Dynamic
Dynamic Profile Name: svlanProfile
State: Active
Session ID: 1
Stacked VLAN Id: 0x8100.1001
VLAN Id: 0x8100.1
Login Time: 2011-11-30 00:18:04 PST

Type: PPPoE
User Name: dualstackuser1@ISP1.com
IP Address: 61.1.1.1
IPv6 Prefix: 2041:1:1::/48
IPv6 User Prefix: 2061:1:1:1::/64
Logical System: default
Routing Instance: ASP-1
Interface: pp0.1073741825
Interface type: Dynamic
Dynamic Profile Name: dualStack-Profile1
MAC Address: 00:00:64:03:01:02
State: Active
Radius Accounting ID: 2
Session ID: 2
Login Time: 2011-11-30 00:18:05 PST

Type: DHCP
IPv6 Prefix: 2041:1:1::/48
Logical System: default
Routing Instance: ASP-1
Interface: pp0.1073741825
Interface type: Static
```

```
MAC Address: 00:00:64:03:01:02
State: Active
Radius Accounting ID: jnpr :3
Session ID: 3
Underlying Session ID: 2
Login Time: 2011-11-30 00:18:35 PST
DHCP Options: len 42
00 08 00 02 0b b8 00 01 00 0a 00 03 00 01 00 00 64 03 01 02
00 06 00 02 00 19 00 19 00 0c 00 00 00 00 00 00 00 00 00 00
00 00
```

show subscribers detail (ACI Interface Set Session)

```
user@host> show subscribers detail
Type: VLAN
Logical System: default
Routing Instance: default
Interface: ge-1/0/0
Interface Set: aci-1001-ge-1/0/0.2800
Interface Set Session ID: 0
Underlying Interface: ge-1/0/0.2800
Dynamic Profile Name: aci-vlan-set-profile-2
Dynamic Profile Version: 1
State: Active
Session ID: 1
Agent Circuit ID: aci-ppp-dhcp-20
Login Time: 2012-05-26 01:54:08 PDT
```

show subscribers detail (PPPoE Subscriber Session with ACI Interface Set)

```
user@host> show subscribers detail
Type: PPPoE
User Name: ppphint2
IP Address: 10.10.1.5
Logical System: default
Routing Instance: default
Interface: pp0.1073741825
Interface type: Dynamic
Interface Set: aci-1001-demux0.1073741824
Interface Set Type: Dynamic
Interface Set Session ID: 2
Underlying Interface: demux0.1073741824
Dynamic Profile Name: aci-vlan-pppoe-profile
Dynamic Profile Version: 1
MAC Address: 00:00:64:39:01:02
State: Active
Radius Accounting ID: 3
Session ID: 3
Agent Circuit ID: aci-ppp-dhcp-dvlan-50
Login Time: 2012-03-07 13:46:53 PST
```

show subscribers extensive

```
user@host> show subscribers extensive
Type: DHCP
User Name: pd-user1
IPv6 Prefix: 2002:db2:ffff:1::/64
Logical System: default
Routing Instance: default
Interface: ge-3/1/3.2
Interface type: Static
```

```
MAC Address: 00:51:ff:ff:00:03
State: Active
Radius Accounting ID: 1
Session ID: 1
Login Time: 2011-08-25 12:12:26 PDT
DHCP Options: len 42
00 08 00 02 00 00 00 01 00 0a 00 03 00 01 00 51 ff ff 00 03
00 06 00 02 00 19 00 19 00 0c 00 00 00 00 00 00 00 00 00 00
00 00
IPv6 Address Pool: pd_pool
IPv6 Network Prefix Length: 48
```

show subscribers extensive (RPF Check Fail Filter)

```
user@host> show subscribers extensive
...
Type: VLAN
  Logical System: default
  Routing Instance: default
  Interface: ae0.1073741824
  Interface type: Dynamic
  Dynamic Profile Name: vlan-prof
  State: Active
  Session ID: 9
  VLAN Id: 100
  Login Time: 2011-08-26 08:17:00 PDT
  IPv4 rpf-check Fail Filter Name: rpf-allow-dhcp
  IPv6 rpf-check Fail Filter Name: rpf-allow-dhcpv6
...
```

show subscribers extensive (L2TP LNS Subscribers on MX Series Routers)

