



**JUNOS® Software**

## **Flow Monitoring Feature Guide**

*Release 10.0*

**Juniper Networks, Inc.**

1194 North Mathilda Avenue  
Sunnyvale, California 94089  
USA

408-745-2000

**[www.juniper.net](http://www.juniper.net)**

Published: 2009-10-08

This product includes the Envoy SNMP Engine, developed by Epilogue Technology, an Integrated Systems Company. Copyright © 1986-1997, Epilogue Technology Corporation. All rights reserved. This program and its documentation were developed at private expense, and no part of them is in the public domain.

This product includes memory allocation software developed by Mark Moraes, copyright © 1988, 1989, 1993, University of Toronto.

This product includes FreeBSD software developed by the University of California, Berkeley, and its contributors. All of the documentation and software included in the 4.4BSD and 4.4BSD-Lite Releases is copyrighted by the Regents of the University of California. Copyright © 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994. The Regents of the University of California. All rights reserved.

GateD software copyright © 1995, the Regents of the University. All rights reserved. Gate Daemon was originated and developed through release 3.0 by Cornell University and its collaborators. Gated is based on Kirton's EGP, UC Berkeley's routing daemon (routed), and DCN's HELLO routing protocol. Development of Gated has been supported in part by the National Science Foundation. Portions of the GateD software copyright © 1988, Regents of the University of California. All rights reserved. Portions of the GateD software copyright © 1991, D. L. S. Associates.

This product includes software developed by Maker Communications, Inc., copyright © 1996, 1997, Maker Communications, Inc.

Juniper Networks, the Juniper Networks logo, JUNOS, NetScreen, ScreenOS, and Steel-Belted Radius are registered trademarks of Juniper Networks, Inc. in the United States and other countries. JUNOSe is a trademark of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Products made or sold by Juniper Networks or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

*JUNOS® Software Flow Monitoring Feature Guide,*

Release 10.0

Copyright © 2009, Juniper Networks, Inc.

All rights reserved. Printed in USA.

Writing: Merisha Wazna, Fawn Damitio, Richard Hendricks, and Gary Matthews

Editing: Sonia Saruba

Illustration: Merisha Wazna, Faith Bradford, Fawn Damitio, Nathaniel Woodward, and Richard Hendricks

Cover Design: Edmonds Design

Revision History

October 2009—R1 JUNOS 10.0

The information in this document is current as of the date listed in the revision history.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. The JUNOS Software 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

**READ THIS END USER LICENSE AGREEMENT ("AGREEMENT") BEFORE DOWNLOADING, INSTALLING, OR USING THE SOFTWARE.** BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE OR OTHERWISE EXPRESSING YOUR AGREEMENT TO THE TERMS CONTAINED HEREIN, YOU (AS CUSTOMER OR IF YOU ARE NOT THE CUSTOMER, AS A REPRESENTATIVE/AGENT AUTHORIZED TO BIND THE CUSTOMER) CONSENT TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT OR CANNOT AGREE TO THE TERMS CONTAINED HEREIN, THEN (A) DO NOT DOWNLOAD, INSTALL, OR USE THE SOFTWARE, AND (B) YOU MAY CONTACT JUNIPER NETWORKS REGARDING LICENSE TERMS.

1. **The Parties.** The parties to this Agreement are (i) Juniper Networks, Inc. (if the Customer's principal office is located in the Americas) or Juniper Networks (Cayman) Limited (if the Customer's principal office is located outside the Americas) (such applicable entity being referred to herein as "Juniper"), and (ii) the person or organization that originally purchased from Juniper or an authorized Juniper reseller the applicable license(s) for use of the Software ("Customer") (collectively, the "Parties").

2. **The Software.** In this Agreement, "Software" means the program modules and features of the Juniper or Juniper-supplied software, for which Customer has paid the applicable license or support fees to Juniper or an authorized Juniper reseller, or which was embedded by Juniper in equipment which Customer purchased from Juniper or an authorized Juniper reseller. "Software" also includes updates, upgrades and new releases of such software. "Embedded Software" means Software which Juniper has embedded in or loaded onto the Juniper equipment and any updates, upgrades, additions or replacements which are subsequently embedded in or loaded onto the equipment.

3. **License Grant.** Subject to payment of the applicable fees and the limitations and restrictions set forth herein, Juniper grants to Customer a non-exclusive and non-transferable license, without right to sublicense, to use the Software, in executable form only, subject to the following use restrictions:

a. Customer shall use Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.

b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius or IMS AAA software on multiple computers or virtual machines (e.g., Solaris zones) requires multiple licenses, regardless of whether such computers or virtualizations are physically contained on a single chassis.

c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface, processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.

d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.

e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

The foregoing license is not transferable or assignable by Customer. No license is granted herein to any user who did not originally purchase the applicable license(s) for the Software from Juniper or an authorized Juniper reseller.

4. **Use Prohibitions.** Notwithstanding the foregoing, the license provided herein does not permit the Customer to, and Customer agrees not to and shall not: (a) modify, unbundle, reverse engineer, or create derivative works based on the Software; (b) make unauthorized copies of the Software (except as necessary for backup purposes); (c) rent, sell, transfer, or grant any rights in and to any copy of the Software, in any form, to any third party; (d) remove any proprietary notices, labels, or marks on or in any copy of the Software or any product in which the Software is embedded; (e) distribute any copy of the Software to any third party, including as may be embedded in Juniper equipment sold in the secondhand market; (f) use any 'locked' or key-restricted feature, function, service, application, operation, or capability without first purchasing the applicable license(s) and obtaining a valid key from Juniper, even if such feature, function, service, application, operation, or capability is enabled without a key; (g) distribute any key for the Software provided by Juniper to any third party; (h) use the Software in any manner that extends or is broader than the uses purchased by Customer from Juniper or an authorized Juniper reseller; (i) use Embedded Software on non-Juniper equipment; (j) use Embedded Software (or make it available for use) on Juniper equipment that the Customer did not originally purchase from Juniper or an authorized Juniper reseller; (k) disclose the results of testing or benchmarking of the Software to any third party without the prior written consent of Juniper; or (l) use the Software in any manner other than as expressly provided herein.

5. **Audit.** Customer shall maintain accurate records as necessary to verify compliance with this Agreement. Upon request by Juniper, Customer shall furnish such records to Juniper and certify its compliance with this Agreement.

6. **Confidentiality.** The Parties agree that aspects of the Software and associated documentation are the confidential property of Juniper. As such, Customer shall exercise all reasonable commercial efforts to maintain the Software and associated documentation in confidence, which at a minimum includes restricting access to the Software to Customer employees and contractors having a need to use the Software for Customer's internal business purposes.

7. **Ownership.** Juniper and Juniper's licensors, respectively, retain ownership of all right, title, and interest (including copyright) in and to the Software, associated documentation, and all copies of the Software. Nothing in this Agreement constitutes a transfer or conveyance of any right, title, or interest in the Software or associated documentation, or a sale of the Software, associated documentation, or copies of the Software.

8. **Warranty, Limitation of Liability, Disclaimer of Warranty.** The warranty applicable to the Software shall be as set forth in the warranty statement that accompanies the Software (the "Warranty Statement"). Nothing in this Agreement shall give rise to any obligation to support the Software. Support services may be purchased separately. Any such support shall be governed by a separate, written support services agreement. TO THE MAXIMUM EXTENT PERMITTED BY LAW, JUNIPER SHALL NOT BE LIABLE FOR ANY LOST PROFITS, LOSS OF DATA, OR COSTS OR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, THE SOFTWARE, OR ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. IN NO EVENT SHALL JUNIPER BE LIABLE FOR DAMAGES ARISING FROM UNAUTHORIZED OR IMPROPER USE OF ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE, EXCEPT AS EXPRESSLY PROVIDED IN THE WARRANTY STATEMENT TO THE EXTENT PERMITTED BY LAW, JUNIPER DISCLAIMS ANY AND ALL WARRANTIES IN AND TO THE SOFTWARE (WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE), INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT DOES JUNIPER WARRANT THAT THE SOFTWARE, OR ANY EQUIPMENT OR NETWORK RUNNING THE SOFTWARE, WILL OPERATE WITHOUT ERROR OR INTERRUPTION, OR WILL BE FREE OF VULNERABILITY TO INTRUSION OR ATTACK. In no event shall Juniper's or its suppliers' or licensors' liability to Customer, whether in contract, tort (including negligence), breach of warranty, or otherwise, exceed the price paid by Customer for the Software that gave rise to the claim, or if the Software is embedded in another Juniper product, the price paid by Customer for such other product. Customer acknowledges and agrees that Juniper has set its prices and entered into this Agreement in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the Parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the Parties.

9. **Termination.** Any breach of this Agreement or failure by Customer to pay any applicable fees due shall result in automatic termination of the license granted herein. Upon such termination, Customer shall destroy or return to Juniper all copies of the Software and related documentation in Customer's possession or control.

10. **Taxes.** All license fees payable under this agreement are exclusive of tax. Customer shall be responsible for paying Taxes arising from the purchase of the license, or importation or use of the Software. If applicable, valid exemption documentation for each taxing jurisdiction shall be provided to Juniper prior to invoicing, and Customer shall promptly notify Juniper if their exemption is revoked or modified. All payments made by Customer shall be net of any applicable withholding tax. Customer will provide reasonable assistance to Juniper in connection with such withholding taxes by promptly: providing Juniper with valid tax receipts and other required documentation showing Customer's payment of any withholding taxes; completing appropriate applications that would reduce the amount of withholding tax to be paid; and notifying and assisting Juniper in any audit or tax proceeding related to transactions hereunder. Customer shall comply with all applicable tax laws and regulations, and Customer will promptly pay or reimburse Juniper for all costs and damages related to any liability incurred by Juniper as a result of Customer's non-compliance or delay with its responsibilities herein. Customer's obligations under this Section shall survive termination or expiration of this Agreement.

11. **Export.** Customer agrees to comply with all applicable export laws and restrictions and regulations of any United States and any applicable foreign agency or authority, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. Customer shall be liable for any such violations. The version of the Software supplied to Customer may contain encryption or other capabilities restricting Customer's ability to export the Software without an export license.

12. **Commercial Computer Software.** The Software is "commercial computer software" and is provided with restricted rights. Use, duplication, or disclosure by the United States government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7201 through 227.7202-4, FAR 12.212, FAR 27.405(b)(2), FAR 52.227-19, or FAR 52.227-14(ALT III) as applicable.

13. **Interface Information.** To the extent required by applicable law, and at Customer's written request, Juniper shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of applicable fee, if any. Customer shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which Juniper makes such information available.

14. **Third Party Software.** Any licensor of Juniper whose software is embedded in the Software and any supplier of Juniper whose products or technology are embedded in (or services are accessed by) the Software shall be a third party beneficiary with respect to this Agreement, and such licensor or vendor shall have the right to enforce this Agreement in its own name as if it were Juniper. In addition, certain third party software may be provided with the Software and is subject to the accompanying license(s), if any, of its respective owner(s). To the extent portions of the Software are distributed under and subject to open source licenses obligating Juniper to make the source code for such portions publicly available (such as the GNU General Public License ("GPL") or the GNU Library General Public License ("LGPL")), Juniper will make such source code portions (including Juniper modifications, as appropriate) available upon request for a period of up to three years from the date of distribution. Such request can be made in writing to Juniper Networks, Inc., 1194 N. Mathilda Ave., Sunnyvale, CA 94089, ATTN: General Counsel. You may obtain a copy of the GPL at <http://www.gnu.org/licenses/gpl.html>, and a copy of the LGPL at <http://www.gnu.org/licenses/lgpl.html>.

15. **Miscellaneous.** This Agreement shall be governed by the laws of the State of California without reference to its conflicts of laws principles. The provisions of the U.N. Convention for the International Sale of Goods shall not apply to this Agreement. For any disputes arising under this Agreement, the Parties hereby consent to the personal and exclusive jurisdiction of, and venue in, the state and federal courts within Santa Clara County, California. This Agreement constitutes the entire and sole agreement between Juniper and the Customer with respect to the Software, and supersedes all prior and contemporaneous

agreements relating to the Software, whether oral or written (including any inconsistent terms contained in a purchase order), except that the terms of a separate written agreement executed by an authorized Juniper representative and Customer shall govern to the extent such terms are inconsistent or conflict with terms contained herein. No modification to this Agreement nor any waiver of any rights hereunder shall be effective unless expressly assented to in writing by the party to be charged. If any portion of this Agreement is held invalid, the Parties agree that such invalidity shall not affect the validity of the remainder of this Agreement. This Agreement and associated documentation has been written in the English language, and the Parties agree that the English version will govern. (For Canada: Les parties aux présentes confirment leur volonté que cette convention de même que tous les documents y compris tout avis qui s'y rattache, soient rédigés en langue anglaise. (Translation: The parties confirm that this Agreement and all related documentation is and will be in the English language)).



# Table of Contents

## Part 1

## Flow Monitoring

---

### Chapter 1

### Flow Monitoring Concepts and Reference Material

**3**

Overview of Flow Monitoring .....	3
Passive Flow Monitoring .....	4
Active Flow Monitoring .....	5
System Requirements .....	6
Passive Flow Monitoring System Requirements .....	6
Active Flow Monitoring System Requirements .....	7
Active Flow Monitoring PIC Specifications .....	8

### Chapter 2

### Passive Flow Monitoring

**13**

Configuring Passive Flow Monitoring .....	13
Monitoring Traffic with a VRF Instance and a Monitoring Group .....	14
Specifying a Firewall Filter to Select Traffic to Monitor .....	14
Configuring Input Interfaces, Monitoring Services Interfaces, and Export Interfaces .....	15
Establishing a VRF Instance for the Monitored Traffic .....	18
Configuring a Monitoring Group to Send Traffic to the Flow Server .....	19
Configuring Policy Options .....	20
Option: Stripping MPLS Labels on ATM, Ethernet-Based, and SONET/SDH Interfaces .....	21
Using Port Mirroring and Filter-Based Forwarding to Copy and Redirect Traffic .....	22
Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding .....	22
Specifying Port Mirroring Input and Output .....	23
Creating a Firewall Filter to Split the Port-Mirrored Traffic into Different Instances .....	24
Applying the Firewall Filter to a Tunnel PIC Interface .....	25
Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations .....	25
Configuring a Routing Table Group to Add Interface Routes into the Forwarding Instance .....	26

Option: Using an ES PIC to Send Traffic to a Packet Analyzer .....	26
Option: Applying a Firewall Filter to an Output Interface .....	28
Using a Flow Collector Interface to Process and Export Multiple Flow Records .....	28
Monitoring Traffic Using a Dynamic Flow Capture Interface .....	33
Using a Dynamic Flow Capture Interface to Monitor Traffic On Demand .....	34
Configuring the Capture Group .....	35
Configuring the Content Destination .....	36
Configuring the Control Source .....	36
Configuring the Dynamic Flow Capture Interface .....	37
Option: Configuring Thresholds .....	38
Option: Configuring System Logging .....	38
Option: Monitoring Dynamic Flow Capture by Using SNMP .....	39
Hardware and Software Considerations .....	39

### **Chapter 3                      Passive Flow Monitoring Configuration Examples                      41**

Example: Passive Flow Monitoring Configuration .....	41
Verifying Your Work .....	48
Example: Flow Collector Interface Configuration .....	55
Verifying Your Work .....	60
Example: Dynamic Flow Capture Configuration .....	65
Verifying Your Work .....	67
Router 1 .....	67

### **Chapter 4                      Active Flow Monitoring                      69**

Configuring Active Flow Monitoring .....	69
Defining a Firewall Filter to Select Traffic for Active Flow Monitoring .....	72
Configuring the Interfaces That Will Be Actively Monitored .....	73
Enabling the Monitoring Services, Adaptive Services, or Multiservices Interfaces and the Export Interface .....	73
Flow Records Collection .....	74
Collecting Flow Records .....	74
Collecting Flow Records with a Sampling Group .....	75
Collecting Flow Records with an Accounting Group .....	76
Replicating Routing Engine-Based Sampling to Multiple Flow Servers .....	77
Collecting Flow Records with a Template .....	78
Routing Engine-Based Sampling to Multiple Flow Servers .....	79

Replicating Version 9 Flow Aggregation to Multiple Flow Servers .....	80
Option: Configuring an Aggregate Export Timer .....	81
Option: Configuring Port Mirroring .....	81
Option: Configuring Port Mirroring with Filter-Based Forwarding and a Monitoring Group .....	82
Option: Sending Traffic to Multiple Export Interfaces by Using Next-Hop Groups .....	82
Option: Using the Flow-Tap Application to Send Packets to a Mediation Device .....	83
Option: Using the Flow-Tap Application to Send Packets to a Mediation Device .....	83
Flow-Tap Architecture .....	84
Configuring the Flow-Tap Interface .....	85
Configuring Flow-Tap Security Properties .....	86
Flow-Tap Application Restrictions .....	86
Example: Flow-Tap Configuration .....	87

## **Chapter 5      Active Flow Monitoring Configuration Examples      89**

Example: Sampling Configuration .....	89
Verifying Your Work .....	91
Example: Sampling and Discard Accounting Configuration .....	93
Verifying Your Work .....	96
Example: Multiple Port Mirroring with Next-Hop Groups Configuration .....	97
Example: Sampling Instance Configuration .....	100
Example Network Details .....	101
Example Router Configuration .....	102
Configuration Commands Used for the Configuration Example .....	103
Verifying Your Work .....	104

## **Chapter 6      Formats for Flow Monitoring Output      107**

Flow Monitoring Output Formats .....	107
Version 5 Formats and Fields .....	108
Version 8 Formats and Fields .....	111
Version 9 Formats and Fields .....	117
More Information About Passive and Active Flow Monitoring .....	123

## **Part 2      Index**

Index .....	127
-------------	-----



# List of Figures

## Part 1

## Flow Monitoring

<b>Chapter 1</b>	<b>Flow Monitoring Concepts and Reference Material</b>	<b>3</b>
	Figure 1: Passive Flow Monitoring Application Topology .....	4
<b>Chapter 2</b>	<b>Passive Flow Monitoring</b>	<b>13</b>
	Figure 2: Dynamic Flow Capture Topology .....	35
<b>Chapter 3</b>	<b>Passive Flow Monitoring Configuration Examples</b>	<b>41</b>
	Figure 3: Passive Flow Monitoring—Topology Diagram .....	41
	Figure 4: Flow Collector Interface Topology Diagram .....	55
<b>Chapter 4</b>	<b>Active Flow Monitoring</b>	<b>69</b>
	Figure 5: Flow-Tap Topology Diagram .....	85
<b>Chapter 5</b>	<b>Active Flow Monitoring Configuration Examples</b>	<b>89</b>
	Figure 6: Active Flow Monitoring—Sampling Configuration Topology Diagram .....	89
	Figure 7: Active Flow Monitoring—Sampling and Discard Accounting Topology Diagram .....	93
	Figure 8: Active Flow Monitoring—Multiple Port Mirroring with Next-Hop Groups Topology Diagram .....	98
	Figure 9: Active Flow Monitoring—Sampling Instance Configuration Topology Diagram .....	101
<b>Chapter 6</b>	<b>Formats for Flow Monitoring Output</b>	<b>107</b>
	Figure 10: Version 5 Packet Header Format .....	108
	Figure 11: Version 5 Flow-Export Flow Header Format .....	109
	Figure 12: Version 8 Template Flow Format .....	111
	Figure 13: Version 8 AS Aggregation Flow Entry Format .....	112
	Figure 14: Version 8 Protocol/Port Aggregation Flow Entry Format .....	113
	Figure 15: Version 8 Prefix Aggregation Flow Entry Format .....	114
	Figure 16: Version 8 Source Prefix Aggregation Flow Entry Format .....	115
	Figure 17: Version 8 Destination Prefix Aggregation Flow Entry Format .....	116
	Figure 18: Version 9 Flow Header Format .....	119
	Figure 19: Version 9 Template FlowSet Format .....	119
	Figure 20: Version 9 Data FlowSet Format .....	121
	Figure 21: Version 9 Options Template Format .....	122
	Figure 22: Active Flow Monitoring Version 9 Options Data Record Format .....	123