```
user@host> show subscribers extensive
Type: L2TP
User Name: user1@jnpr.net
IP Address: 10.1.32.58
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: si-5/2/0.1073749824
Interface type: Dynamic
Dynamic Profile Name: dyn-lns-profile2
Dynamic Profile Version: 1
State: Active
Radius Accounting ID: 8001
Session ID: 8001
Login Time: 2011-04-25 20:27:50 IST
IPv4 Input Filter Name: classify-si-5/2/0.1073749824-in
IPv4 Output Filter Name: classify-si-5/2/0.1073749824-out
```

show subscribers extensive (IPv4 and IPv6 Dual Stack)

```
user@host> show subscribers extensive
Type: VLAN
Logical System: default
Routing Instance: default
Interface: demux0.1073741824
Interface type: Dynamic
Dynamic Profile Name: svlanProfile
State: Active
Session ID: 1
Stacked VLAN Id: 0x8100.1001
```

```
VLAN Id: 0x8100.1
Login Time: 2011-11-30 00:18:04 PST

Type: PPPoE
User Name: dualstackuser1@ISP1.com
IP Address: 61.1.1.1
IPv6 Prefix: 2041:1:1::/48
IPv6 User Prefix: 2061:1:1:1::/64
Logical System: default
Routing Instance: ASP-1
Interface: pp0.1073741825
Interface type: Dynamic
Dynamic Profile Name: dualStack-Profile1
MAC Address: 00:00:64:03:01:02
State: Active
Radius Accounting ID: 2
Session ID: 2
Login Time: 2011-11-30 00:18:05 PST
IPv6 Delegated Network Prefix Length: 48
IPv6 Interface Address: 2061:1:1:1::1/64
IPv6 Framed Interface Id: 1:1:2:2
IPv4 Input Filter Name: FILTER-IN-pp0.1073741825-in
IPv4 Output Filter Name: FILTER-OUT-pp0.1073741825-out
IPv6 Input Filter Name: FILTER-IN6-pp0.1073741825-in
IPv6 Output Filter Name: FILTER-OUT6-pp0.1073741825-out

Type: DHCP
IPv6 Prefix: 2041:1:1::/48
Logical System: default
Routing Instance: ASP-1
Interface: pp0.1073741825
Interface type: Static
MAC Address: 00:00:64:03:01:02
State: Active
Radius Accounting ID: jnpr :3
Session ID: 3
Underlying Session ID: 2
Login Time: 2011-11-30 00:18:35 PST
DHCP Options: len 42
00 08 00 02 0b b8 00 01 00 0a 00 03 00 01 00 00 64 03 01 02
00 06 00 02 00 19 00 19 00 0c 00 00 00 00 00 00 00 00 00 00
00 00
IPv6 Delegated Network Prefix Length: 48
```

show subscribers extensive (Effective Shaping-Rate)

```
user@host> show subscribers extensive
Type: VLAN
Logical System: default
Routing Instance: default
Interface: demux0.1073741837
Interface type: Dynamic
Interface Set: ifset-1
Underlying Interface: ae1
Dynamic Profile Name: svlan-dhcp-test
State: Active
Session ID: 1
Stacked VLAN Id: 0x8100.201
VLAN Id: 0x8100.201
Login Time: 2011-11-30 00:18:04 PST
```

```
Effective shaping-rate: 31000000k
...
```

show subscribers aci-interface-set-name detail (Subscriber Sessions Using Specified ACI Interface Set)

```
user@host> show subscribers aci-interface-set-name aci-1003-ge-1/0/0.4001 detail
Type: VLAN
Logical System: default
Routing Instance: default
Interface: ge-1/0/0.
Underlying Interface: ge-1/0/0.4001
Dynamic Profile Name: aci-vlan-set-profile
Dynamic Profile Version: 1
State: Active
Session ID: 13
Agent Circuit ID: aci-ppp-vlan-10
Login Time: 2012-03-12 10:41:56 PDT

Type: PPPoE
User Name: ppphint2
IP Address: 10.10.1.7
Logical System: default
Routing Instance: default
Interface: pp0.1073741834
Interface type: Dynamic
Interface Set: aci-1003-ge-1/0/0.4001
Interface Set Type: Dynamic
Interface Set Session ID: 13
Underlying Interface: ge-1/0/0.4001
Dynamic Profile Name: aci-vlan-pppoe-profile
Dynamic Profile Version: 1
MAC Address: 00:00:65:26:01:02
State: Active
Radius Accounting ID: 14
Session ID: 14
Agent Circuit ID: aci-ppp-vlan-10
Login Time: 2012-03-12 10:41:57 PDT
```

show subscribers agent-circuit-identifier detail (Subscriber Sessions Using Specified ACI Substring)

```
user@host> show subscribers agent-circuit-identifier aci-ppp-vlan detail
Type: VLAN
Logical System: default
Routing Instance: default
Interface: ge-1/0/0.
Underlying Interface: ge-1/0/0.4001
Dynamic Profile Name: aci-vlan-set-profile
Dynamic Profile Version: 1
State: Active
Session ID: 13
Agent Circuit ID: aci-ppp-vlan-10
Login Time: 2012-03-12 10:41:56 PDT

Type: PPPoE
User Name: ppphint2
IP Address: 10.10.1.7
Logical System: default
Routing Instance: default
Interface: pp0.1073741834
Interface type: Dynamic
Interface Set: aci-1003-ge-1/0/0.4001
```

```
Interface Set Type: Dynamic
Interface Set Session ID: 13
Underlying Interface: ge-1/0/0.4001
Dynamic Profile Name: aci-vlan-pppoe-profile
Dynamic Profile Version: 1
MAC Address: 00:00:65:26:01:02
State: Active
Radius Accounting ID: 14
Session ID: 14
Agent Circuit ID: aci-ppp-vlan-10
Login Time: 2012-03-12 10:41:57 PDT
```

show subscribers interface extensive

```
user@host> show subscribers interface demux0.1073741826 extensive
Type: VLAN
User Name: test1@test.com
Logical System: default
Routing Instance: testnet
Interface: demux0.1073741826
Interface type: Dynamic
Dynamic Profile Name: profile-vdemux-relay-23qos
MAC Address: 00:00:6e:56:01:04
State: Active
Radius Accounting ID: 12
Session ID: 12
Stacked VLAN Id: 0x8100.1500
VLAN Id: 0x8100.2902
Login Time: 2011-10-20 16:21:59 EST

Type: DHCP
User Name: test1@test.com
IP Address: 172.16.200.6
IP Netmask: 255.255.255.0
Logical System: default
Routing Instance: testnet
Interface: demux0.1073741826
Interface type: Static
MAC Address: 00:00:6e:56:01:04
State: Active
Radius Accounting ID: 21
Session ID: 21
Login Time: 2011-10-20 16:24:33 EST
Service Sessions: 2

Service Session ID: 25
Service Session Name: SUB-QOS
State: Active

Service Session ID: 26
Service Session Name: service-cb-content
State: Active
IPv4 Input Filter Name: content-cb-in-demux0.1073741826-in
IPv4 Output Filter Name: content-cb-out-demux0.1073741826-out
```

show subscribers logical-system terse

```
user@host> show subscribers logical-system test1 terse
```

```
Interface           IP Address/VLAN ID   User Name         LS:RI
demux0.1073741825     101.0.0.3            RETAILER1-CLIENT  test1:retailer1
demux0.1073741826     102.0.0.3            RETAILER2-CLIENT  test1:retailer2
```

show subscribers physical-interface count

```
user@host> show subscribers physical-interface ge-1/0/0 count
Total subscribers: 3998, Active Subscribers: 3998
```

show subscribers routing-instance inst1 count

```
user@host> show subscribers routing-instance inst1 count
Total Subscribers: 188, Active Subscribers: 183
```

show subscribers stacked-vlan-id detail

```
user@host> show subscribers stacked-vlan-id 101 detail
Type: VLAN
Interface: ge-1/2/0.1073741824
Interface type: Dynamic
Dynamic Profile Name: svlan-prof
State: Active
Stacked VLAN Id: 0x8100.101
VLAN Id: 0x8100.100
Login Time: 2009-03-27 11:57:19 PDT
```

show subscribers stacked-vlan-id vlan-id detail (Combined Output)