# List of Tables

## Part 1

## Flow Monitoring

<b>Chapter 1</b>	<b>Flow Monitoring Concepts and Reference Material</b>	<b>3</b>
	Table 1: Monitoring Services PIC Specifications .....	8
	Table 2: Monitoring Services II PIC Specifications .....	9
	Table 3: Adaptive Services PIC Specifications .....	9
	Table 4: MultiServices 100 PIC .....	9
	Table 5: MultiServices 400 PIC .....	10
	Table 6: MultiServices 500 PIC .....	10
<b>Chapter 2</b>	<b>Passive Flow Monitoring</b>	<b>13</b>
	Table 7: Passive Flow Monitoring PIC Support .....	13
	Table 8: Name Format Macros .....	29
<b>Chapter 3</b>	<b>Passive Flow Monitoring Configuration Examples</b>	<b>41</b>
	Table 9: Output Fields for the show passive-monitoring error Command .....	49
	Table 10: Output Fields for the show passive-monitoring flow Command .....	50
	Table 11: Output Fields for the show passive-monitoring memory Command .....	52
	Table 12: Output Fields for the show passive-monitoring status Command .....	53
	Table 13: Output Fields for the show passive-monitoring usage Command .....	54
	Table 14: Flow Collector Interface Transfer Log Fields .....	63
	Table 15: Flow Collector Interface File Fields in Order of Appearance .....	64
<b>Chapter 4</b>	<b>Active Flow Monitoring</b>	<b>69</b>
	Table 16: Passive and Active Flow Monitoring PIC Support .....	69
<b>Chapter 6</b>	<b>Formats for Flow Monitoring Output</b>	<b>107</b>
	Table 17: Export Version 5 Packet Header Fields .....	108
	Table 18: Export Version 5 Flow-Export Flow Header Fields .....	109
	Table 19: Version 8 Flow Template Fields .....	112
	Table 20: Version 8 AS Aggregation Flow Entry Fields .....	112
	Table 21: Version 8 Protocol/Port Aggregation Flow Entry Fields .....	113
	Table 22: Version 8 Prefix Aggregation Flow Entry Fields .....	114
	Table 23: Version 8 Source Prefix Aggregation Flow Entry Fields .....	115
	Table 24: Version 8 Destination Prefix Aggregation Flow Entry Fields .....	116
	Table 25: Flow Monitoring Version 9 Template Formats .....	117
	Table 26: Version 9 Flow Header Fields .....	119
	Table 27: Version 9 Template FlowSet Fields .....	120
	Table 28: Field Type Definitions Supported in the JUNOS Software .....	120
	Table 29: Version 9 Data FlowSet Format .....	122
	Table 30: Version 9 Options Template Format .....	122

Table 31: Active Flow Monitoring Version 9 Options Data Record  
Format .....123

## **Part 1**

# **Flow Monitoring**

- Flow Monitoring Concepts and Reference Material on page 3
- Passive Flow Monitoring on page 13
- Passive Flow Monitoring Configuration Examples on page 41
- Active Flow Monitoring on page 69
- Active Flow Monitoring Configuration Examples on page 89
- Formats for Flow Monitoring Output on page 107



## Chapter 1

# Flow Monitoring Concepts and Reference Material

This section contains the following topics:

- Overview of Flow Monitoring on page 3
- Passive Flow Monitoring on page 4
- Active Flow Monitoring on page 5
- System Requirements on page 6
- Passive Flow Monitoring System Requirements on page 6
- Active Flow Monitoring System Requirements on page 7
- Active Flow Monitoring PIC Specifications on page 8
- Terms and Acronyms on page 11

## Overview of Flow Monitoring

---

The flow monitoring application performs traffic flow monitoring and enables lawful interception of packets transiting between two routers. Traffic flows can either be passively monitored by an offline router or actively monitored by a router participating in the network.

Using a Juniper Networks router, a selection of Physical Interface Cards (PICs) for M Series and T Series routers—including the Monitoring Services PIC, Monitoring Services II PIC, Adaptive Services PIC, and MultiServices PICs—and other networking hardware, you can monitor traffic flow and export the monitored traffic. Monitoring traffic allows you to do the following:

- Gather and export detailed information about traffic flows between source and destination routers in your network.
- Sample all incoming traffic on the monitoring interface and present the data in record format.
- Encrypt or tunnel outgoing records, intercepted traffic, or both.
- Direct filtered traffic to different packet analyzers and present the data in its original format.
- Intercept unwanted traffic, discard it, and perform accounting on the discarded packets.

There are two main types of flow monitoring:

- Active Flow Monitoring
- Passive Flow Monitoring

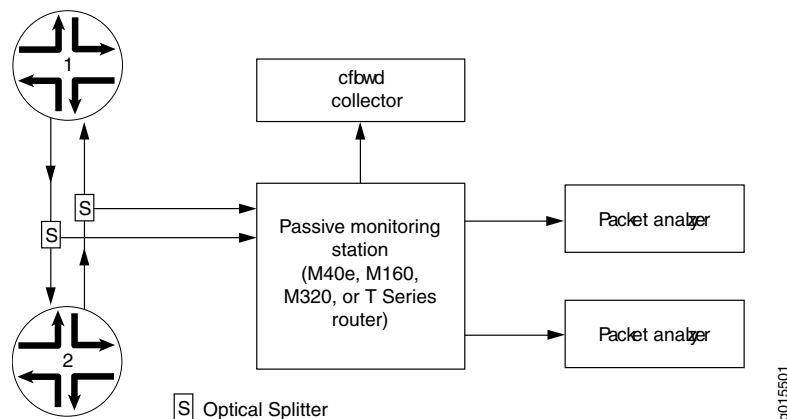
- Related Topics**
- Active Flow Monitoring on page 5
  - Passive Flow Monitoring on page 4

## Passive Flow Monitoring

Flow monitoring version 5 supports passive flow monitoring. Versions 8 and 9 do not support passive flow monitoring.

The M40e, M160, M320, MX Series, or T Series router that is used for passive flow monitoring does not route packets from monitored interfaces, nor does it run any routing protocols related to those interfaces; it only passes along intercepted traffic and receives traffic flows. Figure 1 on page 4 shows a typical topology for the passive flow monitoring application.

**Figure 1: Passive Flow Monitoring Application Topology**



Traffic travels normally between Router 1 and Router 2. To redirect IPv4 traffic, you insert an optical splitter on the interface between these two routers. The optical splitter copies and redirects the traffic to the monitoring station. The optical cable connects only the receive port on the monitoring station, never the transmit port. This configuration allows the monitoring station to receive traffic only from the router being monitored but never to transmit it back.

If you are monitoring traffic flow, the Internet Processor II ASIC in the router forwards a copy of the traffic to the Monitoring Services or Monitoring Services II PIC in the monitoring station. If there is more than one Monitoring Services PIC installed, the monitoring station distributes the load of the incoming traffic across the multiple PICs. The Monitoring Services PICs generate flow records in version 5 format, and the records are exported to the flow collector.

When you are performing lawful interception of packets, the Internet Processor II ASIC filters the incoming traffic and forwards it to the Tunnel Services PIC. Filter-based forwarding is then applied to direct the traffic to the packet analyzers. Optionally, the intercepted traffic or the flow records can be encrypted by the ES PIC and then sent to their destination. With additional configuration, flow records can be processed by a flow collector and flows can be captured dynamically.

With MPLS passive monitoring, the router can process MPLS packets with label values that do not have corresponding entries in the `mpls.0` routing table. You can divert these unrecognized MPLS packets, remove the MPLS labels, and redirect the underlying IPv4 packets. This is equivalent to a default route for MPLS packets or a promiscuous label. Because this application does not use a Monitoring Services PIC, see the *JUNOS MPLS Applications Configuration Guide* for more information about MPLS passive monitoring.

- Related Topics**
- Overview of Flow Monitoring on page 3
  - Active Flow Monitoring on page 5

## Active Flow Monitoring

---

Flow monitoring versions 5, 8, and 9 support active flow monitoring. For active flow monitoring, the monitoring station participates in the network as an active router. The major actions the router can perform during active flow monitoring are as follows:

- Sampling—The router selects and analyzes only a portion of the traffic.
- Sampling with templates—The router selects, analyzes, and arranges a portion of the traffic into templates.
- Sampling per sampling instance—The router selects, analyzes, and arranges a portion of the traffic according to the configuration and binding of a sampling instance.
- Port mirroring—The router copies entire packets and sends the copies to another interface.
- Multiple port mirroring—The router sends multiple copies of monitored packets to multiple export interfaces with the `next-hop-group` statement at the `[edit forwarding-options]` hierarchy level.
- Discard accounting—The router accounts for selected traffic before discarding it. Such traffic is not forwarded out of the router. Instead, the traffic is quarantined and deleted.
- Flow-tap processing—The router processes requests for active flow monitoring dynamically by using the Dynamic Tasking Control Protocol (DTCP).

- Related Topics**
- Overview of Flow Monitoring on page 3
  - Passive Flow Monitoring on page 4

## System Requirements

---

This section describes the system requirements for flow monitoring and contains the following sections:

- Passive Flow Monitoring System Requirements on page 6
- Active Flow Monitoring System Requirements on page 7
- Active Flow Monitoring PIC Specifications on page 8

## Passive Flow Monitoring System Requirements

---

To perform passive flow monitoring, your system must meet these minimum requirements:

- JUNOS Release 8.5 or later for passive flow monitoring support on the MX Series MultiServices routers
- JUNOS Release 8.4 or later for passive flow monitoring support on the MultiServices 400 PIC (Type 2)
- JUNOS Release 7.6 or later to clear error and flow statistics with the **clear passive-monitoring statistics** command
- JUNOS Release 7.5 or later for support of the dynamic flow capture (DFC) Management Information Base (MIB)
- JUNOS Release 7.4 or later for dynamic flow capture on Monitoring Services III PICs installed in T Series and M320 routers, and port mirroring of IPv6 packets
- JUNOS Release 7.3 or later for passive flow monitoring on selected Ethernet-based interfaces and filter-based forwarding on output interfaces
- JUNOS Release 7.1 or later for passive flow monitoring and flow collection services on Monitoring Services II PICs installed in T Series and M320 routers
- JUNOS Release 6.4 or later for support of the next-hop IP address field in flow monitoring version 5 records
- JUNOS Release 6.2 or later for ATM2 intelligent queuing (IQ) interface passive monitoring, flow collection services, and MPLS label stripping
- JUNOS Release 6.1 or later for MPLS passive monitoring
- JUNOS Release 6.0 or later for the Monitoring Services II PIC
- JUNOS Release 5.7 or later for the automatic insertion of autonomous system (AS) numbers and SNMP index values for interfaces into flow records
- JUNOS Release 5.4 or later for the Monitoring Services PIC
- M40e, M1 60, M320, MX Series, or T Series router with an Internet Processor II ASIC or later
- Type 1 enhanced FPCs
- Two optical splitters

- A Tunnel Services PIC (required if you wish to send traffic to more than one analyzer)
- An input interface from the following list:
  - SONET/SDH PIC—OC3, OC12, or OC48
  - ATM2 IQ PIC—OC3 or OC12
  - 4-port Fast Ethernet PIC
  - Gigabit Ethernet PIC—4-port with small form-factor pluggable transceiver (SFP) or 10-port with SFP
  - 1-port 10-Gigabit Ethernet PIC with XENPAK
- Outgoing PICs to connect to the flow collector or packet analyzer
- Flow monitoring version 5 collector
- ES PIC and packet analyzers (optional)

**Related Topics**

- Active Flow Monitoring System Requirements on page 7
- Active Flow Monitoring PIC Specifications on page 8

## Active Flow Monitoring System Requirements

---

To implement active flow monitoring, your system must meet these minimum requirements:

- JUNOS 9.3R2 or later for IPv6 support on flow monitoring version 9
- JUNOS 9.3R2 or later for multiple flows for flow monitoring version 9
- JUNOS Release 9.0 or later for version 9 flow aggregation to multiple flow servers
- JUNOS Release 8.5 or later for active flow monitoring support on MultiServices 500 PICs
- JUNOS Release 8.3 or later for flow monitoring version 9 support, MPLS support, and active flow monitoring support on MultiServices 100 and 400 PICs
- JUNOS Release 8.2 or later for M120 router support and for flow monitoring version 5 and 8 support on MultiServices 100 and 400 PICs
- JUNOS Release 8.1 or later for the flow-tap services application on Adaptive Services II PICs installed in M7i, M10i, M20, M40e, M320, and T Series routers
- JUNOS Release 7.4 or later for port mirroring of IPv6 packets
- JUNOS Release 7.3 or later for active flow monitoring on Adaptive Services II PICs installed in TX Matrix platforms
- JUNOS Release 7.0 or later for active flow monitoring on Adaptive Services II PICs installed in T Series and M320 routers
- JUNOS Release 7.0 or later for active flow monitoring on J Series Services Routers
- JUNOS Release 6.0 or later for the Adaptive Services PIC

- JUNOS Release 5.7 or later for the automatic insertion of AS numbers and SNMP index values for input and output interfaces into records, port mirroring to multiple ports, and discard accounting
- JUNOS Release 5.6 or later for the Monitoring Services PIC
- M5, M7i, M10, M10i, M20, M40e, M120, M160, M320, or T Series router with an Internet Processor II ASIC or later; or a J Series Services Router
- Type 1 enhanced FPCs
- Two M Series or T Series PICs or J Series Physical Interface Modules (PIMs) of your choice: One to receive incoming traffic and one to forward outgoing traffic (the second PIC or PIM is not necessary for discard accounting)
- Export PICs to connect to the collector or packet analyzer
- Tunnel Services PIC (required for multiple port mirroring or mo- interface load balancing)
- Flow collector version 5, 8, or 9
- ES PIC and packet analyzers (optional)

**Related Topics**

- Passive Flow Monitoring System Requirements on page 6
- Active Flow Monitoring PIC Specifications on page 8

## Active Flow Monitoring PIC Specifications

---

For Monitoring Services PIC specifications, see Table 1 on page 8 and Table 2 on page 9. For Adaptive Services PIC specifications, see Table 3 on page 9. For MultiServices PIC specifications, see Table 4 on page 9 and Table 5 on page 10.

**Table 1: Monitoring Services PIC Specifications**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	DB-9 diagnostic serial console port
Status LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The service is not running.</li> <li>■ Green—The service is running under acceptable load.</li> <li>■ Amber—The service is overloaded.</li> </ul>

**Table 2: Monitoring Services II PIC Specifications**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	N/A
Status LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The flow collector is not running.</li> <li>■ Green—The flow collector is running under acceptable load.</li> <li>■ Amber—The flow collector is overloaded.</li> </ul>

**Table 3: Adaptive Services PIC Specifications**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	N/A
Status LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	One tricolor: <ul style="list-style-type: none"> <li>■ Off—The flow collector is not running.</li> <li>■ Green—The flow collector is running under acceptable load.</li> <li>■ Amber—The flow collector is overloaded.</li> </ul>

**Table 4: MultiServices 100 PIC**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	N/A

**Table 4: MultiServices 100 PIC (continued)**

Specification	Description
Status LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The service is not running.</li> <li>■ Green—The service is running under acceptable load.</li> <li>■ Amber—The service is overloaded.</li> </ul>

**Table 5: MultiServices 400 PIC**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	N/A
Status LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The service is not running.</li> <li>■ Green—The service is running under acceptable load.</li> <li>■ Amber—The service is overloaded.</li> </ul>

**Table 6: MultiServices 500 PIC**

Specification	Description
Physical dimensions	Single-wide PIC that occupies one PIC slot
Connectors	N/A

**Table 6: MultiServices 500 PIC (continued)**

Specification	Description
Status LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The PIC is offline; it is safe to remove it from the chassis.</li> <li>■ Green—The PIC is operating normally.</li> <li>■ Amber—The PIC is initializing.</li> <li>■ Red—The PIC has an error or failure; no further harm can be done by removing it from the chassis.</li> </ul>
Application LED	<p>One tricolor:</p> <ul style="list-style-type: none"> <li>■ Off—The service is not running.</li> <li>■ Green—The service is running under acceptable load.</li> <li>■ Amber—The service is overloaded.</li> </ul>

- Related Topics**
- Passive Flow Monitoring System Requirements on page 6
  - Active Flow Monitoring System Requirements on page 7

## Terms and Acronyms

### A

- active flow monitoring** Technique to lawfully intercept and observe specified data network traffic on an active router participating in the network.
- Adaptive Services PIC** Advanced PIC that handles active flow monitoring, Network Address Translation (NAT), stateful firewall, and intrusion detection functions. For more information on the Adaptive Services PIC, see the *JUNOS Services Interfaces Configuration Guide*.

### C

- cflowd** Version 5 and version 8 flow monitoring process that captures flow information from network traffic and exports this data into summary tables. Once captured, flow data can be analyzed as needed. For more information about cflowd, see <http://www.caida.org>.
- content destination** A recipient of monitored packets sent by a DTCP or dynamic flow capture-enabled monitoring station.
- control source** A dynamic flow capture client that wants to monitor electronic data or voice transfer over the network. The control source sends filter requests to the dynamic flow capture-enabled monitoring station by using DTCP.

### D

**DTCP (Dynamic Tasking Control Protocol)**

Protocol used to specify filtering criteria in a dynamic flow capture environment.

**dynamic flow capture**

Technique that allows DTCP-enabled control sources to send specified filtering criteria in real time to a monitoring station. The monitoring station passively monitors the specified traffic flows on demand and sends the captured packets to content destinations.

**E**

---

**ES PIC**

PIC that handles encryption and security services (such as IP Security [IPSec]).

**F**

---

**flow collector interface**

Converted Monitoring Services II PIC that processes multiple flow records into compressed ASCII data files and exports these files to an FTP server.

**M**

---

**Monitoring Services II PIC**

Advanced PIC that handles passive flow monitoring functions.

**Monitoring Services III PIC**

Advanced PIC that handles dynamic flow capture functions.

**Monitoring Services PIC**

Original PIC that handles passive and active flow monitoring functions.

**MultiServices 100 PIC**

Also referred to as MultiServices PIC Type 1. Advanced PIC that handles active flow capture functions.

**MultiServices 400 PIC**

Also referred to as MultiServices PIC Type 2. Advanced PIC that handles active flow capture functions.

**MultiServices 500 PIC**

Also referred to as MultiServices PIC Type 3. Advanced PIC that handles active flow capture functions.

**P**

---

**passive flow monitoring**

Technique to lawfully intercept and observe specified data network traffic on a passive flow monitoring station not participating in the network.