```
user@host> show subscribers stacked-vlan-id 101 vlan-id 100 detail
Type: VLAN
Interface: ge-1/2/0.1073741824
Interface type: Dynamic
Dynamic Profile Name: svlan-prof
State: Active
Stacked VLAN Id: 0x8100.101
VLAN Id: 0x8100.100
Login Time: 2009-03-27 11:57:19 PDT
```

show subscribers stacked-vlan-id vlan-id interface detail (Combined Output for a Specific Interface)

```
user@host> show subscribers stacked-vlan-id 101 vlan-id 100 interface ge-1/2/0.* detail
Type: VLAN
Interface: ge-1/2/0.1073741824
Interface type: Dynamic
Dynamic Profile Name: svlan-prof
State: Active
Stacked VLAN Id: 0x8100.101
VLAN Id: 0x8100.100
Login Time: 2009-03-27 11:57:19 PDT
```

show subscribers user-name detail

```
user@host> show subscribers user-name larry1 detail
Type: DHCP
User Name: larry1
IP Address: 100.0.0.37
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: ge-1/0/0.1
Interface type: Static
Dynamic Profile Name: foo
```

```
MAC Address: 00:10:94:00:00:01
State: Active
Radius Accounting ID: 1
Session ID: 1
Login Time: 2011-11-07 08:25:59 PST
DHCP Options: len 52
35 01 01 39 02 02 40 3d 07 01 00 10 94 00 00 01 33 04 00 00
00 3c 0c 15 63 6c 69 65 6e 74 5f 50 6f 72 74 20 2f 2f 32 2f
37 2d 30 2d 30 37 05 01 06 0f 21 2c
```

show subscribers vlan-id

```
user@host> show subscribers vlan-id 100
Interface           IP Address                              User Name
ge-1/0/0.1073741824
ge-1/2/0.1073741825
```

show subscribers vlan-id detail

```
user@host> show subscribers vlan-id 100 detail
Type: VLAN
Interface: ge-1/0/0.1073741824
Interface type: Dynamic
Dynamic Profile Name: vlan-prof-tpid
State: Active
VLAN Id: 100
Login Time: 2009-03-11 06:48:54 PDT

Type: VLAN
Interface: ge-1/2/0.1073741825
Interface type: Dynamic
Dynamic Profile Name: vlan-prof-tpid
State: Active
VLAN Id: 100
Login Time: 2009-03-11 06:48:54 PDT
```

show subscribers vpi vci extensive (PPPoE-over-ATM Subscriber Session)