## Chapter 2

# Passive Flow Monitoring

This section contains the following topics:

- Configuring Passive Flow Monitoring on page 13
- Monitoring Traffic with a VRF Instance and a Monitoring Group on page 14
- Using Port Mirroring and Filter-Based Forwarding to Copy and Redirect Traffic on page 22
- Using a Flow Collector Interface to Process and Export Multiple Flow Records on page 28
- Monitoring Traffic Using a Dynamic Flow Capture Interface on page 33
- Hardware and Software Considerations on page 39

## Configuring Passive Flow Monitoring

Table 7 on page 13 shows which Juniper Networks PICs and routers support passive flow monitoring. The PICs receive passively monitored network traffic from an input interface (SONET/SDH, ATM2 IQ, Fast Ethernet, Gigabit Ethernet, or 10-Gigabit Ethernet), convert the received packets into flow records, and export them to a flow server for further analysis.

**Table 7: Passive Flow Monitoring PIC Support**

PIC Type	M40e	M160	T Series/ M320
Monitoring Services PIC	Yes	Yes	No
Monitoring Services II PIC	Yes	Yes	Yes
Monitoring Services III PIC	Yes	Yes	Yes
MultiServices 400 PIC (Type 2)	Yes	No	Yes

The key configuration hierarchy statement for passive flow monitoring is the **monitoring** statement found at the [edit forwarding-options] hierarchy level. At minimum, you must configure a VRF routing instance to direct the traffic to a monitoring services interface for flow processing.

However, there are several options you can use that add complexity to passive flow monitoring. For example, you can configure the router to direct traffic into a routing instance and deliver the traffic into a monitoring group. You can also use port mirroring and filter-based forwarding to copy and redirect traffic. Optionally, you can configure the monitoring station to encrypt flow output before it is sent to a flow server for processing, to send flow records to a flow collector, or to process on-demand monitoring requests with dynamic flow capture.

- Related Topics**
- Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding on page 22
  - Using a Flow Collector Interface to Process and Export Multiple Flow Records on page 28
  - Using a Dynamic Flow Capture Interface to Monitor Traffic On Demand on page 34
  - Hardware and Software Considerations on page 39

## Monitoring Traffic with a VRF Instance and a Monitoring Group

---

The first way you can implement passive flow monitoring is to direct traffic into a VRF routing instance and use a monitoring group to export this traffic to a flow server for analysis. Complete the following tasks:

- Specifying a Firewall Filter to Select Traffic to Monitor on page 14
- Configuring Input Interfaces, Monitoring Services Interfaces, and Export Interfaces on page 15
- Establishing a VRF Instance for the Monitored Traffic on page 18
- Configuring a Monitoring Group to Send Traffic to the Flow Server on page 19
- Configuring Policy Options on page 20
- Option: Stripping MPLS Labels on ATM, Ethernet-Based, and SONET/SDH Interfaces on page 21

### ***Specifying a Firewall Filter to Select Traffic to Monitor***

When you define a firewall filter, you select the initial traffic to be monitored. To configure a firewall filter, include the `filter` statement at the `[edit firewall family inet]` hierarchy level. All filtered traffic to be monitored must be accepted.

```
[edit]
firewall {
  family inet {
    filter input-monitoring-filter {
      term 1 {
        from {
          destination-address {
            10.7.0.0/16;
          }
        }
        then {
          count counter1;
        }
      }
    }
  }
}
```

```

        accept;
    }
}
term 2 {
    from {
        destination-address {
            10.6.0.0/16;
        }
    }
    then {
        count counter2;
        accept;
    }
}
}
}
}

```

### **Configuring Input Interfaces, Monitoring Services Interfaces, and Export Interfaces**

After creating the input filter, you need to configure the interfaces where traffic will enter the router. To enable passive flow monitoring for SONET/SDH input interfaces, include the **passive-monitor-mode** statement at the **[edit interfaces so-fpc/pic/port unit unit-number]** hierarchy level. This mode disables the router from participating in the network as an active device. On SONET/SDH interfaces, passive monitor mode suppresses SONET keepalives.

For ATM2 IQ interfaces, passive monitor mode suppresses the sending and receiving of ATM Operations, Administration, and Maintenance (OAM) and Integrated Local Management Interface (ILMI) control messages. To enable passive flow monitoring for ATM2 IQ input interfaces, include the **passive-monitor-mode** statement at the **[edit interfaces at-fpc/pic/port]** hierarchy level. ATM passive monitoring supports the following interface encapsulation types: Cisco-compatible ATM Network Layer Protocol ID (NLPID) (**atm-cisco-nlpid**), ATM NLPID (**atm-nlpid**), ATM Point-to-Point Protocol (PPP) over ATM Adaptation Layer 5 (AAL5)/ logical link control (LLC) (**atm-ppp-llc**), ATM PPP over raw AAL5 (**atm-ppp-vc-mux**), ATM LLC/ subnetwork attachment point (SNAP) (**atm-snap**), and ATM virtual circuit (VC) multiplexing (**atm-vc-mux**).

Ethernet-based interfaces support both per-port passive monitoring and per-VLAN passive monitoring. For Fast Ethernet interfaces, include the **passive-monitor-mode** statement at the **[edit interfaces fe-fpc/pic/port]** hierarchy level. For Gigabit Ethernet interfaces, include the **passive-monitor-mode** statement at the **[edit interfaces ge-fpc/pic/port]** hierarchy level. On Ethernet-based interfaces, passive monitor mode disables the Routing Engine from receiving packets and prevents the routing table from transmitting packets. You can verify this by the presence of the **No-receive** and **No-transmit** interface flags in the output of the **show interfaces (fe | ge)-fpc/pic/port** command.



**NOTE:** The following restrictions apply to passive flow monitoring on Ethernet-based interfaces:

- No special encapsulation types are allowed, so you must configure Ethernet encapsulations only.
- When you configure the `passive-monitor-mode` statement, destination MAC address filters applied to incoming interfaces are disabled by default.
- The `flow-control` statement at the [edit interfaces *ge-fpc/pic/port* *gigether-options*] or [edit interfaces *fe-fpc/pic/port* *fastether-options*] hierarchy level does not work when passive flow monitoring is enabled.

In addition to passive monitor mode, apply the previously defined firewall filter to the interface with the `filter` statement at the [edit interfaces *interface-name-fpc/pic/port* *unit unit-number* *family inet*] hierarchy level:

```
[edit]
interfaces {
  so-0/0/0 {
    description "SONET/SDH input interface";
    encapsulation ppp;
    unit 0 {
      passive-monitor-mode;
      family inet {
        filter {
          input input-monitoring-filter;
        }
      }
    }
  }
}
at-1/0/0 {
  description "ATM2 IQ input interface";
  passive-monitor-mode;
  atm-options {
    pic-type atm2;
    vpi 0 {
      maximum-vcs 255;
    }
  }
  unit 0 {
    encapsulation atm-snap;
    vci 0.100;
    family inet {
      filter {
        input input-monitoring-filter;
      }
    }
  }
}
ge-2/0/0 {
  description "Gigabit Ethernet input interface";
  passive-monitor-mode;
  unit 0 {
```

```

        family inet {
            filter {
                input input-monitoring-filter;
            }
        }
    }
}

```

Configure the interfaces on the Monitoring Services PIC or Monitoring Services II PIC with the **family inet** statement at the **[edit interfaces mo-fpc/pic/port unit unit-number]** hierarchy level. The statement allows the interfaces to process IPv4 traffic received from the input interfaces.

When you use VRF instances, you need to configure two logical interfaces. The first (unit 0) is part of the inet.0 routing table and sources the flow packets. The second (unit 1) is configured as part of the VRF instance so the monitoring services interface can serve as a valid next hop for packets received in the instance.

You can also capture options packets and time-to-live (TTL) exceeded information when the monitoring services interface processes flow records. To configure, include the **receive-options-packets** and **receive-ttl-exceeded** statements at the **[edit interfaces mo-fpc/pic/port unit unit-number family inet]** hierarchy level:

```

[edit]
interfaces {
    mo-4/0/0 {
        unit 0 {
            family inet {
                receive-options-packets;
                receive-ttl-exceeded;
            }
        }
        unit 1 {
            family inet;
        }
    }
    mo-4/1/0 {
        unit 0 {
            family inet;
        }
        unit 1 {
            family inet;
        }
    }
    mo-4/2/0 {
        unit 0 {
            family inet;
        }
        unit 1 {
            family inet;
        }
    }
    mo-4/3/0 {
        unit 0 {

```

```

        family inet;
    }
    unit 1 {
        family inet;
    }
}

```

You must also configure the export interface where flow packets exit the monitoring station and are sent to the flow server.

On output interfaces, you can apply a firewall filter that leads to a filter-based forwarding routing instance. This is useful if you want to port-mirror traffic to multiple Monitoring Services PICs or flow collection services interfaces. To configure, include the `output` statement at the [edit interfaces *interface-name* unit *logical-unit-number* family *inet* filter] hierarchy level. For more information, see “Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations” on page 25.

```

[edit]
interfaces
fe-3/0/0 {
    description "export interface to flow server";
    unit 0 {
        family inet;
        address ip-address;
        filter {
            output output-filter-name;
        }
    }
}

```

## Establishing a VRF Instance for the Monitored Traffic

After the firewall filter and interfaces are ready, create a VPN routing and forwarding (VRF) instance. The filtered traffic enters the VRF instance and is shared only between the input interfaces and the monitoring services output interfaces. In this case, a group of four monitoring services interfaces is used as the next hop.

```

[edit]
routing-instances {
    monitoring-vrf {
        instance-type vrf;
        interface so-0/0/0.0;
        interface so-0/1/0.0;
        interface mo-4/0/0.1;
        interface mo-4/1/0.1;
        interface mo-4/2/0.1;
        route-distinguisher 69:1;
        vrf-import monitoring-vrf-import;
        vrf-export monitoring-vrf-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop [mo-4/0/0.1 mo-4/1/0.1 mo-4/2/0.1];
            }
        }
    }
}

```

```
}
}
```

## Configuring a Monitoring Group to Send Traffic to the Flow Server

You collect flow records by specifying output interfaces in a monitoring group. In general, the monitoring services interfaces are the output interfaces. The logical unit number on the output interfaces when used in conjunction with a VRF instance must be 1. To configure, include the `output` statement at the [edit forwarding-options monitoring *group-name* family inet] hierarchy level.



**NOTE:** Because routing instances determine the input interface, the `input` statement at the [edit forwarding-options monitoring *group-name* family inet] hierarchy level has been removed in JUNOS Release 6.0 and later. If you have a configuration that contains this old statement, we recommend that you update your configuration and remove the statement.

As part of the `mo-fpc/pic/port` statement at the [edit forwarding-options monitoring *group-name* family inet output interface] hierarchy level, you must specify a source address for transmission of flow information. You can use the router ID IP address, the IP address of the input interface, or any local IP address of your choice as the source address. If you provide a different `source-address` statement for each monitoring services output interface, you can track which interface processes a particular flow record.

All other statements at this level (`engine-id`, `engine-type`, `input-interface-index`, and `output-interface-index`) are dynamically generated, but can be configured manually. To reset outgoing interface or incoming interface indexes that were once configured manually, configure the `input-interface-index` or `output-interface-index` statements with a value of 0 at the [edit forwarding-options monitoring *group-name* family inet output interface *interface-name*] hierarchy level.

To specify the flow server IP address and port number, include the `flow-server ip-address port port-number` statement at the [edit forwarding-options monitoring *group-name* family inet output] hierarchy level. You can specify up to eight flow servers in a monitoring group and the IP address for each server must be unique. Flow records are exported and load-balanced between all active flow servers.

Once you configure the VRF and monitoring group statements, traffic enters the input interfaces, passes to the monitoring services interfaces for processing, and is discarded. The resulting flow description packets exit the monitoring station through the export interface. If you want traffic to travel to destinations other than the monitoring services interfaces, or need to establish additional analysis, see the section “Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding” on page 22.



**NOTE:** You must complete interface configuration on the Monitoring Services or Monitoring Services II PIC before an interface can be added into a monitoring group. For more information, see “Configuring Input Interfaces, Monitoring Services Interfaces, and Export Interfaces” on page 15.

```
[edit]
forwarding-options {
  monitoring group1 {
    family inet {
      output {
        export-format cflowd-version-5;
        flow-active-timeout 60;
        flow-inactive-timeout 30;
        flow-server 192.168.245.1 port 2055;
        flow-server 192.168.245.2 port 2055;
        interface mo-4/0/0.1 {
          engine-id 1;
          engine-type 1;
          input-interface-index 44;
          output-interface-index 54;
          source-address 192.168.245.1;
        }
        interface mo-4/1/0.1 {
          engine-id 2;
          engine-type 1;
          input-interface-index 45;
          output-interface-index 55;
          source-address 192.168.245.1;
        }
        interface mo-4/2/0.1 {
          engine-id 3;
          engine-type 1;
          input-interface-index 46;
          output-interface-index 56;
          source-address 192.168.245.1;
        }
      }
    }
  }
}
```

## Configuring Policy Options

When you use a group of next hops in your monitoring group, you can load-balance traffic and distribute it to the export interfaces if you configure policy options. To configure, include the **load-balance per-packet** statement at the `[edit policy-options policy-statement policy-name then]` hierarchy level. You can also reject import and export of VRF routes by including the **reject** statement at the `[edit policy-options policy-statement policy-name then]` hierarchy level.

```
[edit]
routing-options {
  forwarding-table {
```

```

        export pplb;
    }
}
policy-options {
    policy-statement monitoring-vrf-import {
        then {
            reject;
        }
    }
    policy-statement monitoring-vrf-export {
        then {
            reject;
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
}

```

### **Option: Stripping MPLS Labels on ATM, Ethernet-Based, and SONET/SDH Interfaces**

Because flow monitoring can be performed only on IPv4 packets, any packets containing MPLS labels must have the labels removed before monitoring can occur. To remove MPLS labels from packets as they enter an ATM2 IQ, Ethernet-based, or SONET/SDH interface, include the **pop-all-labels** statement at the [edit interfaces *interface-name-fpc/pic/port* (atm | fastether | gigether | sonet)-options mpls] hierarchy level. If you use static MPLS labels, we recommend you assign label values from 10000 through 99999 to avoid using the label ranges reserved by the JUNOS Software.

To remove a specified number of labels from selected packets with MPLS labels, include the **required-depth** statement at the [edit interfaces *interface-name-fpc/pic/port* (atm | fastether | gigether | sonet)-options mpls pop-all-labels] hierarchy level. A **required-depth** value of 1 removes labels from all packets containing only one MPLS label, a value of 2 removes labels from all packets containing only two MPLS labels, and a value of [1 2] removes labels from all packets containing either one or two MPLS labels. The **required-depth** value of [1 2] is the default setting. When you configure the **required-depth** statement, you must configure the same value for all ports on the same PIC.

The labels are removed and discarded as soon as they arrive at the interface. As a result, no MPLS filters can be applied to the stripped labels, no statistics are generated for the labels, and you cannot apply an IP filter to the incoming packets. No Tunnel Services PIC is required to perform MPLS label stripping.

```

[edit]
interfaces {
    at-/fpc/pic/port {
        atm-options {
            mpls {
                pop-all-labels {
                    required-depth 1;
                }
            }
        }
    }
}

```

```

    }
  }
}
(fe | ge)-fpc/pic/port {
  (fastether | gigether)-options {
    mpls {
      pop-all-labels {
        required-depth [1 2];
      }
    }
  }
}
so-fpc/pic/port {
  sonet-options {
    mpls {
      pop-all-labels {
        required-depth 2;
      }
    }
  }
}
}

```

## Using Port Mirroring and Filter-Based Forwarding to Copy and Redirect Traffic

This section discusses additional techniques you can use with the passive flow monitoring application.

- Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding on page 22
- Specifying Port Mirroring Input and Output on page 23
- Creating a Firewall Filter to Split the Port-Mirrored Traffic into Different Instances on page 24
- Applying the Firewall Filter to a Tunnel PIC Interface on page 25
- Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations on page 25
- Configuring a Routing Table Group to Add Interface Routes into the Forwarding Instance on page 26
- Option: Using an ES PIC to Send Traffic to a Packet Analyzer on page 26
- Option: Applying a Firewall Filter to an Output Interface on page 28

### Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding

This section discusses additional techniques you can use with the passive flow monitoring application:

- In addition to flow analysis, you can analyze a copy of the original traffic with a single packet analyzer. To implement this technique, divert traffic with a filter-based forwarding routing instance and send the monitored traffic through a physical interface to the packet analyzer.

- You can cluster the traffic into different groups and redirect this traffic to multiple packet analyzers. For example, you can break traffic flows into TCP groups and UDP groups and send these groups of packets to different analyzers. To accomplish this, you use port mirroring and send a copy of the original traffic to a Tunnel PIC. Then you can apply a firewall filter, split the traffic into your desired groups, and send these groups toward different exit interfaces leading to the packet analyzers. This technique provides maximum flexibility for traffic analysis.
- For secure transmission of the copied or grouped traffic, you can encrypt the diverted traffic with an ES PIC and send this traffic to a packet analyzer over an IP Security (IPSec) tunnel.

To implement the filter-based forwarding enhancement methods, see the following sections:

- Specifying Port Mirroring Input and Output on page 23
- Creating a Firewall Filter to Split the Port-Mirrored Traffic into Different Instances on page 24
- Applying the Firewall Filter to a Tunnel PIC Interface on page 25
- Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations on page 25
- Configuring a Routing Table Group to Add Interface Routes into the Forwarding Instance on page 26
- Option: Using an ES PIC to Send Traffic to a Packet Analyzer on page 26
- Option: Applying a Firewall Filter to an Output Interface on page 28

## Specifying Port Mirroring Input and Output

This step works in conjunction with the action specified by the `port-mirror` statement configured at the `[edit firewall family (inet | inet6) filter filter-name term term-name then]` hierarchy level. At this point, you select input and output statements to determine where the copies of the IPv4 or IPv6 packets are sent. To configure, include the `input` and `output` statements at the `[edit forwarding-options port-mirroring family family-name]` hierarchy level. The traffic to be monitored is copied, port-mirrored, and sent to the packet analyzer for analysis. On M Series routers, you can port-mirror either IPv4 or IPv6 packets at one time. On M120, M320, and T Series routers, you can port-mirror both IPv4 and IPv6 packets simultaneously.

The port-mirrored copy of the traffic can travel only to a single next hop. As a result, only one type of analysis can be performed if the packets are sent to a packet analyzer through a physical next hop. If more than one type of analysis is desired, a tunnel interface must be used as the next hop for port mirroring. When the mirrored copy of the traffic arrives at the virtual tunnel interface, it can be filtered, split into groups, and redirected to multiple exit interfaces and packet analyzers.

For your input requirements, include the `rate` and `run-length` statements at the `[edit forwarding-options port-mirroring family family-name input]` hierarchy level. For your output requirements, specify the target interface with the `interface` statement at the `[edit forwarding-options port-mirroring family family-name output]` hierarchy level.

By default, a filter cannot be applied to an interface where port-mirrored traffic is received. To allow the tunnel services interface to be used as a filtered next hop, include the `no-filter-check` statement at the `[edit forwarding-options port-mirroring family-name output]` hierarchy level.

```
[edit]
forwarding-options {
  port-mirroring {
    family (inet | inet6) {
      input {
        rate 1;
        run-length 5;
      }
      output {
        interface vt-0/2/0.0;
        no-filter-check;
      }
    }
  }
}
```



**NOTE:** Before JUNOS Release 7.4, you could configure the `input` and `output` statements at the `[edit forwarding-options port-mirroring]` hierarchy level. However, this older syntax has been revised to extend port-mirroring support to IPv6 packets. If you have a configuration that contains the older syntax, we recommend that you update your configuration to the new syntax listed above.

---

## Creating a Firewall Filter to Split the Port-Mirrored Traffic into Different Instances

If you need to split the copy of the monitored traffic into separate groups and send these filtered packets to different analyzers, devise a firewall filter that selects some traffic for sampling and some traffic for discarding. In this case, UDP traffic is sent into one routing instance, TCP traffic is diverted into a second routing instance, and all other traffic is discarded. In a later step, you will define the filter-based forwarding routing instances specified in the `then` statements shown in this filter.

```
[edit]
firewall {
  family inet {
    filter tunnel-interface-filter {
      term tcp {
        from {
          protocol tcp;
        }
        then {
          count tcp;
          routing-instance tcp-routing-table;
        }
      }
      term udp {
        from {
```

```

        protocol udp;
    }
    then {
        count udp;
        routing-instance udp-routing-table;
    }
}
term rest {
    then {
        count rest;
        discard;
    }
}
}
}
}

```

### ***Applying the Firewall Filter to a Tunnel PIC Interface***

Once the firewall filter is defined, apply it as an input filter on a tunnel interface. This is required if the firewall filter defines two or more types of traffic or export interfaces. However, if the firewall filter only specifies one type of traffic and one export interface, you can apply the filter directly to the export interface.

```

[edit]
interfaces {
  vt-0/2/0 {
    unit 0 {
      family inet {
        filter {
          input tunnel-interface-filter;
        }
      }
    }
  }
}
}

```

### ***Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations***

The firewall filter called `tunnel-interface-filter` that you made earlier sends UDP traffic into one filter-based forwarding routing instance called `udp-routing-table`, sends TCP traffic into a second filter-based forwarding routing instance called `tcp-routing-table`, and discards all other packets. Here you will configure the filter-based forwarding instances.

Configure an export interface for each of your routing instances by including a static next hop. To configure, include the `route` statement at the `[edit routing-instances instance-name routing-options static]` hierarchy level and specify a next-hop address or interface.

```

[edit]
routing-instances {
  tcp-routing-table {

```

```

instance-type forwarding;
routing-options {
  static {
    route 0.0.0.0/0 next-hop es-3/1/0.0;
  }
}
}
udp-routing-table {
  instance-type forwarding;
  routing-options {
    static {
      route 0.0.0.0/0 next-hop 10.9.1.2;
    }
  }
}
}
}

```

### **Configuring a Routing Table Group to Add Interface Routes into the Forwarding Instance**

Next, import the interface routes into the forwarding instance. This step is necessary because the next hops specified in the forwarding instances must be installed in the forwarding instances themselves. To configure, include the `import-rib` statement at the `[edit routing-options rib-groups group-name]` hierarchy level. The `export` statement at the `[edit routing-options forwarding-table]` hierarchy level and the `pplb` policy enable load balancing.

```

[edit]
routing-options {
  interface-routes {
    rib-group inet bc-vrf;
  }
  rib-groups {
    bc-vrf {
      import-rib [inet.0 tcp-routing-table.inet.0 udp-routing-table.inet.0];
    }
  }
  forwarding-table {
    export pplb;
  }
}
policy-options {
  policy-statement pplb {
    then {
      load-balance per-packet;
    }
  }
}
}

```

### **Option: Using an ES PIC to Send Traffic to a Packet Analyzer**

You can send some or all of the traffic securely to the packet analyzer using IPSec and an ES PIC. In this case, the TCP traffic is encrypted, sent over an IPSec tunnel,

and received by the packet analyzer. For more information on configuring IPsec on the ES PIC, see the [IPsec Feature Guide](#) or the *JUNOS System Basics Configuration Guide*.

```
[edit]
interfaces {
  es-3/1/0 {
    unit 0 {
      tunnel {
        source 10.8.8.1;
        destination 10.8.8.2;
      }
      family inet {
        ipsec-sa sa-esp;
        address 3.3.3.1/32 {
          destination 3.3.3.2;
        }
      }
    }
  }
  fe-3/2/1 {
    unit 0 {
      family inet {
        address 10.8.8.1/30;
      }
    }
  }
}
security {
  ipsec {
    proposal esp-sha1-3des {
      protocol esp;
      authentication-algorithm hmac-sha1-96;
      encryption-algorithm 3des-cbc;
      lifetime-seconds 180;
    }
    policy esp-group2 {
      perfect-forward-secrecy {
        keys group2;
      }
      proposals esp-sha1-3des;
    }
    security-association sa-esp {
      mode tunnel;
      dynamic {
        ipsec-policy esp-group2;
      }
    }
  }
  ike {
    proposal ike-esp {
      authentication-method pre-shared-keys;
      dh-group group2;
      authentication-algorithm sha1;
      encryption-algorithm 3des-cbc;
      lifetime-seconds 180;
    }
  }
}
```

```

policy 10.8.8.2 {
    mode aggressive;
    proposals ike-esp;
    pre-shared-key ascii-text "$9$QmQnuORrIMBIds2oiH0BIESe";
}
}
}

```

### Option: Applying a Firewall Filter to an Output Interface

On output interfaces, you can apply a firewall filter that leads to a filter-based forwarding routing instance. This is useful if you want to port-mirror traffic to multiple Monitoring Services PICs or flow collection services interfaces. To configure, include the `output` statement at the `[edit interfaces interface-name unit logical-unit-number family inet filter]` hierarchy level.

```

[edit]
interfaces
fe-3/1/0 {
    description "export interface to flow collection services interfaces";
    unit 0 {
        family inet;
        address ip-address;
        filter {
            output output-filter-name;
        }
    }
}
}

```

## Using a Flow Collector Interface to Process and Export Multiple Flow Records

Basic passive monitoring can sometimes create a large number of flow records. However, you can manage multiple flow records with a flow collector interface. You can create a flow collector interface from a Monitoring Services II PIC. The flow collector interface combines multiple flow records received from a monitoring services interface into a compressed ASCII data file and exports the file to an FTP server.

To convert a Monitoring Services II PIC into a flow collector interface, include the `flow-collector` statement at the `[edit chassis fpc fpc-slot pic pic-slot monitoring-services application]` hierarchy level. To restore the monitoring functions of a Monitoring Services II PIC, include the `monitor` statement at the `[edit chassis fpc fpc-slot pic pic-slot monitoring-services application]` hierarchy level.

After you commit the configuration to convert the PIC between the `monitor` and `flow-collector` service types, you must take the PIC offline and then bring the PIC back online. Rebooting the router does not enable the new service type. You can use the Monitoring Services II PIC for either flow collection or monitoring, but not both types of service simultaneously.

A flow collector interface, designated by the `cp-fpc/pic/port` interface name, requires three logical interfaces for correct operation. Units 0 and 1 are used respectively as export channels 0 and 1 to send the compressed ASCII data files to an FTP server. You must include a class-of-service (CoS) configuration for these two export channels

to provide adequate bandwidth for file transmission. Unit 2 is used as a flow receive channel to receive flow records from a monitoring services interface.



**NOTE:** Unlike conventional interfaces, IP addresses for flow collector logical interfaces set up a point-to-point connection between the Routing Engine and the flow collector. The **address** statement at the [edit interfaces *cp-fpc/pic/port* unit *unit-number* family *inet*] hierarchy level corresponds to the IP address of the Routing Engine. Likewise, the **destination** statement at the [edit interfaces *cp-fpc/pic/port* unit *unit-number* family *inet* address *ip-address*] hierarchy level corresponds to the IP address of the flow collector interface. As a result, you must configure the **destination** statement for Units 0 and 1 (export channels 0 and 1) with *local* addresses that can reach the FTP server. Similarly, configure the **destination** statement for Unit 2 (flow receive channel) with a *local* IP address so it can reach the monitoring services interface that sends flow records.

To activate flow collector services after the Monitoring Services II PIC is converted into a flow collector, include the **flow-collector** statement at the [edit services] hierarchy level. You also need to configure several additional components:

- Destination of the FTP server—Determines where the compressed ASCII data files are sent after the flow records are collected and processed. To specify the destination FTP server, include the **destinations** statement at the [edit services **flow-collector**] hierarchy level. You can specify up to two FTP server destinations and include the password for each configured server. If two FTP servers are configured, the first server in the configuration is the primary server and the second is a backup server.
- File specifications—Preset data file formats, name formats, and transfer characteristics. Files are sent by FTP to the destination FTP server when the timer expires or when a preset number of records are received, whichever comes first. To set the data file format, include the **data-format** statement at the [edit services **flow-collector** file-specification *file-name*] hierarchy level. The default data format is **flow-compressed**. To set the export timer and file size thresholds, include the **transfer** statement at the [edit services **flow-collector** file-specification *file-name*] hierarchy level and specify values for the **timeout** and **record-level** options. The default values are 600 seconds for **timeout** and 500,000 records for **record-level**.

To set the filename format, include the **name-format** statement at the [edit services **flow-collector** file-specification *file-name*] hierarchy level. Common name format macros that you can use in your configuration are included in Table 8 on page 29.

**Table 8: Name Format Macros**

Field	Expansion
{ <i>am_pm</i> }	AM or PM
{ <i>date</i> }	Expands to the current date, using the { <i>month</i> }, { <i>day</i> }, and { <i>year</i> } macros.
{ <i>day</i> }	01 to 31

**Table 8: Name Format Macros** *(continued)*

Field	Expansion
{ <i>day_abbrev</i> }	Sun through Sat
{ <i>day_full</i> }	Sunday through Saturday
{ <i>generation_number</i> }	Expands to a unique, sequential number for each new file created.
{ <i>hour_12</i> }	01 to 12
{ <i>hour_24</i> }	00 to 23
{ <i>ifalias</i> }	Expands to a description string for the logical interface.
{ <i>minute</i> }	00 to 59
{ <i>month</i> }	01 to 12
{ <i>month_abbrev</i> }	Jan through Dec
{ <i>month_full</i> }	January through December
{ <i>num_zone</i> }	-2359 to +2359
{ <i>second</i> }	00 to 60
{ <i>time</i> }	Expands to the time the file is created, using the { <i>hour_24</i> }, { <i>minute</i> }, and { <i>second</i> } macros.
{ <i>time_zone</i> }	Time zone code name of the locale ( <b>gmt</b> , <b>pst</b> , and so on).
{ <i>year</i> }	1970, 2008, and so on.
{ <i>year_abbrev</i> }	00 to 99

- Input interface-to-flow collector interface mappings—Match an input interface with a flow collector interface and apply the preset file specifications to the input interface. To configure the default flow collector and file specifications for all input interfaces, include the **file-specification** and **collector** statements at the [edit services flow-collector interface-map] hierarchy level. To override the default settings and apply flow collector and file specifications to a specific input interface, include the **file-specification** and **collector** statements at the [edit services flow-collector interface-map *interface-name*] hierarchy level.
- Transfer log settings—Allow you to configure the destination FTP server where log files containing the transfer activity history for a flow collector interface are to be archived, the name for the log file, and the amount of time the router waits before sending the log file to the FTP server. To configure, include the **archive-sites**, **filename-prefix**, and **maximum-age** statements at the [edit services flow-collector transfer-log-archive] hierarchy level. The default value for the

**maximum-age** statement is 120 minutes, with a range of 1 to 360 minutes. Also, you can configure up to five FTP archive site servers to receive log files.

- **Miscellaneous settings**—Allow you to configure values for the IP address of the analyzer, an identifier for the analyzer, the maximum number of times the flow collector interface attempts to send transfer log files to the FTP server, and the amount of time the flow collector interface waits between retry attempts. To configure, include the **analyzer-address**, **analyzer-id**, **retry**, and **retry-delay** statements at the **[edit services flow-collector]** hierarchy level. The range for the **retry** statement is 0 through 10 retry attempts. The default for the **retry-delay** statement is 30 seconds and the range is 0 through 60 seconds.

To specify a flow collector interface as the destination for flow records coming from a Monitoring Services or Monitoring Services II PIC, include the **collector-pic** statement at the **[edit forwarding-options monitoring group-name family inet output flow-export-destination]** hierarchy level. You can select either the flow collector interface or a flow server as the destination for flow records, but you cannot select both destination types simultaneously.

There is also a Juniper Networks enterprise Management Information Base (MIB) for the flow collector interface. The Flow Collector Services MIB allows you to use SNMP to monitor the flow collector interface. The MIB provides statistics on files, records, memory, FTP, and error states of a flow collector interface. It also provides SNMP traps for unavailable destinations, unsuccessful file transfers, flow overloading, and memory overloading. For more information, see the *JUNOS Network Management Configuration Guide* or view the enterprise-specific Juniper Networks MIBs at <http://www.juniper.net/techpubs/software/junos/mibs.html>.

In summary, to implement the flow collector service, include statements at the **[edit chassis]**, **[edit interfaces]**, **[edit forwarding-options]**, and **[edit services]** hierarchy levels. The excerpt on the following pages shows the flow collector service configuration hierarchy. For a full configuration example, see “Example: Flow Collector Interface Configuration” on page 55.

```
[edit]
chassis {
  fpc fpc-slot {
    pic pic-slot {
      monitoring-services {
        application flow-collector;
      }
    }
  }
}
interfaces {
  cp-fpc/pic/port {
    description "flow_collector_interface";
    unit 0 {
      family inet {
        address ip-address {
          destination ip-address;
        }
      }
    }
  }
}
```

```

    unit 1 {
        family inet {
            address ip-address {
                destination ip-address;
            }
        }
    }
    unit 2 {
        family inet {
            address ip-address {
                destination ip-address;
            }
        }
    }
}
interface-fpc/pic/port {
    description "export_interface";
    unit 0 {
        family inet {
            address ip-address;
        }
    }
}
mo-fpc/pic/port {
    description "monitoring_services_interface";
    unit 0 {
        family inet;
    }
}
SONET/SDH, ATM2 IQ, or Ethernet-based-interface-fpc/pic/port {
    description "input_interface";
    encapsulation encapsulation-type;
    passive-monitor-mode; # Apply to the logical interface for SONET/SDH
}
}
forwarding-options {
    monitoring group1 {
        family inet {
            output {
                export-format cflowd-version-5;
                flow-active-timeout value;
                flow-inactive-timeout value;
                flow-export-destination collector-pic;
                interface mo-fpc/pic/port {
                    source-address ip-address;
                }
            }
        }
    }
}
}
services {
    flow-collector {
        analyzer-address ip-address;
        analyzer-id name;
        retry value;
        retry-delay seconds;
    }
}

```

```

destinations {
  "ftp://username@ftp-server-address-1//directory/" {
    password "encrypted-password";
  }
  "ftp://username@ftp-server-address-2//directory/" {
    password "encrypted-password";
  }
}
file-specification {
  file-specification-name {
  }
  data-format flow-compressed;
  transfer timeout value record-level size;
}
}
interface-map {
  file-specification file-specification-name;
  collector cp-fpc/pic/port;
  interface-name {
    file-specification file-specification-name;
    collector cp-fpc/pic/port;
  }
}
transfer-log-archive {
  filename-prefix filename;
  maximum-age timeout-value;
  archive-sites {
    "ftp://username@ip-address//directory/" {
      password "encrypted-password";
    }
  }
}
}
}

```

## Monitoring Traffic Using a Dynamic Flow Capture Interface

---

Dynamic flow capture enables you to capture packet flows based on filtering criteria that you specify in real time.

- Using a Dynamic Flow Capture Interface to Monitor Traffic On Demand on page 34
- Configuring the Capture Group on page 35
- Configuring the Content Destination on page 36
- Configuring the Control Source on page 36
- Configuring the Dynamic Flow Capture Interface on page 37
- Option: Configuring Thresholds on page 38
- Option: Configuring System Logging on page 38
- Option: Monitoring Dynamic Flow Capture by Using SNMP on page 39

## Using a Dynamic Flow Capture Interface to Monitor Traffic On Demand

Dynamic flow capture enables you to capture packet flows based on filtering criteria that you specify in real time. Unlike traditional flow monitoring that requires filtering criteria to be established before operation, dynamic flow capture uses an on demand control protocol that allows you to modify the filtering criteria as network conditions change.

The dynamic flow capture architecture consists of one or more *control sources* that send Dynamic Tasking Control Protocol (DTCP) requests to a *monitoring station*. The requests contain filtering criteria that specify which incoming traffic should be monitored, and the monitoring station forwards any packets that match the filter criteria to a set of one or more *content destinations*.

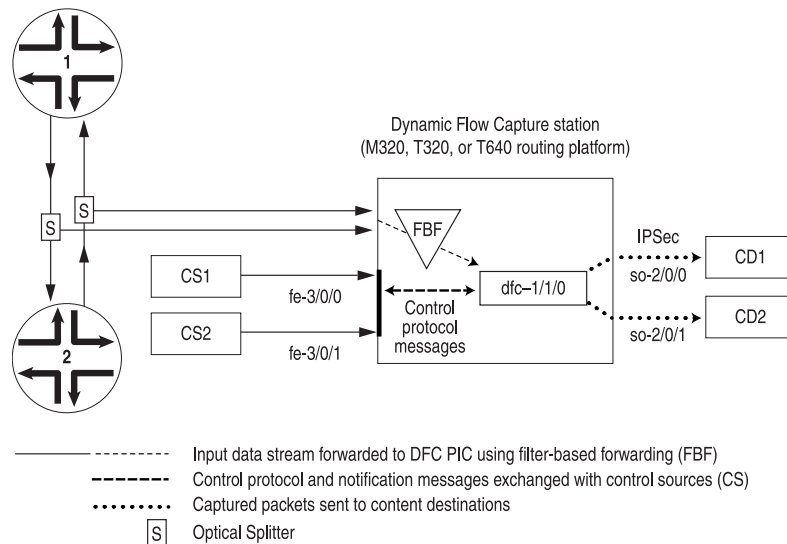
- Control source—A client that wants to monitor electronic data or voice transfer over the network. The control source sends filter requests to the Juniper Networks router using DTCP. The control source is identified by a unique identifier and an optional list of IP addresses.
- Monitoring station—A Juniper Networks T Series or M320 router configured with one or more Monitoring Services III PICs which support dynamic flow capture processing. The monitoring station processes the requests from the control sources, creates the filters, monitors incoming data flows, and sends the matched packets to the appropriate content destinations.
- Content destination—Recipient of the matched packets from the monitoring station. Typically the matched packets are sent using an IPSec tunnel from the monitoring station to another router connected to the content destination. The content destination and the control source can be located on the same host.



**NOTE:** The DFC PIC forwards the entire packet content to the content destination, rather than just a content record.

---

Figure 2 on page 35 shows a sample topology that contains control sources, a monitoring station, and content destinations.

**Figure 2: Dynamic Flow Capture Topology**

g017075

To configure dynamic flow capture, perform the following tasks:

- Configuring the Capture Group on page 35
- Configuring the Content Destination on page 36
- Configuring the Control Source on page 36
- Configuring the Dynamic Flow Capture Interface on page 37
- Option: Configuring Thresholds on page 38
- Option: Configuring System Logging on page 38
- Option: Monitoring Dynamic Flow Capture by Using SNMP on page 39

## Configuring the Capture Group

A dynamic flow capture capture group defines a profile of dynamic flow capture configuration information. The static configuration includes information about control sources, content destinations, and notification destinations. Dynamic configuration is added through interaction with control sources using a control protocol.