```
user@host> show subscribers vpi 40 vci 50 extensive
Type: PPPoE
User Name: testuser
IP Address: 100.0.0.2
IP Netmask: 255.255.0.0
Logical System: default
Routing Instance: default
Interface: pp0.0
Interface type: Static
MAC Address: 00:00:65:23:01:02
State: Active
Radius Accounting ID: 2
Session ID: 2
ATM VPI: 40
ATM VCI: 50
Login Time: 2012-12-03 07:49:26 PST
IP Address Pool: pool_1
IPv6 Framed Interface Id: 200:65ff:fe23:102
```

show subscribers summary

Syntax

show subscribers summary
< detail | extensive | terse>
<count>
physical-interface *physical-interface-name*
<all | logical-system *logical-system* pic | port | routing-instance *routing-instance* |slot>

Release Information

Command introduced in Junos OS Release 10.2.

Description

Display summary information for subscribers.

Options

detail | extensive | terse—(Optional) Display the specified level of output.

count—(Optional) Display the count of total subscribers and active subscribers for any specified option.

logical-system—(Optional) Display subscribers whose logical system matches the specified logical system.

physical-interface-name—(M120, M320, and MX Series routers only) (Optional) Display a count of subscribers whose physical interface matches the specified physical interface, by subscriber state, client type and LS:RI.

pic—(M120, M320, and MX Series routers only) (Optional) Display a count of subscribers by PIC number and the total number of subscribers.

port—(M120, M320, and MX Series routers only) (Optional) Display a count of subscribers by port number and the total number of subscribers.

routing-instance—(Optional) Display subscribers whose routing instance matches the specified routing instance.

slot—(M120, M320, and MX Series routers only) (Optional) Display a count of subscribers by FPC slot number and the total number of subscribers.

NOTE: **Due to display limitations, logical system and routing instance output values are truncated when necessary.**

Required Privilege Level

view

Related Documentation

- show subscribers on page 210

List of Sample Output

show subscribers summary on page 230
show subscribers summary all on page 230
show subscribers summary physical-interface on page 230
show subscribers summary physical-interface pic on page 231

show subscribers summary physical-interface port on page 231
show subscribers summary physical-interface slot on page 231
show subscribers summary pic on page 231
show subscribers summary pic (Aggregated Ethernet Interfaces) on page 232
show subscribers summary port on page 232
show subscribers summary slot on page 232
show subscribers summary terse on page 232

Output Fields Table 15 on page 229 lists the output fields for the **show subscribers** command. Output fields are listed in the approximate order in which they appear.

Table 15: show subscribers Output Fields

Field Name	Field Description
Subscribers by State	Number of subscribers summarized by state. The summary information includes the following: • Init—Number of subscriber currently in the initialization state. • Configured—Number of configured subscribers. • Active—Number of active subscribers. • Terminating—Number of subscribers currently terminating. • Terminated—Number of terminated subscribers. • Total—Total number of subscribers for all states.
Subscribers by Client Type	Number of subscribers summarized by client type. Client types can include DHCP, L2TP, PPP, PPPOE, STATIC-INTERFACE, and VLAN. Also displays the total number of subscribers for all client types (Total).
Subscribers by LS:RI	Number of subscribers summarized by logical system:routing instance (LS:RI) combination. Also displays the total number of subscribers for all LS:RI combinations (Total).
Interface	Interface associated with the subscriber. The router or switch displays subscribers whose interface matches or begins with the specified interface. The * character indicates a continuation of addresses for the same session. For aggregated Ethernet interfaces, the output of the **summary (pic | port | slot)** options prefixes the interface name with **ae0:**.
Count	Count of subscribers displayed for each PIC, port, or slot when those options are specified with the **summary** option. For an aggregated Ethernet configuration, the total subscriber count does not equal the sum of the individual PIC, port, or slot counts, because each subscriber can be in more than one aggregated Ethernet link.
Total Subscribers	Total number of subscribers for all physical interfaces, all PICS, all ports, or all LS:RI slots.
IP Address/VLAN ID	Subscriber IP address or VLAN ID associated with the subscriber in the form *tpid.vlan-id*
User Name	Name of subscriber.
LS:RI	Logical system and routing instance associated with the subscriber.

Sample Output

show subscribers summary

```
user@host> show subscribers summary

Subscribers by State
 Init      3
 Configured    2
 Active   183
 Terminating    2
 Terminated    1

 TOTAL        191

Subscribers by Client Type
 DHCP     107
 PPP       76
 VLAN       8

 TOTAL        191
```

show subscribers summary all

```
user@host> show subscribers summary all
Subscribers by State
 Init         3
 Configured       2
 Active      183
 Terminating       2
 Terminated       1

 TOTAL          191

Subscribers by Client Type
 DHCP      107
 PPP          76
 VLAN          8

 TOTAL         191

Subscribers by LS:RI
 default:default   1
 default:ri1     28
 default:ri2     16
 ls1:default    22
 ls1:riA     38
 ls1:riB     44
 logsysX:routinstY  42

 TOTAL     191
```

show subscribers summary physical-interface

```
user@host> show subscribers summary physical-interface ge-1/0/0
Subscribers by State
   Active: 3998
   Total: 3998

Subscribers by Client Type
   DHCP: 3998
```

```
   Total: 3998

Subscribers by LS:RI
   default:default: 3998
   Total: 3998
```

show subscribers summary physical-interface pic

```
user@host> show subscribers summary physical-interface ge-0/2/0 pic
Subscribers by State
   Active: 4825
   Total: 4825

Subscribers by Client Type
   DHCP: 4825
   Total: 4825

Subscribers by LS:RI
   default:default: 4825
   Total: 4825
```

show subscribers summary physical-interface port

```
user@host> show subscribers summary physical-interface ge-0/3/0 port
Subscribers by State
   Active: 4825
   Total: 4825

Subscribers by Client Type
   DHCP: 4825
   Total: 4825

Subscribers by LS:RI
   default:default: 4825
   Total: 4825
```

show subscribers summary physical-interface slot

```
user@host> show subscribers summary physical-interface ge-2/0/0 slot
Subscribers by State
   Active: 4825
   Total: 4825