To configure a capture group, include the `capture-group` statement at the `[edit services dynamic-flow-capture]` hierarchy level:

```
[edit services dynamic-flow-capture]
capture-group client-name {
  content-destination identifier {
    address address;
    ttl hops;
  }
  control-source identifier {
    allowed-destinations [ destination ];
    no-syslog;
    notification-targets [ address address port port-number ];
  }
}
```

```

        service-port port-number;
        shared-key value;
        source-addresses [ address ];
    }
    input-packet-rate-threshold rate;
    interfaces interface-name;
    pic-memory-threshold percentage percentage;
}

```

To specify the **capture-group**, assign it a unique *client-name* that associates the information with the requesting control sources.

## Configuring the Content Destination

You must specify a destination for the packets that match dynamic flow capture filter criteria. To configure, include the **content-destination** statement at the [edit services dynamic-flow-capture capture-group *client-name*] hierarchy level:

```

[edit services dynamic-flow-capture capture-group client-name]
content-destination identifier {
    address address;
    ttl hops;
}

```

Assign the **content-destination** a unique *identifier*. In addition, you must specify its IP address, and you can optionally set a time-to-live (TTL) value for the IP-IP header:

- **address**—The dynamic flow capture interface appends an IP header with this destination address on the matched packet (with its own IP header and contents intact) and sends it out to the content destination.
- **ttl**—By default, the TTL value is 255, with a range from 0 through 255.

## Configuring the Control Source

You configure information about the control source, including allowed source addresses, destinations, and authentication key values. To configure the control source information, include the **control-source** statement at the [edit services dynamic-flow-capture] hierarchy level:

```

[edit services dynamic-flow-capture capture-group client-name]
control-source identifier {
    allowed-destinations [ destination-identifier ];
    no-syslog;
    notification-targets [ address address port port-number ];
    service-port port-number;
    shared-key value;
    source-addresses [ address ];
}

```

Assign the **control-source** statement with a unique *identifier*. You can also include values for the following statements:

- **allowed-destinations**—One or more content destination identifiers to which this control source can request matched data to be sent in its control protocol requests. If you do not specify any content destinations, all available destinations are allowed.
- **notification-targets**—One or more destinations to which the dynamic flow capture interface can log information about control protocol-related events and other events such as PIC startup messages. You configure each **notification-target** entry with an IP **address** value and a User Datagram Protocol (UDP) **port** number.
- **service-port**—UDP port number to which the control protocol requests are directed. Control protocol requests that are not directed to this port are discarded by dynamic flow capture interfaces.
- **shared-key**—A 20-byte authentication key value shared between the control source and the dynamic flow capture monitoring station.
- **source-addresses**—One or more allowed IP addresses from which the control source can send control protocol requests to the dynamic flow capture monitoring station. These are /32 addresses.

## Configuring the Dynamic Flow Capture Interface

You specify the interface that interacts with the control sources configured in the same dynamic flow capture group. A Monitoring Services III PIC can belong to only one capture group, and you can configure only one PIC for each group.

To configure a dynamic flow capture interface, include the **interfaces** statement at the [edit services dynamic-flow-capture] hierarchy level:

```
[edit services dynamic-flow-capture capture-group client-name]  
  interfaces interface-name;
```

You specify dynamic flow capture interfaces using the **dfc-** identifier at the [edit interfaces] hierarchy level. Three logical units are required on each dynamic flow capture interface, numbered 0, 1, and 2. You cannot configure any other logical interfaces.

- unit 0 processes control protocol requests and responses.
- unit 1 receives monitored data.
- unit 2 transmits the matched packets to the destination address.

The following example shows the configuration necessary to set up a dynamic flow capture interface:

```
[edit interfaces dfc-0/0/0]  
  unit 0 {  
    family inet {  
      address 10.1.0.0/32 { # Address of the Routing Engine for the DFC PIC.  
        destination 10.36.100.1; # Address of DFC PIC; used by the  
          # control source to correspond with the monitoring station.      }  
    }  
  }
```

```

    }
  }
}
unit 1 { # Receives data packets on this logical interface.
  family inet;
}
unit 2 { # Sends copies of matched packets from this logical interface.
  family inet;
}

```

In addition, you must configure the dynamic flow capture application to run on the DFC PIC in the correct chassis location. The following example shows this configuration at the [edit chassis] hierarchy level:

```

[edit chassis]
fpc 0 {
  pic 0 {
    monitoring-services application dynamic-flow-capture;
  }
}

```

For more information on configuring chassis properties, see the *JUNOS System Basics Configuration Guide*.

### Option: Configuring Thresholds

You can optionally specify threshold values for situations in which warning messages will be recorded in the system log:

- Input packet rate to the dynamic flow capture interfaces
- Memory usage on the dynamic flow capture interfaces

To configure, include the `input-packet-rate-threshold` or `pic memory-threshold` statements at the [edit services dynamic-flow-capture capture-group *client-name*] hierarchy level:

```

[edit services dynamic-flow-capture capture-group client-name]
input-packet-rate-threshold rate;
pic-memory-threshold percentage percentage;

```

If these statements are not configured, no threshold messages are logged. The threshold settings are configured for the capture group as a whole.

### Option: Configuring System Logging

By default, control protocol activity is logged as a separate system log facility, `dfc`. To modify the filename or level at which control protocol activity is recorded, include the following statements at the [edit syslog] hierarchy level:

```

[edit syslog]
file dfc.log {
  dfc any;
}

```

To cancel logging, include the `no-syslog` statement at the `[edit services dynamic-flow-capture capture-group client-name control-source identifier]` hierarchy level:

```
[edit services dynamic-flow-capture capture-group client-name control-source identifier]
no-syslog;
```

### Option: Monitoring Dynamic Flow Capture by Using SNMP

In JUNOS Release 7.5 and later, the Dynamic Flow Capture MIB provides a way to monitor dynamic flow capture information by using Simple Network Management Protocol (SNMP). The MIB provides the same information that you can view with the `show services dynamic-flow-capture content-destination`, `show services dynamic-flow-capture control-source`, and `show services dynamic-flow-capture statistics` commands. For more information, see the *JUNOS Network Management Configuration Guide*.

## Hardware and Software Considerations

There are several hardware and software considerations when you implement passive flow monitoring. When defining the hardware requirements of the monitoring station, keep in mind the following:

- The input interfaces on the monitoring station must be SONET/SDH interfaces (OC3, OC12, or OC48), ATM2 IQ interfaces (OC3 or OC12), 4-port Fast Ethernet interfaces, Gigabit Ethernet interfaces with SFP (4-port or 10-port), or 1-port 10-Gigabit Ethernet interfaces with XENPAK.
- To monitor the flows in both directions for a single interface, the monitoring station must have two SONET/SDH, ATM2 IQ, or Ethernet-based receive ports, one for each direction of flow. In Figure 1 on page 4, the monitoring station needs one port to monitor the traffic flowing from Router 1 to Router 2, and a second port to monitor the traffic flowing from Router 2 to Router 1.
- The Monitoring Services PICs must be installed in a Type 1 enhanced FPC slot.
- Type 1 and Type 2 Tunnel Services PICs are supported.
- Use an ES PIC to encrypt the flow export.

When defining a traffic monitoring strategy, keep in mind the following:

- The monitoring station collects only IPv4 packets. All other packet formats are discarded and not counted.
- You can set the amount of time a data flow can be inactive before the monitoring station terminates the flow and exports the flow data. To set the timer, include the `flow-inactive-timeout` statement at the `[edit forwarding-options monitoring group-name family inet output]` hierarchy level. The timer value can be from 15 seconds through 1800 seconds, with a default value of 60 seconds.

You can also configure the monitoring station to collect periodic flow reports for flows that last longer than the configured active timeout. To set this activity timer, include the `flow-active-timeout` statement at the `[edit forwarding-options monitoring`

*group-name family inet output*] hierarchy level. The timer value can be from 60 seconds through 1800 seconds, with a default value of 180 seconds.

- Multiple expired flows are exported together, if possible. A UDP packet is sent when one of the following conditions is met:
  - When 30 flows are contained in the current packet, the flows are exported.
  - If there are fewer than 30 flows but the export timer expires, the flows are exported one second after the timer expires.
- TCP and UDP flows are considered differently:
  - TCP flows watch for a segment containing the **FIN** bit and a subsequent acknowledgement (**ACK**) to detect the end of a flow. Alternately, a TCP reset (**RST**) can also indicate the end of a flow. When these TCP combinations are detected, the flow expires. The **FIN+ACK** and **RST** cases cover most TCP stream closures. For all other flows, an inactive timeout is needed.
  - All non-TCP flows, such as UDP, depend on timeout mechanisms for export.
- The default MTU value for SONET/SDH interfaces is 4474 bytes; for Gigabit Ethernet and Fast Ethernet interfaces, it is 1500 bytes. If the monitoring station receives packets exceeding 4474 bytes, they are discarded; no fragmentation is performed. Note that the supported MTU size on the Gigabit Ethernet or Fast Ethernet PICs might exceed 1500 bytes, depending on the type of PIC.
- Any incoming traffic that is discarded is not forwarded to packet analyzers.
- The interfaces on the monitoring station that collect intercepted traffic must be configured with Cisco HDLC or PPP encapsulation.
- You must always use a standard interface (for example, one that follows the usual *interface-name-fpc/pic/slot* format) to send flow records to a flow server. Flow data generated by the Monitoring Services or Monitoring Services II PICs will not be delivered to the server across the **fxp0** interface.
- You can send version 5 records to multiple flow servers. You can configure up to eight servers and flow traffic is load-balanced between the servers in a round-robin fashion. If one of the servers ceases operation, flow traffic load-balances automatically between the remaining active servers. To configure, include up to eight **flow-server** statements at the [*edit forwarding-options monitoring group-name output*] hierarchy level.

## Chapter 3

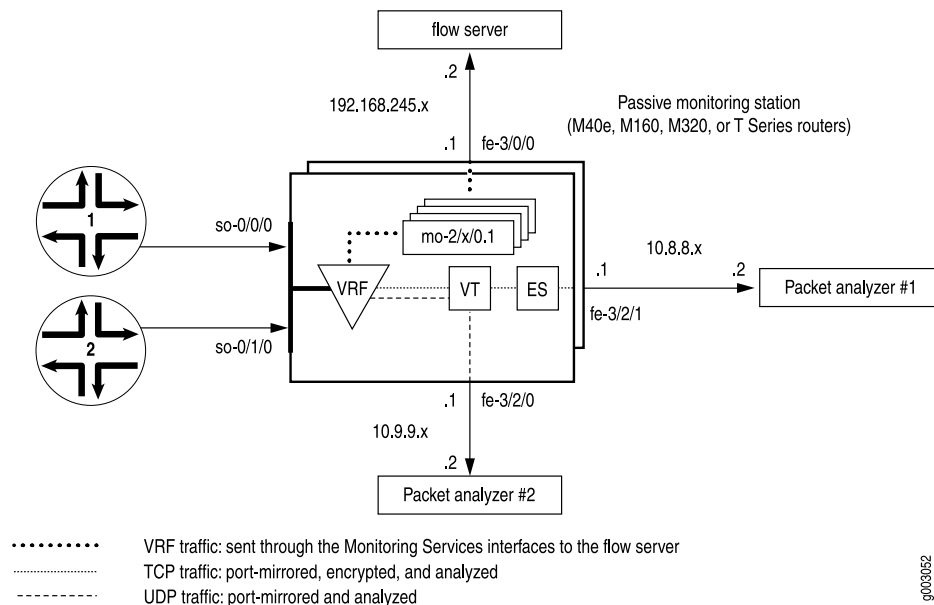
# Passive Flow Monitoring Configuration Examples

This section contains configuration examples and commands you can issue to verify a passive flow monitoring configuration:

- Example: Passive Flow Monitoring Configuration on page 41
- Example: Flow Collector Interface Configuration on page 55
- Example: Dynamic Flow Capture Configuration on page 65

## Example: Passive Flow Monitoring Configuration

**Figure 3: Passive Flow Monitoring—Topology Diagram**



In Figure 3 on page 41, traffic enters the monitoring station through interfaces so-0/0/0 and so-0/1/0. After the firewall filter accepts the traffic to be monitored, the packets enter a VRF instance.

The original packets travel within the VRF instance to the Monitoring Services PIC for flow processing. The final flow packets are sent from the monitoring services interfaces out the **fe-3/0/0** interface to a flow server.

A copy of the accepted traffic is port-mirrored to the Tunnel PIC. As the copied packets enter the tunnel interface, a second firewall filter separates TCP and UDP packets and places them into two filter-based forwarding instances. The UDP instance directs the UDP packets to a packet analyzer attached to **fe-3/2/0**. The TCP instance sends the TCP packets to the ES PIC for encryption and the ES PIC sends the packets to a second packet analyzer connected to **fe-3/2/1**.

Your first step is to define a firewall filter to select packets for monitoring. All filtered traffic must be accepted, and the **port-mirror** statement at the [edit firewall family inet filter *filter-name* term *term-name* then] hierarchy level facilitates port mirroring.

Next, configure the input SONET/SDH interfaces and apply the firewall filter that you just defined. The **passive-monitor-mode** statement disables SONET keepalives on the SONET/SDH interfaces and enables passive flow monitoring.

Configure all other interfaces that you will use with the monitoring application, including the monitoring services interfaces, the export interfaces, the tunnel interface, and the ES interface. Once the interfaces are in place, configure a VRF instance and monitoring group to direct the original packets from the input interfaces to the monitoring services interfaces for processing. The resulting flow description packets exit **fe-3/0/0** to reach the flow server.

Next, configure statements to port-mirror the monitored traffic to a tunnel interface. Design a firewall filter that selects some of this copied traffic for further analysis and some of the traffic for discarding. In this case, isolate TCP and UDP traffic and direct these two flows into separate filter-based forwarding routing instances. Remember to apply the filter to the tunnel interface to enable the separation of TCP traffic from UDP traffic. Also, import the interface routes into the forwarding instances with a routing table group.

In the filter-based forwarding instances, define static route next hops. The next hop for the TCP instance is the ES interface and the next hop for the UDP instance is the packet analyzer connected to **fe-3/2/0**. Finally, configure IPsec so that the next hop for the TCP traffic is the second packet analyzer attached to **fe-3/2/1**.

```
[edit]
interfaces {
  so-0/0/0 { # Traffic enters the router on this interface.
    description " input interface";
    encapsulation ppp;
    unit 0 {
      passive-monitor-mode; # Disables SONET keepalives.
      family inet {
        filter {
          input input-monitoring-filter; # The firewall filter is applied here.
        }
      }
    }
  }
  so-0/1/0 { # Traffic enters the router on this interface.
```

```

description " input interface";
encapsulation ppp;
unit 0 {
    passive-monitor-mode; # Disables SONET keepalives.
    family inet {
        filter {
            input input-monitoring-filter; # The firewall filter is applied here.
        }
    }
}
}
es-3/1/0 { # This is where the TCP traffic enters the ES PIC.
    unit 0 {
        tunnel {
            source 10.8.8.1;
            destination 10.8.8.2;
        }
        family inet {
            ipsec-sa sa-esp;
            address 3.3.3.1/32 {
                destination 3.3.3.2;
            }
        }
    }
}
fe-3/0/0 { # Flow records exit here and travel to the flow server.
    description " export interface to the flow server";
    unit 0 {
        family inet;
        address 192.168.245.1/30;
    }
}
fe-3/2/0 { # This export interface for UDP traffic leads to a packet analyzer.
    description " export interface to the packet analyzer";
    unit 0 {
        family inet {
            address 10.9.9.1/30;
        }
    }
}
fe-3/2/1 { # This IPSec tunnel source exports TCP traffic to a packet analyzer.
    unit 0 {
        family inet {
            address 10.8.8.1/30;
        }
    }
}
mo-4/0/0 { # This marks the beginning of the monitoring services interfaces.
    unit 0 { # Unit 0 is part of the inet.0 routing table and generates flow records.
        family inet;
    }
    unit 1 { # Unit 1 receives monitored traffic and is part of the VRF instance.
        family inet;
    }
}
mo-4/1/0 {

```

```

    unit 0 { # Unit 0 is part of the inet.0 routing table and generates flow records.
        family inet;
    }
    unit 1 { # Unit 1 receives monitored traffic and is part of the VRF instance.
        family inet;
    }
}
mo-4/2/0 {
    unit 0 { # Unit 0 is part of the inet.0 routing table and generates flow records.
        family inet;
    }
    unit 1 { # Unit 1 receives monitored traffic and is part of the VRF instance.
        family inet;
    }
}
mo-4/3/0 {
    unit 0 { # Unit 0 is part of the inet.0 routing table and generates flow records.
        family inet;
    }
    unit 1 { # Unit 1 receives monitored traffic and is part of the VRF instance.
        family inet;
    }
}
vt-0/2/0 { # The tunnel services interface receives the port-mirrored traffic.
    unit 0 {
        family inet {
            filter {
                input tunnel-interface-filter; # The filter splits traffic into TCP and UDP
            }
        }
    }
}
}
forwarding-options {
    monitoring group1 { # Monitored traffic is processed by the monitoring services
        family inet { # interfaces and flow records are sent to the flow server.
            output {
                export-format cflowd-version-5;
                flow-active-timeout 60;
                flow-inactive-timeout 30;
                flow-server 192.168.245.2 port 2055; # IP address and port for server.
                interface mo-4/0/0.1 { # Use monitoring services interfaces for output.
                    engine-id 1; # engine and interface-index statements are optional.
                    engine-type 1;
                    input-interface-index 44;
                    output-interface-index 54;
                    source-address 192.168.245.1; # This is the IP address of fe-3/0/0.
                }
                interface mo-4/1/0.1 {
                    engine-id 2; # engine and interface-index statements are optional.
                    engine-type 1;
                    input-interface-index 45;
                    output-interface-index 55;
                    source-address 192.168.245.1; # This is the IP address of fe-3/0/0.
                }
                interface mo-4/2/0.1 {

```

```

        engine-id 3; # engine and interface-index statements are optional.
        engine-type 1;
        input-interface-index 46;
        output-interface-index 56;
        source-address 192.168.245.1; # This is the IP address of fe-3/0/0.
    }
    interface mo-4/3/0.1 {
        engine-id 4; # engine and interface-index statements are optional.
        engine-type 1;
        input-interface-index 47;
        output-interface-index 57;
        source-address 192.168.245.1; # This is the IP address of fe-3/0/0.
    }
}

port-mirroring { # Copies the traffic and sends it to the Tunnel Services PIC.
    family inet {
        input {
            rate 1;
            run-length 1;
        }
        output {
            interface vt-0/2/0.0;
            no-filter-check;
        }
    }
}

routing-options { # This installs the interface routes into the forwarding instances.
    interface-routes {
        rib-group inet bc-vrf;
    }
    rib-groups {
        bc-vrf {
            import-rib [inet.0 tcp-routing-table.inet.0 udp-routing-table.inet.0];
        }
    }
    forwarding-table {
        export pplb; # Applies per-packet load balancing to the forwarding table.
    }
}

policy-options {
    policy-statement monitoring-vrf-import {
        then reject;
    }
    policy-statement monitoring-vrf-export {
        then reject;
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}

security { # This sets IPSec options for the ES PIC.