Subscribers by Client Type
   DHCP: 4825
   Total: 4825

Subscribers by LS:RI
   default:default: 4825
   Total: 4825
```

show subscribers summary pic

```
user@host> show subscribers summary pic
Interface           Count
ge-1/0              1000
ge-1/3              1000

Total Subscribers: 2000
```

show subscribers summary pic (Aggregated Ethernet Interfaces)

```
user@host> show subscribers summary pic
Interface           Count
ae0: ge-1/0         801
ae0: ge-1/3         801

Total Subscribers: 801
```

show subscribers summary port

```
user@host> show subscribers summary port
Interface           Count
ge-1                2000

Total Subscribers: 2000
```

show subscribers summary slot

```
user@host> show subscribers summary slot
Interface           Count
ge-1                2000

Total Subscribers: 2000
```

show subscribers summary terse

```
user@host> show subscribers summary terse
Interface           IP Address/VLAN ID   User Name         LS:RI
ge-1/3/0.1073741824   100                                   default:default
demux0.1073741824     100.0.0.10           WHOLESALER-CLIENT default:default
demux0.1073741825     101.0.0.3            RETAILER1-CLIENT  test1:retailer1
demux0.1073741826     102.0.0.3            RETAILER2-CLIENT  test1:retailer2
```

PART 4

Troubleshooting

CHAPTER 14

Acquiring Troubleshooting Information

Collecting Subscriber Access Logs Before Contacting Juniper Technical Support

Problem When you experience a subscriber access problem in your network, we recommend that you collect certain logs before you contact Juniper Technical Support. This topic shows you the most useful logs for a variety of network implementations. In addition to the relevant log information, you must also collect standard troubleshooting information and send it to Juniper Technical Support in your request for assistance.

Solution To collect standard troubleshooting information:

- Redirect the command output to a file.

  user@host> **request support information | save rsi-1**

To configure logging to assist Juniper Technical Support:

1. Review the following blocks of statements to determine which apply to your configuration.

   ```
   [edit]
   set system syslog archive size 100m files 25
   set system auto-configuration traceoptions file filename
   set system auto-configuration traceoptions file filename size 100m files 25
   set protocols ppp-service traceoptions file filename size 100m files 25
   set protocols ppp-service traceoptions level all
   set protocols ppp-service traceoptions flag all
   set protocols ppp traceoptions file filename size 100m files 25
   set protocols ppp traceoptions level all
   set protocols ppp traceoptions flag all
   set protocols ppp monitor-session all
   set interfaces pp0 traceoptions flag all
   set demux traceoptions file filename size 100m files 25
   set demux traceoptions level all
   set demux traceoptions flag all
   set system processes dhcp-service traceoptions file filename
   set system processes dhcp-service traceoptions file size 100m
   set system processes dhcp-service traceoptions file files 25
   set system processes dhcp-service traceoptions flag all
   set class-of-service traceoptions file filename
   set class-of-service traceoptions file size 100m
   set class-of-service traceoptions flag all
   set class-of-service traceoptions file files 25
   set routing-options traceoptions file filename
   set routing-options traceoptions file size 100m
   set routing-options traceoptions flag all
   set routing-options traceoptions file files 25
   set interfaces traceoptions file filename
   set interfaces traceoptions file size 100m
   set interfaces traceoptions flag all
   set interfaces traceoptions file files 25
   set system processes general-authentication-service traceoptions file filename
   set system processes general-authentication-service traceoptions file size 100m
   set system processes general-authentication-service traceoptions flag all
   set system processes general-authentication-service traceoptions file files 25
   ```

2. Copy the relevant statements into a text file and modify the log filenames as you want.
3. Copy the statements from the text file and paste them into the CLI on your router to configure logging.
4. Commit the logging configuration to begin collecting information.

NOTE: The maximum file size for DHCP local server and DHCP relay log files is 1 GB. The maximum number of log files for DHCP local server and DHCP relay is 1000.

BEST PRACTICE: Enable these logs only to collect information when troubleshooting specific problems. Enabling these logs during normal operations can result in reduced system performance.

Related Documentation

- *Compressing Troubleshooting Logs from /var/logs to Send to Juniper Technical Support*

PART 5

Index

Index

Symbols

A

B

C

D

T

U