```

```

ipsec {
  proposal esp-sha1-3des {
    protocol esp;
    authentication-algorithm hmac-sha1-96;
    encryption-algorithm 3des-cbc;
    lifetime-seconds 180;
  }
  policy esp-group2 {
    perfect-forward-secrecy {
      keys group2;
    }
    proposals esp-sha1-3des;
  }
  security-association sa-esp {
    mode tunnel;
    dynamic {
      ipsec-policy esp-group2;
    }
  }
}
ike {
  proposal ike-esp {
    authentication-method pre-shared-keys;
    dh-group group2;
    authentication-algorithm sha1;
    encryption-algorithm 3des-cbc;
    lifetime-seconds 180;
  }
  policy 10.8.8.2 {
    mode aggressive;
    proposals ike-esp;
    pre-shared-key ascii-text "$9$QmQnuORrIMBIds2oiH0BIESe";
  }
}
}
firewall {
  family inet {
    filter input-monitoring-filter { # This filter selects traffic to send into the VRF
      term 1 { # instance and prepares the traffic for port mirroring.
        from {
          destination-address {
            10.7.0.0/16;
          }
        }
        then {
          port-mirror;
          accept;
        }
      }
      term 2 {
        from {
          destination-address {
            10.6.0.0/16;
          }
        }
        then accept;
      }
    }
  }
}

```

```

    }
  }
  filter tunnel-interface-filter { # This filter breaks the port-mirrored traffic into two
    term tcp { # filter-based forwarding instances: TCP packets and UDP packets.
      from {
        protocol tcp;
      }
      then { # This counts TCP packets and sends them into a TCP instance.
        count tcp;
        routing-instance tcp-routing-table;
      }
    }
    term udp {
      from {
        protocol udp;
      }
      then { # This counts UDP packets and sends them into a UDP instance.
        count udp;
        routing-instance udp-routing-table;
      }
    }
    term rest {
      then {
        count rest;
        discard;
      }
    }
  }
}
}
routing-instances {
  monitoring-vrf { # This is the VRF instance where you send the traffic. It contains
    instance-type vrf; # the input interface and the monitoring services interfaces.
    interface so-0/0/0.0; # Traffic enters the router on these input interfaces.
    interface so-0/1/0.0;
    interface mo-4/0/0.1;
    interface mo-4/1/0.1; # These are output interfaces (use them as
    interface mo-4/2/0.1; # output interfaces in your monitoring group).
    interface mo-4/3/0.1;
    route-distinguisher 69:1;
    vrf-import monitoring-vrf-import;
    vrf-export monitoring-vrf-export;
    routing-options { # Sends traffic to a group of monitoring services interfaces.
      static {
        route 0.0.0.0/0 next-hop [mo-4/0/0.1 mo-4/1/0.1
          mo-4/2/0.1 mo-4/3/0.1];
      }
    }
  }
}
tcp-routing-table { # This is the filter-based forwarding instance for TCP traffic.
  instance-type forwarding;
  routing-options { # The next hop is the ES PIC.
    static {
      route 0.0.0.0/0 next-hop es-3/1/0.0;
    }
  }
}

```

```

    }
    udp-routing-table { # This is the filter-based forwarding instance for UDP traffic.
        instance-type forwarding;
        routing-options { # The next hop is the second packet analyzer.
            static {
                route 0.0.0.0/0 next-hop 10.9.1.2;
            }
        }
    }
}

```

## Verifying Your Work

To verify that your configuration is correct, use the following commands on the monitoring station that is configured for passive flow monitoring:

- `show route 0/0`
- `show passive-monitoring error`
- `show passive-monitoring flow`
- `show passive-monitoring memory`
- `show passive-monitoring status`
- `show passive-monitoring usage`

To clear statistics for the `show passive-monitoring error` and `show passive-monitoring flow` commands, issue the `clear passive-monitoring (all | interface-name)` command.

You can also view passive flow monitoring status with the Simple Network Management Protocol (SNMP). The following Management Information Base (MIB) tables are supported:

- `jnxPMonErrorTable`—Corresponds to the `show passive-monitoring error` command.
- `jnxPMonFlowTable`—Corresponds to the `show passive-monitoring flow` command.
- `jnxPMonMemoryTable`—Corresponds to the `show passive-monitoring memory` command.

The following section shows the output of the `show` commands used with the configuration example:

```

user@mon-station> show route 0/0
<skip inet.0>

```

# We are only concerned with the routing-instance route.

```

bc-vrf.inet.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
bc-vrf.inet.0:+ = Active Route, - = Last Active, * = Both
0.0.0.0/0          *[Static/5] 5d 17:34:57
                   via mo-4/0/0.1
                   > via mo-4/1/0.1
                   via mo-4/2/0.1
                   via mo-4/3/0.1

```

```

tcp-rt.inet.0: 13 destinations, 13 routes (12 active, 0 holddown, 1
hidden)
+ = Active Route, - = Last Active, * = Both
0.0.0.0/0          *[Static/5] 19:24:39
                   > via es-3/1/0.0
                   : <other interface routes>
udp-rt.inet.0: 13 destinations, 13 routes (12 active, 0 holddown, 1
hidden)
+ = Active Route, - = Last Active, * = Both
0.0.0.0/0          *[Static/5] 19:24:39
                   > to 10.9.1.2 via fe-3/2/0.0
                   : <other interface routes>

```



**NOTE:** For all `show passive-monitoring` commands, the output obtained when using a wildcard (such as `*`) or the `all` option is based on the configured interfaces listed at the `[edit forwarding-options monitoring group-name]` hierarchy level. In the output from the configuration example, you see information only for the configured interfaces `mo-4/0/0`, `mo-4/1/0`, `mo-4/2/0`, and `mo-4/3/0`.

Many of the statements you can configure in a monitoring group, such as `engine-id` and `engine-type`, are visible in the output of the `show passive-monitoring` commands.

**Table 9: Output Fields for the `show passive-monitoring error` Command**

Field	Explanation
Packets dropped (no memory)	Number of packets dropped because of memory.
Packets dropped (not IP)	Number of non-IP packets dropped.
Packets dropped (not IPv4)	Number of packets dropped because they failed the IPv4 check.
Packets dropped (header too small)	Number of packets dropped because the packet length or IP header length was too small.
Memory allocation failures	Number of flow record memory allocation failures. A small number reflects failures to replenish the free list. A large number indicates the monitoring station is almost out of memory space.
Memory free failures	Number of flow record memory frees.
Memory free list failures	Number of flow records received from free list that failed. Memory is nearly exhausted or too many new flows greater than 128K are being created in one second.
Memory warning	The flows have exceeded 1 million packets per second (Mpps) on a Monitoring Services PIC or 2 Mpps on a Monitoring Services II PIC. The response can be <b>Yes</b> or <b>No</b> .
Memory overload	The memory has been overloaded. The response is <b>Yes</b> or <b>No</b> .

**Table 9: Output Fields for the show passive-monitoring error Command** (continued)

Field	Explanation
PPS overload	In packets per second, whether the PIC is receiving more traffic than the configured threshold. The response can be Yes or No.
BPS overload	In bytes per second, whether the PIC is receiving more traffic than the configured threshold. The response can be Yes or No.

```

user@mon-station> show passive-monitoring error all
Passive monitoring interface: mo-4/0/0, Local interface index: 44
Error information
  Packets dropped (no memory): 0, Packets dropped (not IP): 0
  Packets dropped (not IPv4): 0, Packets dropped (header too small): 0
  Memory allocation failures: 0, Memory free failures: 0
  Memory free list failures: 0
  Memory warning: No, Memory overload: No, PPS overload: No, BPS overload: No

Passive monitoring interface: mo-4/1/0, Local interface index: 45
Error information
  Packets dropped (no memory): 0, Packets dropped (not IP): 0
  Packets dropped (not IPv4): 0, Packets dropped (header too small): 0
  Memory allocation failures: 0, Memory free failures: 0
  Memory free list failures: 0
  Memory warning: No, Memory overload: No, PPS overload: No, BPS overload: No

Passive monitoring interface: mo-4/2/0, Local interface index: 46
Error information
  Packets dropped (no memory): 0, Packets dropped (not IP): 0
  Packets dropped (not IPv4): 0, Packets dropped (header too small): 0
  Memory allocation failures: 0, Memory free failures: 0
  Memory free list failures: 0
  Memory warning: No, Memory overload: No, PPS overload: No, BPS overload: No

Passive monitoring interface: mo-4/3/0, Local interface index: 47
Error information
  Packets dropped (no memory): 0, Packets dropped (not IP): 0
  Packets dropped (not IPv4): 0, Packets dropped (header too small): 0
  Memory allocation failures: 0, Memory free failures: 0
  Memory free list failures: 0
  Memory warning: No, Memory overload: No, PPS overload: No, BPS overload: No

```

**Table 10: Output Fields for the show passive-monitoring flow Command**

Field	Explanation
Flow packets	Number of packets received by an operational PIC.
Flow bytes	Number of bytes received by an operational PIC.

**Table 10: Output Fields for the show passive-monitoring flow Command** (continued)

Field	Explanation
Flow packets 10-second rate	Number of packets per second handled by the PIC and displayed as a 10-second average.
Flow bytes 10-second rate	Number of bytes per second handled by the PIC and displayed as a 10-second average.
Active flows	Number of currently active flows tracked by the PIC.
Total flows	Total number of flows received by an operational PIC.
Flows exported	Total number of flows exported by an operational PIC.
Flows packets exported	Total number of flow packets exported by an operational PIC.
Flows inactive timed out	Total number of flows that are exported because of inactivity.
Flows active timed out	Total number of long-lived flows that are exported because of an active timeout.

```

user@mon-station> show passive-monitoring flow all
Passive monitoring interface: mo-4/0/0, Local interface index: 44
  Flow information
    Flow packets: 6533434, Flow bytes: 653343400
    Flow packets 10-second rate: 0, Flow bytes 10-second rate: 0
    Active flows: 0, Total flows: 1599
    Flows exported: 1599, Flows packets exported: 55
    Flows inactive timed out: 1599, Flows active timed out: 0

Passive monitoring interface: mo-4/1/0, Local interface index: 45
  Flow information
    Flow packets: 6537780, Flow bytes: 653778000
    Flow packets 10-second rate: 0, Flow bytes 10-second rate: 0
    Active flows: 0, Total flows: 1601
    Flows exported: 1601, Flows packets exported: 55
    Flows inactive timed out: 1601, Flows active timed out: 0

Passive monitoring interface: mo-4/2/0, Local interface index: 46
  Flow information
    Flow packets: 6529259, Flow bytes: 652925900
    Flow packets 10-second rate: 0, Flow bytes 10-second rate: 0
    Active flows: 0, Total flows: 1599
    Flows exported: 1599, Flows packets exported: 55
    Flows inactive timed out: 1599, Flows active timed out: 0

Passive monitoring interface: mo-4/3/0, Local interface index: 47
  Flow information
    Flow packets: 6560741, Flow bytes: 656074100
    Flow packets 10-second rate: 0, Flow bytes 10-second rate: 0

```

Active flows: 0, Total flows: 1598  
 Flows exported: 1598, Flows packets exported: 55  
 Flows inactive timed out: 1598, Flows active timed out: 0

**Table 11: Output Fields for the show passive-monitoring memory Command**

Field	Explanation
Allocation count	Number of flow records allocated.
Free count	Number of flow records freed.
Maximum allocated	Maximum number of flow records allocated since the monitoring station booted. This number represents the peak number of flow records allocated at a time.
Allocations per second	Flow records allocated per second during the last statistics interval on the PIC.
Frees per second	Flow records freed per second during the last statistics interval on the PIC.
Total memory used	Total amount of memory currently used (in bytes).
Total memory free	Total amount of memory currently free (in bytes).

```
user@mon-station> show passive-monitoring memory all
```

```
Passive monitoring interface: mo-4/0/0, Local interface index: 44
Memory utilization
  Allocation count: 1600, Free count: 1599, Maximum allocated: 1600
  Allocations per second: 3200, Frees per second: 1438
  Total memory used (in bytes): 103579176, Total memory free (in bytes): 163914184

Passive monitoring interface: mo-4/1/0, Local interface index: 45
Memory utilization
  Allocation count: 1602, Free count: 1601, Maximum allocated: 1602
  Allocations per second: 3204, Frees per second: 1472
  Total memory used (in bytes): 103579176, Total memory free (in bytes): 163914184

Passive monitoring interface: mo-4/2/0, Local interface index: 46
Memory utilization
  Allocation count: 1600, Free count: 1599, Maximum allocated: 1600
  Allocations per second: 3200, Frees per second: 1440
  Total memory used (in bytes): 103579176, Total memory free (in bytes): 163914184

Passive monitoring interface: mo-4/3/0, Local interface index: 47
Memory utilization
  Allocation count: 1599, Free count: 1598, Maximum allocated: 1599
  Allocations per second: 3198, Frees per second: 1468
  Total memory used (in bytes): 103579176, Total memory free (in bytes): 163914184
```

**Table 12: Output Fields for the show passive-monitoring status Command**

Field	Explanation
Interface state	Indicates whether the interface is monitoring (operating properly), disabled (administratively disabled), or not monitoring (not configured).
Group index	Integer that represents the monitoring group of which the PIC is a member. (This does not indicate the number of monitoring groups.)
Export interval	Configured export interval for flow records, in seconds.
Export format	Configured export format (only v5 is currently supported).
Protocol	Protocol the PIC is configured to monitor (only IPv4 is currently supported).
Engine type	Configured engine type that is inserted in output flow packets.
Engine ID	Configured engine ID that is inserted in output flow packets.
Route record count	Number of routes recorded.
IFL to SNMP index count	Number of logical interfaces mapped to an SNMP index.
AS count	Number of AS boundaries that the flow has crossed.
Time set	Indicates whether the time stamp is in place.
Configuration set	Indicates whether the monitoring configuration is set.
Route record set	Indicates whether routes are being recorded.
IFL SNMP map set	Indicates whether logical interfaces are being mapped to an SNMP index.

```

user@mon-station> show passive-monitoring status all
Passive monitoring interface: mo-4/0/0, Local interface index: 44
  Interface state: Monitoring
  Group index: 0
  Export interval: 15 secs, Export format: cflowd v5
  Protocol: IPv4, Engine type: 1, Engine ID: 1
  Route record count: 13, IFL to SNMP index count: 30, AS count: 1
  Time set: Yes, Configuration set: Yes
  Route record set: Yes, IFL SNMP map set: Yes

Passive monitoring interface: mo-4/1/0, Local interface index: 45
  Interface state: Monitoring
  Group index: 0
  Export interval: 15 secs, Export format: cflowd v5
  Protocol: IPv4, Engine type: 1, Engine ID: 2
  Route record count: 13, IFL to SNMP index count: 30, AS count: 1
  Time set: Yes, Configuration set: Yes
  Route record set: Yes, IFL SNMP map set: Yes

Passive monitoring interface: mo-4/2/0, Local interface index: 46
  Interface state: Monitoring
  Group index: 0

```

```

Export interval: 15 secs, Export format: cflowd v5
Protocol: IPv4, Engine type: 1, Engine ID: 3
Route record count: 13, IFL to SNMP index count: 30, AS count: 1
Time set: Yes, Configuration set: Yes
Route record set: Yes, IFL SNMP map set: Yes

Passive monitoring interface: mo-4/3/0, Local interface index: 47
Interface state: Monitoring
Group index: 0
Export interval: 15 secs, Export format: cflowd v5
Protocol: IPv4, Engine type: 1, Engine ID: 4
Route record count: 13, IFL to SNMP index count: 30, AS count: 1
Time set: Yes, Configuration set: Yes
Route record set: Yes, IFL SNMP map set: Yes

```

**Table 13: Output Fields for the show passive-monitoring usage Command**

Field	Explanation
Uptime	Time, in milliseconds, that the PIC has been operational.
Interrupt time	Cumulative time that the PIC spent in processing packets since the last PIC reset.
Load (5 second)	CPU load on the PIC averaged over 5 seconds. The number is a percentage obtained by dividing the time spent on active tasks by the total elapsed time.
Load (1 minute)	CPU load on the PIC averaged over 1 minute. The number is a percentage obtained by dividing the time spent on active tasks by the total elapsed time.

```

user@mon-station> show passive-monitoring usage *
Passive monitoring interface: mo-4/0/0, Local interface index: 44
CPU utilization
  Uptime: 653155 milliseconds, Interrupt time: 40213754 microseconds
  Load (5 second): 20%, Load (1 minute): 17%

Passive monitoring interface: mo-4/1/0, Local interface index: 45
CPU utilization
  Uptime: 652292 milliseconds, Interrupt time: 40223178 microseconds
  Load (5 second): 22%, Load (1 minute): 15%

Passive monitoring interface: mo-4/2/0, Local interface index: 46
CPU utilization
  Uptime: 649491 milliseconds, Interrupt time: 40173645 microseconds
  Load (5 second): 22%, Load (1 minute): 10098862%

Passive monitoring interface: mo-4/3/0, Local interface index: 47
CPU utilization
  Uptime: 657328 milliseconds, Interrupt time: 40368704 microseconds
  Load (5 second): 1%, Load (1 minute): 15%

```

## Example: Flow Collector Interface Configuration

**Figure 4: Flow Collector Interface Topology Diagram**

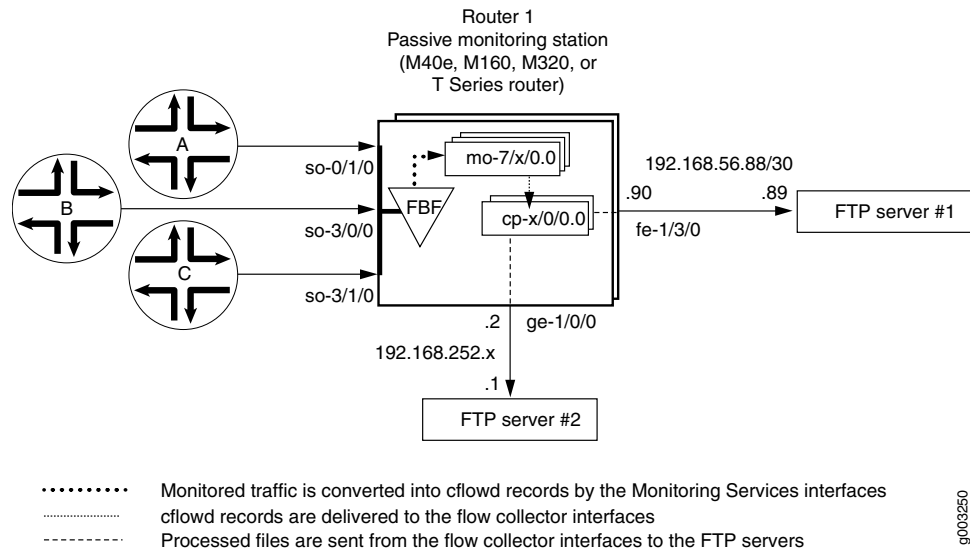


Figure 4 on page 55 shows the path traveled by monitored traffic as it passes through the router. Packets arrive at input interfaces `so-0/1/0`, `so-3/0/0`, and `so-3/1/0`. The raw packets are directed into a filter-based forwarding routing instance and processed into flow records by the monitoring services interfaces `mo-7/1/0`, `mo-7/2/0`, and `mo-7/3/0`. The flow records are compressed into files at the flow collector interfaces `cp-6/0/0` and `cp-7/0/0` and sent to the FTP server for analysis. Finally, a mandatory class-of-service (CoS) configuration is applied to export channels 0 and 1 on the flow collector interfaces to manage the outgoing processed files.

```
Router 1 [edit]
chassis {
  fpc 6 {
    pic 0 {
      monitoring-services {
        application flow-collector; # This converts a Monitoring Services II PIC
      } # into a flow collector interface.
    }
  }
  fpc 7 {
    pic 0 {
      monitoring-services {
        application flow-collector; # This converts a Monitoring Services II PIC
      } # into a flow collector interface.
    }
  }
}
interfaces {
  cp-6/0/0 {
    unit 0 {# Logical interface .0 on a flow collector interface is export
      family inet { # channel 0 and sends records to the FTP server.

```

```

    filter {
        output cp-ftp; # Apply the CoS filter here.
    }
    address 10.0.0.1/32 {
        destination 10.0.0.2;
    }
}
unit 1 { # Logical interface .1 on a flow collector interface is export
family inet { # channel 1 and sends records to the FTP server.
    filter {
        output cp-ftp; # Apply the CoS filter here.
    }
    address 10.1.1.1/32 {
        destination 10.1.1.2;
    }
}
unit 2 { # Logical interface .2 on a flow collector interface is the flow
family inet { # receive channel that communicates with the Routing Engine.
    address 10.2.2.1/32 { # Do not apply a CoS filter on logical interface .2.
        destination 10.2.2.2;
    }
}
}
}
cp-7/0/0 {
unit 0 { # Logical interface .0 on a flow collector interface is export
family inet { # channel 0 and sends records to the FTP server.
    filter {
        output cp-ftp; # Apply the CoS filter here.
    }
    address 10.3.3.1/32 {
        destination 10.3.3.2;
    }
}
unit 1 { # Logical interface .1 on a flow collector interface is export
family inet { # channel 1 and sends records to the FTP server.
    filter {
        output cp-ftp; # Apply the CoS filter here.
    }
    address 10.4.4.1/32 {
        destination 10.4.4.2;
    }
}
unit 2 { # Logical interface .2 on a flow collector interface is the flow
family inet { # receive channel that communicates with the Routing Engine.
    address 10.5.5.1/32 { # Do not apply a CoS filter on logical interface .2.
        destination 10.5.5.2;
    }
}
}
}
fe-1/3/0 { # This is the exit interface leading to the first FTP server.

```

```

    unit 0 {
        family inet {
            address 192.168.56.90/30;
        }
    }
}
ge-1/0/0 { # This is the exit interface leading to the second FTP server.
    unit 0 {
        family inet {
            address 192.168.252.2/24;
        }
    }
}
mo-7/1/0 { # This is the first interface that creates flow records.
    unit 0 {
        family inet;
    }
}
mo-7/2/0 { # This is the second interface that creates flow records.
    unit 0 {
        family inet;
    }
}
mo-7/3/0 { # This is the third interface that creates flow records.
    unit 0 {
        family inet;
    }
}
so-0/1/0 { # This is the first input interface that receives traffic to be monitored.
    encapsulation ppp;
    unit 0 {
        passive-monitor-mode; # This allows the interface to be passively monitored.
        family inet {
            filter {
                input catch; # The filter-based forwarding filter is applied here.
            }
        }
    }
}
so-3/0/0 { # This is the second interface that receives traffic to be monitored.
    encapsulation ppp;
    unit 0 {
        passive-monitor-mode; # This allows the interface to be passively monitored.
        family inet {
            filter {
                input catch; # The filter-based forwarding filter is applied here.
            }
        }
    }
}
so-3/1/0 { # This is the third interface that receives traffic to be monitored.
    encapsulation ppp;
    unit 0 {
        passive-monitor-mode; # This allows the interface to be passively monitored.
        family inet {
            filter {

```

```

        input catch; # The filter-based forwarding filter is applied here.
    }
}
}
}
forwarding-options {
    monitoring group1 { # Always define your monitoring group here.
        family inet {
            output {
                export-format cflowd-version-5;
                flow-active-timeout 60;
                flow-inactive-timeout 15;
                flow-export-destination collector-pic; # Sends records to the flow collector.
                interface mo-7/1/0.0 {
                    source-address 192.168.252.2;
                }
                interface mo-7/2/0.0 {
                    source-address 192.168.252.2;
                }
                interface mo-7/3/0.0 {
                    source-address 192.168.252.2;
                }
            }
        }
    }
}
routing-options {
    interface-routes {
        rib-group inet common;
    }
    rib-groups {
        common {
            import-rib [ inet.0 fbf_instance.inet.0 ];
        }
    }
    forwarding-table {
        export pplb;
    }
}
policy-options {
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
class-of-service { # A class-of-service configuration for the flow collector interface
    interfaces { # is mandatory when implementing flow collector services.
        cp-6/0/0 {
            scheduler-map cp-map;
        }
        cp-7/0/0 {
            scheduler-map cp-map;
        }
    }
}

```

```

scheduler-maps {
  cp-map {
    forwarding-class best-effort scheduler Q0;
    forwarding-class expedited-forwarding scheduler Q1;
    forwarding-class network-control scheduler Q3;
  }
}
schedulers {
  Q0 {
    transmit-rate remainder;
    buffer-size percent 90;
  }
  Q1 {
    transmit-rate percent 5;
    buffer-size percent 5;
    priority strict-high;
  }
  Q3 {
    transmit-rate percent 5;
    buffer-size percent 5;
  }
}
}
firewall {
  family inet {
    filter cp-ftp { # This filter provides CoS for flow collector interface traffic.
      term t1 {
        then forwarding-class expedited-forwarding;
      }
    }
  }
  filter catch { # This firewall filter sends incoming traffic into the
    interface-specific; # filter-based forwarding routing instance.
    term def {
      then {
        count counter;
        routing-instance fbf_instance;
      }
    }
  }
}
routing-instances {
  fbf_instance { # This instance sends traffic to the monitoring services interface.
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop mo-7/1/0.0;
      }
    }
  }
}
}
services {
  flow-collector { # Define properties for flow collector interfaces here.
    analyzer-address 10.10.10.1; # This is the IP address of the analyzer.
    analyzer-id server1; # This helps to identify the analyzer.
    retry 3; # Maximum number of attempts by the PIC to send a file transfer log.
  }
}

```

```

retry-delay 30; # The time interval between attempts to send a file transfer log.
destinations { # This defines the FTP servers that receive flow collector output.
  "ftp://user@192.168.56.89//tmp/collect1/" { # The primary FTP server.
    password "$9$IXJK8xN-w2oZdbZDHmF3001"; # SECRET-DATA
  }
  "ftp://user@192.168.252.1//tmp/collect2/" { # The second FTP server.
    password "$9$elbvL7-dsgaGVwGjkP3nOBI"; # SECRET-DATA
  }
}
file-specification { # Define sets of flow collector characteristics here.
  def-spec {
  }
  data-format flow-compressed; # The default compressed output format.
}
f1 {
  name-format "cFlowd-py69Ni69-0-%D_%T-%I_%N.bcp.bi.gz";
  data-format flow-compressed; # The default compressed output format.
  transfer timeout 1800 record-level 1000000; # Here are configured values.
}
}
interface-map { # Allows you to map interfaces to flow collector interfaces.
  file-specification def-spec; # Flows generated for default traffic are sent to the
  collector cp-7/0/0; # default flow collector interface cp-7/0/0.
  so-0/1/0.0 { # Flows generated for the so-0/1/0 interface are sent
    collector cp-6/0/0; # to cp-6/0/0, and the file-specification used is "default".
  }
  so-3/0/0.0 { # Flows generated for the so-3/0/0 interface are sent
    file-specification f1; # to cp-6/0/0, and the file-specification used is "f1."
    collector cp-6/0/0;
  }
  so-3/1/0.0; # Because no settings are defined, flows generated for this
}
transfer-log-archive { # Sends flow collector interface log files to an FTP server.
  filename-prefix so_3_0_0_log;
  maximum-age 15;
  archive-sites {
    "ftp://user@192.168.56.89//tmp/transfers/" {
      password "$9$IFaEyevMXNVsWLsgaU.m6/C";
    }
  }
}
}

```

## Verifying Your Work

To verify that your flow collector configuration is working, use the following commands on the monitoring station that is configured for flow collection:

- clear services flow-collector statistics
- request services flow-collector change-destination (primary | secondary)
- request services flow-collector test-file-transfer
- show services flow-collector file interface (detail | extensive | terse)

- `show services flow-collector (detail | extensive)`
- `show services flow-collector input interface (detail | extensive | terse)`

The following section shows the output of the `show` commands used with the configuration example:

```
user@router1> show services flow-collector input interface cp-6/0/0 detail
Interface                               Packets      Bytes
mo-7/1/0.0                             6170        8941592

user@router1> show services flow-collector interface all detail
Flow collector interface: cp-6/0/0
Interface state: Collecting flows
  Packets      Bytes      Flows Uncompressed   Compressed   FTP bytes FTP files
                Bytes      Bytes      Bytes      Bytes
        6736   9757936   195993   21855798   3194148           0           0
Flow collector interface: cp-7/0/0
Interface state: Collecting flows
  Packets      Bytes      Flows Uncompressed   Compressed   FTP bytes FTP files
                Bytes      Bytes      Bytes      Bytes
           0         0         0         0         0           0           0

user@router1> show services flow-collector input interface cp-6/0/0 extensive
Interface                               Packets      Bytes
mo-7/1/0.0                             6260        9074096

user@router1> show services flow-collector interface cp-6/0/0 extensive
Flow collector interface: cp-6/0/0
Interface state: Collecting flows
Memory:
  Used: 19593212, Free: 479528656
Input:
  Packets: 6658, per second: 0, peak per second: 0
  Bytes: 9647752, per second: 12655, peak per second: 14311
  Flow records processed: 193782, per second: 252, peak per second: 287
Allocation:
  Blocks allocated: 174, per second: 0, peak per second: 0
  Blocks freed: 0, per second: 0, peak per second: 0
  Blocks unavailable: 0, per second: 0, peak per second: 0
Files:
  Files created: 1, per second: 0, peak per second: 0
  Files exported: 0, per second: 0, peak per second: 0
  Files destroyed: 0, per second: 0, peak per second: 0
Throughput:
  Uncompressed bytes: 21075152, per second: 52032, peak per second: 156172
  Compressed bytes: 3079713, per second: 7618, peak per second: 22999
Packet drops:
  No memory: 0, Not IP: 0
  Not IPv4: 0, Too small: 0
  Fragments: 0, ICMP: 0
  TCP: 0, Unknown: 0
  Not JUNOS flow: 0
File Transfer:
  FTP bytes: 0, per second: 0, peak per second: 0
  FTP files: 0, per second: 0, peak per second: 0
  FTP failure: 0
Export channel: 0
Current server: Secondary
Primary server state: OK, Secondary server state: OK
```

```

Export channel: 1
  Current server: Secondary
  Primary server state: OK, Secondary server state: OK

user@router1> show services flow-collector file interface cp-6/0/0 terse
File name                               Flows State
cFlowd-py69Ni69-0-20031112_014301-so_3_0_0_0.bcp.bi.gz 185643 Active

user@router1> show services flow-collector file interface cp-6/0/0 detail
Filename: cFlowd-py69Ni69-0-20031112_014301-so_3_0_0_0.bcp.bi.gz
Throughput:
  Flow records: 187067, Uncompressed bytes: 21121960, Compressed bytes: 2965643

Status:
  State: Active, Transfer attempts: 0

user@router1> show services flow-collector file interface cp-6/0/0 extensive
Filename: cFlowd-py69Ni69-0-20031112_014301-so_3_0_0_0.bcp.bi.gz
Throughput:
  Flow records: 188365, per second: 238, peak per second: 287
  Uncompressed bytes: 21267756, per second: 27007, peak per second: 32526
  Compressed bytes: 2965643, per second: 0, peak per second: 22999
Status:
  Compressed blocks: 156, Block count: 156
  State: Active, Transfer attempts: 0

```

To clear statistics for a flow collector interface, issue the `clear services flow-collector statistics interface (all | interface-name)` command.

Another useful flow collector option allows you to change the FTP server from primary to secondary and test for FTP transfers. To force the flow collector interface to use a primary or secondary FTP server, include the `primary` or `secondary` option when you issue the `request services flow-collector change-destination interface cp-fpc/pic/port` command.

If you configure only one primary server and issue this command with the `primary` option, you receive the error message “Destination change not needed.” If the secondary server is not configured and you issue this command with the `secondary` option, you receive the error message “Destination not configured.” Otherwise, when both servers are configured properly, successful output appears as follows.

```

user@router1> request services flow-collector change-destination interface
cp-6/0/0 primary
Flow collector interface: cp-6/0/0
Interface state: Collecting flows
Destination change successful

user@router1> request services flow-collector change-destination interface
cp-6/0/0 secondary
Flow collector interface: cp-6/0/0
Interface state: Collecting flows
Destination change successful

```

Other options for the `request services flow-collector change-destination interface cp-fpc/pic/port` command are `immediately` (which forces an instant switchover), `gracefully` (the default behavior that allows a gradual switchover), `clear-files` (which purges existing data files), and `clear-logs` (which purges existing log files).

To verify that transfer log files are being scheduled for delivery to the FTP servers, issue the `request services flow-collector test-file-transfer filename interface cp-fpc/pic/port` command. Include the desired export channel (zero or one) and target FTP server (primary or secondary) with this command.

```
user@router> request services flow-collector test-file-transfer test_file
interface cp-6/0/0 channel-one primary
Flow collector interface: cp-6/0/0
Interface state: Collecting flows
Response: Test file transfer successfully scheduled
```

Another way you can check for the success of your file transfers is by analyzing the transfer log. A transfer log sends detailed information about files that are collected and processed by the flow collector interface. Table 14 on page 63 explains the various fields available in the transfer log.

**Table 14: Flow Collector Interface Transfer Log Fields**

Field	Explanation
fn	Filename
sz	File size
nr	Number of records
ts	Timestamp with the format of year (4 digits), month (2 digits), day (2 digits), hours (2 digits), minutes (2 digits), and seconds (2 digits).
sf	Success flag—The values are 1 for success and 0 for failure.
ul	Server URL
rc	FTP result code
er	FTP error text
tt	Transfer time

This is an example of a successful transfer log:

```
fn="cFlowd-py69Ni69-0-20040227_230438-at_4_0_0_4_3.bcp.bi.gz":sz=552569
:nr=20000:ts="20040227230855":sf=1:ul="ftp://10.63.152.1/tmp/server1/":rc=250:
er="":tt=3280
```

This is an example of a transfer log when an FTP session fails:

```
fn="cFlowd-py69Ni69-0-20040227_230515-at_4_0_0_2_8.bcp.bi.gz":sz=560436
:nr=20000:ts="20040227230855":sf=1:ul="ftp://10.63.152.1/tmp/server1/":rc=250:
er="":tt=3290
```

As the flow collector interface receives and processes flow records, the PIC services logging process (fsad) handles the following tasks:

- When the flow collector interface transfers a file to the FTP server, a temporary log file is created in the `/var/log/flowc` directory. The temporary log file has this filenaming convention:

`< hostname > _ < filename_prefix > _ YYYYMMDD_hhmmss.tmp`

*hostname* is the hostname of the transfer server, *filename\_prefix* is the same value defined with the `filename-prefix` statement at the `[edit services flow-collector transfer-log-archive]` hierarchy level, *YYYYMMDD* is the year, month, and date, and *hhmmss* is the timestamp indicating hours, minutes, and seconds.

- After the log file has been stored in the router for the length of time specified by the `maximum-age` statement at the `[edit services flow-collector transfer-log-archive]` hierarchy level (the default is 120 minutes), the temporary log file is converted to an actual log file and the temporary file is deleted. The new log file retains the same naming conventions, except the extension is `*.log`.
- When the final log file is created and compressed, the PIC services logging process (fsad) tries to send the log file from the `/var/log/flowc` directory to an FTP server. You can specify up to five FTP servers to receive the log files by including the `archive-sites` statement at the `[edit services flow-collector transfer-log-archive]` hierarchy level. The logging process attempts to send the log file to one server at a time, in order of their appearance in the configuration. Upon the first successful transfer, the log file is deleted and the logging process stops sending log files to the remaining FTP servers in the list.
- If the log file transfer is not successful, the log file is moved to the `/var/log/flowc/failed` directory. Every 30 minutes, the logging process tries to resend the log files. After the log files are transferred successfully, they are deleted from the `/var/log/flowc/failed` directory.



**NOTE:** If the memory for a flow collector interface is full, the interface might drop incoming packets.

After the flow collector interface successfully delivers the processed information file to the FTP server, you can analyze the file. The file contains detailed information about the flows collected and processed by the flow collector interface. Table 15 on page 64 explains the various fields available in the flow collector interface file.

**Table 15: Flow Collector Interface File Fields in Order of Appearance**

Field	Explanation
linkDir	Link directory—A randomly generated number used to identify the record
analyzer-address	Analyzer address
analyzer-ID	Analyzer identifier

**Table 15: Flow Collector Interface File Fields in Order of Appearance** *(continued)*

Field	Explanation
ifAlias	Interface identifier
source-address	Source address
destination-address	Destination address
packets	Number of packets
bytes	Number of bytes
start-time	Start time
end-time	End time
source-port	Source port
destination-port	Destination port
tcp_flag	TCP flag
protocol	IP protocol number
src_AS_number	Source AS number
dst_AS_number	Destination AS number

This is an example of output from a flow collector interface file:

```
11799241612374557782|10.10.10.1|server1|at_4_0_0_4|192.168.10.100|10.0.0.1|8|
3136|1077926402|1077926402|8224|12336|27|6|0|0
```

## Example: Dynamic Flow Capture Configuration

The following example shows a complete dynamic flow capture configuration. On Router 1, configure the dynamic flow capture interface, the interfaces that connect to the control source and content destination, and the interface that receives passively monitored traffic. Then, configure the capture group and specify your control source and content destination requirements. Next, configure filter-based forwarding (FBF) to send monitored traffic to logical unit 1 of the dynamic flow capture interface. Finally, configure a firewall filter and routing table groups to complete the configuration.

```
[edit]
interfaces {
  dfc-0/0/0 { # DFC PIC that processes requests from the control source.
    unit 0 {
      family inet {
        address 2.1.0.0/32 { # Address of the Routing Engine for the DFC PIC.
          destination 10.36.100.1; # Address of DFC PIC; used by
```

```

        } # the control source to communicate with the monitoring station.
    }
}
unit 1 { # This logical interface receives data packets.
    family inet;
}
unit 2 { # This logical interface sends out copies of matched packets.
    family inet;
}
}
fe-4/1/2 { # Interface that receives filtering requests from cs1.
    unit 0 {
        family inet {
            address 10.36.41.2/30;
        }
    }
}
ge-7/0/0 { # Interface that sends monitored packets to cd1.
    unit 0 {
        family inet {
            address 10.36.70.1/30;
        }
    }
}
so-1/2/0 { # Interface that receives traffic to be monitored.
    encapsulation ppp;
    unit 0 {
        passive-monitor-mode; # Enables this interface to be passively monitored.
        family inet {
            filter {
                input catch;
            }
        }
    }
}
}
services {
    dynamic-flow-capture {
        capture-group g1 {
            interfaces dfc-0/0/0; # Specifies which interface to use for DFC processing.
            input-packet-rate-threshold 90k; # Traffic threshold for system log messages.
            pic-memory-threshold percentage 80; # Memory threshold for log messages.
            control-source cs1 { # Specifies addresses and ports for the control source.
                source-addresses 10.36.41.1;
                service-port 2400;
                notification-targets {
                    10.36.41.1 port 2100;
                }
                shared-key "$9$ASxdsYoX7wg4aHk";
                allowed-destinations cd1;
            }
            content-destination cd1 { # Specifies content destination addresses and TTL.
                address 10.36.70.2;
                ttl 244;
            }
        }
    }
}

```

```

    }
  }
  firewall {
    filter catch { # Places monitored traffic into the filter-based forwarding instance.
      interface-specific;
      term def {
        then {
          count counter;
          routing-instance fbf_inst;
        }
      }
    }
  }
  routing-instances {
    fbf_inst { # Sends matching traffic to the DFC PIC for processing.
      instance-type forwarding;
      routing-options {
        static {
          route 0.0.0.0/0 next-hop dfc-0/0/0.1;
        }
      }
    }
  }
  routing-options {
    interface-routes {
      rib-group inet common;
    }
    rib-groups {
      common { # Shares routes between the instance and the main routing table.
        import-rib [ inet.0 fbf_inst.inet.0 ];
      }
    }
    forwarding-table {
      export pplb;
    }
  }
}

```

## Verifying Your Work

To verify that your dynamic flow capture configuration is operating correctly, issue the following command:

```
show services dynamic-flow-capture capture-group group-name control-source
source-identifier source-id (detail)
```

The following section shows the output of this command when used with the configuration example.

### Router 1

```
user@router1> show services dynamic-flow-capture control-source capture-group g1 source-identifier cs2
detail
```

```

Capture group: g1, Control source: cs2
Criteria added: 1, Criteria add failed: 0
Active criteria: 2
Static criteria: 0, Dynamic criteria: 2
Control protocol requests: 3
      Add      Delete      List      Refresh      No-op
Requests      1          0          1          0          1
Failed        0          0          0          0          0

Add request rate: 0
Add request peak rate: 1
Bandwidth across all criteria: 0
Total notifications: 0
Restart: 0, Rollover: 0, No-op: 0, Timeout: 0, Congestion: 0, Congestion delete: 0,
Dups dropped: 0
Criteria deleted: 0
Timeout idle: 0, Timeout total: 0, Packets: 0, Bytes: 0
Sequence number: 242

```

To clear dynamic flow capture criteria belonging to a particular control source, issue the **clear services dynamic-flow-capture** command. For more information on other dynamic flow capture-related operational mode commands, see the *JUNOS System Basics and Services Command Reference*.

## Chapter 4

# Active Flow Monitoring

- Configuring Active Flow Monitoring on page 69
- Defining a Firewall Filter to Select Traffic for Active Flow Monitoring on page 72
- Configuring the Interfaces That Will Be Actively Monitored on page 73
- Enabling the Monitoring Services, Adaptive Services, or Multiservices Interfaces and the Export Interface on page 73
- Flow Records Collection on page 74
- Option: Configuring Port Mirroring on page 81
- Option: Configuring Port Mirroring with Filter-Based Forwarding and a Monitoring Group on page 82
- Option: Sending Traffic to Multiple Export Interfaces by Using Next-Hop Groups on page 82
- Option: Using the Flow-Tap Application to Send Packets to a Mediation Device on page 83

## Configuring Active Flow Monitoring

In active flow monitoring, the router participates in both the monitoring application and in the normal routing functionality of the network. Although the Monitoring Services PIC was designed initially for use as an offline passive flow monitoring tool, it can also be used in an active flow monitoring topology.

Table 16 on page 69 shows which Juniper Networks PICs and corresponding routers support active flow monitoring. For more information on Juniper Networks PICs, see the PIC guide that corresponds to your router.

**Table 16: Passive and Active Flow Monitoring PIC Support**

PIC Type and Service	J Series	M5/M10	M7i/M10i	M20	M40e	M120	M160	T Series/ M320	TX Matrix
Monitoring Services PIC: active flow monitoring	No	Yes (version 8 only)	Yes	Yes	Yes	No	Yes (version 8 only)	No	No

**Table 16: Passive and Active Flow Monitoring PIC Support** (*continued*)

PIC Type and Service	J Series	M5/M10	M7i/M10i	M20	M40e	M120	M160	T Series/ M320	TX Matrix
Monitoring Services II PIC: flow collection services	No	No	No	No	Yes	No	Yes (version 8 only)	No	No
Adaptive Services PIC: active flow monitoring	No	Yes (version 8 only)	Yes	Yes	Yes	No	Yes (version 8 only)	No	No
Adaptive Services II PIC: active flow monitoring	No	Yes (version 8 only)	Yes	Yes	Yes	Yes	Yes (version 8 only)	Yes	Yes
Adaptive Services II PIC: flow-tap services	No	No	Yes	Yes	Yes	Yes	No	Yes	No
MultiServices 100 PIC: active flow monitoring	No	No	Yes	No	Yes	No	No	Yes	Yes
MultiServices 400 PIC: active flow monitoring	No	No	No	No	Yes	Yes	No	Yes	Yes
MultiServices 500 PIC: active flow monitoring	No	No	No	No	Yes	Yes	No	Yes	Yes
JUNOS Software-enabled active flow monitoring	Yes (version 8 only)	No	No	No	No	No	No	No	No

Specified packets can be filtered and sent to the monitoring interface. For the Monitoring Services PIC, the interface name contains the **mo-** prefix. For the Adaptive Services PICs and MultiServices PICs, the interface name contains the **sp-** prefix.



**NOTE:** If you upgrade from the Monitoring Services PIC to the Adaptive Services PIC or MultiServices PIC for active flow monitoring, you must modify the interface name of your monitoring interface from **mo-fpc/pic/port** to **sp-fpc/pic/port**.

The major active flow monitoring actions you can configure at the **[edit forwarding-options]** hierarchy level are as follows:

- Sampling, with the **[edit forwarding-options sampling]** hierarchy. This option extracts limited information (such as the source and destination IP address) from a copy of some of the packets in a flow, while the original packets are forwarded to the intended destination. This option is extended to define active sampling on a per Packet Forwarding Engine basis by defining a sampling instance that specifies a name for the sampling parameters and binding the instance to the particular Packet Forwarding Engine.
- Templates, with the **[edit forwarding-options sampling]** and **[edit services monitoring]** hierarchies. With active flow monitoring support for version 5, version 8, and the customizing version 9, you can use templates to organize the data gathered from sampling.
- Discard accounting, with the **[edit forwarding-options accounting]** hierarchy. This option quarantines unwanted packets, creates flow monitoring records that describe the packets, and discards the packets instead of forwarding them.
- Port mirroring, with the **[edit forwarding-options port-mirroring]** hierarchy. This option makes one full copy of all packets in a flow and delivers the copy to a single destination.
- Multiple port mirroring, with the **[edit forwarding-options next-hop-group]** hierarchy. This option allows multiple copies of selected traffic to be delivered to multiple destinations. (Multiple port mirroring requires a Tunnel Services PIC.)
- Flow-tap services processing, with the **[edit services flow-tap]** hierarchy. This option sends copies of packets that match dynamic filter criteria to one or more content destinations.

Unlike passive flow monitoring, you do not need to configure a monitoring group. Instead, you can send filtered packets to a monitoring services or adaptive services interface (**mo-** or **sp-**) by using sampling or discard accounting. Optionally, you can configure port mirroring or multiple port mirroring to direct packets to additional interfaces.

These active flow monitoring options provide a wide variety of actions that can be performed on network traffic flows. However, the following restrictions apply:

- The router can perform either sampling *or* port mirroring at any one time.
- The router can perform either forwarding *or* discard accounting at any one time.

Because the Monitoring Services PIC, Adaptive Services PIC, and MultiServices PIC allow only one action to be performed at any one time, the following configuration options are available:

- Sampling and forwarding
- Sampling and discard accounting
- Port mirroring and forwarding
- Port mirroring and discard accounting
- Sampling and port mirroring on different sets of traffic

To configure active flow monitoring, complete these steps:

- Defining a Firewall Filter to Select Traffic for Active Flow Monitoring on page 72
- Configuring the Interfaces That Will Be Actively Monitored on page 73
- Enabling the Monitoring Services, Adaptive Services, or Multiservices Interfaces and the Export Interface on page 73
- Collecting Flow Records on page 74
- Option: Configuring Port Mirroring on page 81
- Option: Configuring Port Mirroring with Filter-Based Forwarding and a Monitoring Group on page 82
- Option: Sending Traffic to Multiple Export Interfaces by Using Next-Hop Groups on page 82
- Option: Using the Flow-Tap Application to Send Packets to a Mediation Device on page 83

## Defining a Firewall Filter to Select Traffic for Active Flow Monitoring

---

The first step in active flow monitoring is to configure the match conditions for acceptable traffic or quarantined traffic. Common match actions for active flow monitoring include **sample**, **discard accounting**, **port-mirror**, and **accept**. To configure, include the desired action statements and a counter as part of the **then** statement in a firewall filter and apply the filter to an interface.

In sampling, the router reviews a portion of the traffic and sends reports about this sample to the flow monitoring server. Discard accounting traffic is counted and monitored, but not forwarded out of the router. Port-mirrored traffic is copied and sent to another interface. Accepted traffic is forwarded to the intended destination.

Most of these match combinations are valid. However, you can either port-mirror or sample with the same traffic at the same time, but not perform more than one action simultaneously on the same packets.

```
[edit]
firewall {
  family inet {
    filter active_filter {
      term quarantined_traffic {
        from {
          source-address {
            10.36.1.2/32;
          }
        }
        then {
          count quarantined-counter;
          sample;
          discard accounting;
        }
      }
      term copy_and_forward_the_rest {
        then {
```

```

        port-mirror;
        accept;
    }
}
}
}

```

## Configuring the Interfaces That Will Be Actively Monitored

---

Configure the input interfaces and apply the firewall filter that you defined earlier. Unlike passive flow monitoring, the input interfaces for active flow monitoring are not restricted, so you can select most standard network interfaces (such as ATM1 or Ethernet-based interfaces) as the input.

If you configure active flow monitoring with sampling, you can configure an interface filter in place of a firewall filter with the **sampling** statement at the **[edit interfaces interface-name-fpc/port unit unit-number family inet]** hierarchy level.

```

[edit]
interfaces {
  so-2/2/0 {
    unit 0 {
      family inet {
        filter {
          input active_filter;
        }
        address 10.36.11.2/32 {
          destination 10.36.11.1;
        }
        sampling {
          (input | output | [input output]);
        }
      }
    }
  }
}

```

## Enabling the Monitoring Services, Adaptive Services, or Multiservices Interfaces and the Export Interface

---

You configure the monitoring services, adaptive services, or multiservices interfaces with the **family inet** statement so they can process IPv4 traffic. However, you must remember that a monitoring services interface uses an **mo-** prefix and adaptive services and multiservices interfaces use an **sp-** prefix.

```

[edit]
interfaces {
  sp-2/0/0 {
    unit 0 {
      family inet {
        address 10.36.100.1/32 {
          destination 10.36.100.2;
        }
      }
    }
  }
}

```

```

    }
  }
}

```

Active flow monitoring records leave the router through an export interface to reach the flow monitoring server.

```

[edit]
interfaces {
  fe-1/0/0 {
    unit 0 {
      family inet {
        address 10.60.2.2/30;
      }
    }
  }
}

```

## Flow Records Collection

---

Traffic flows can be exported in various flow monitoring version formats for active flow monitoring.

- Collecting Flow Records on page 74
- Collecting Flow Records with a Sampling Group on page 75
- Collecting Flow Records with an Accounting Group on page 76
- Replicating Routing Engine-Based Sampling to Multiple Flow Servers on page 77
- Collecting Flow Records with a Template on page 78
- Routing Engine-Based Sampling to Multiple Flow Servers on page 79
- Replicating Version 9 Flow Aggregation to Multiple Flow Servers on page 80
- Option: Configuring an Aggregate Export Timer on page 81

### Collecting Flow Records

Traffic flows can be exported in flow monitoring version 5, 8, and 9 formats for active flow monitoring. The default export format for flow monitoring records is version 5. To change the export format to flow monitoring version 8, include the `version 8` statement at either the `[edit forwarding-options accounting name output flow-server flow-server-address]` or the `[edit forwarding-options sampling output flow-server flow-server-address]` hierarchy level. To change the export format to flow monitoring version 9, include the `version9 template template-name` statement at the `[edit forwarding-options sampling output flow-server flow-server-address]` hierarchy level. For more information on flow record formats, see “Flow Monitoring Output Formats” on page 107.

To capture flow data generated by the Monitoring Services PIC, Adaptive Services PIC, or MultiServices PIC and export it to a flow server, you can use one of the following active flow monitoring methods:

- Collecting Flow Records with a Sampling Group on page 75
- Collecting Flow Records with an Accounting Group on page 76
- Replicating Routing Engine-Based Sampling to Multiple Flow Servers on page 77
- Collecting Flow Records with a Template on page 78
- Routing Engine-Based Sampling to Multiple Flow Servers on page 79
- Replicating Version 9 Flow Aggregation to Multiple Flow Servers on page 80
- Option: Configuring an Aggregate Export Timer on page 81

### **Collecting Flow Records with a Sampling Group**

If your needs for active flow monitoring are simple, you can collect flow records with a sampling group. Sampling does not require you to configure a monitoring group (as required in passive flow monitoring) because you can configure flow server information in the **sampling** hierarchy. When you wish to sample traffic, include the **sampling** statement at the **[edit forwarding-options]** hierarchy level.

The typical sampling configuration has one input interface and one export interface. The input interface is activated by the **then sample** statement in a firewall filter term. This match condition directs traffic to the sampling process. Alternatively, you can use an interface-based filter in place of a firewall filter if you include the **sampling** statement at the **[edit interfaces interface-name-fpc/pic/port unit unit-number family inet]** hierarchy level.

There are two types of sampling available: PIC-based sampling and Routing Engine-based sampling. PIC-based sampling occurs when a monitoring services or adaptive services interface is the target for the output of the sampling process. To enable PIC-based sampling, include the **interface** statement at the **[edit forwarding-options sampling output]** hierarchy level and specify a monitoring services or adaptive services interface as the output interface. If an output interface is not specified in the sampling configuration, sampling is performed by the Routing Engine.

To specify a flow server in a sampling configuration, include the **flow-server** statement at the **[edit forwarding-options sampling output]** hierarchy level. You must specify the IP address, port number, and flow monitoring version of the destination flow server. Routing Engine-based sampling supports flow aggregation of up to eight flow servers (version 5 servers and version 8 only) at a time. The export packets are replicated to all flow servers configured to receive them. In contrast, PIC-based sampling allows you to specify just one version 5 flow server and one version 8 server simultaneously. Flow servers operating simultaneously must have different IP addresses.

As part of the output interface statements, you must configure a source address. In contrast, the interface-level statements of **engine-id** and **engine-type** are both added automatically. However, you can override these values with manually configured statements to track different flows with a single flow collector, as needed. When you configure sampling, SNMP input and output interface index information is captured in flow records by default.

```
[edit]
forwarding-options {
  sampling {
    input {
      family inet {
        rate 1;
      }
    }
    output {
      flow-server 10.60.2.1 {
        port 2055;
        version 5;
      }
      flow-inactive-timeout 15;
      flow-active-timeout 60;
      interface sp-2/0/0 {
        engine-id 5;
        engine-type 55;
        source-address 10.60.2.2;
      }
    }
  }
}
```

### ***Collecting Flow Records with an Accounting Group***

To perform discard accounting on specified traffic, you can collect flow records with the **accounting** statement at the **[edit forwarding-options]** hierarchy level. Like sampling, your topology must be simple (for example, one input interface and one export interface).

Again, you can collect flow records by specifying input and output interfaces. You can configure the input interface to perform discard accounting by applying a firewall filter that contains the **then discard accounting** statement. This match condition directs the filtered traffic to be converted into flow records and exported for analysis by the monitoring services or adaptive services interface. The original packets are then sent to the discard process. For the output, remember to specify the IP address and port of your flow server and the services interface you plan to use for processing flow records.

You must configure a source address, but the **engine-id** and **engine-type** output interface statements are added automatically. You can override these values manually to track different flows with a single flow collector. SNMP input and output interface index information is captured in flow records by default when you configure discard accounting.

```
[edit]
forwarding-options {
  accounting counter1 {
    output {
      flow-inactive-timeout 65;
      flow-active-timeout 65;
      flow-server 10.60.2.1 {
        port 2055;
```

```

        version 8;
        aggregation {
            protocol-port;
            source-destination-prefix;
        }
    }
    interface sp-2/0/0 {
        engine-id 1;
        engine-type 11;
        source-address 10.60.2.2;
    }
}
}
}

```

### Replicating Routing Engine-Based Sampling to Multiple Flow Servers

Routing Engine-based sampling supports up to eight flow servers for both flow monitoring version 5 and version 8 configurations. The total number of flow servers is limited to eight, regardless of how many are configured for version 5 or version 8.

When you configure version 5 or version 8 sampling, the export packets are replicated to all flow servers configured to receive them. If two flow servers are configured to receive version 5 records, both flow servers will receive records for a specified flow.



**NOTE:** With Routing-Engine-based sampling, if multiple flow servers are configured with version 8 export format, all of them must use the same aggregation type (for example, all flow servers receiving version 8 export could be configured for source-destination aggregation type).

The following configuration example allows replication of export packets to two flow servers.

```

[edit]
forwarding-options {
    sampling {
        input {
            family inet {
                rate 1;
            }
        }
        output {
            flow-server 10.10.3.2 {
                port 2055;
                version 5;
                source-address 192.168.164.119;
            }
            flow-server 172.17.20.62 {
                port 2055;
                version 5;
                source-address 192.168.164.119;
            }
        }
    }
}

```

```

    }
  }
}

```

## Collecting Flow Records with a Template

Flow monitoring version 9, which is based upon RFC 3954, provides a way to organize flow data into templates. Version 9 also provides a way to actively monitor MPLS traffic in addition to IPv4 and IPv6 traffic. Version 9 is not supported on the AS-I PIC.

To activate templates in flow monitoring, you must configure a template and include that template in the version 9 flow monitoring configuration. Version 9 does not work in conjunction with versions 5 and 8.

To configure a version 9 template, include the **template *template-name*** statement at the [edit services flow-monitoring version9] hierarchy level. The JUNOS Software supports four different templates: **ipv4-template**, **ipv6-template**, **mpls-template**, and **mpls-ipv4-template**. To view the fields selected in each of these templates, see “Version 9 Formats and Fields” on page 117.

```

[edit]
services flow-monitoring {
  version9 { # Specifies flow monitoring version 9.
    template mpls { # Specifies template you are configuring.
      template-refresh-rate {
        packets 6000; # The default is 4800 packets and the range is 1–480000
        # packets.
        seconds 90; # The default is 60 seconds and the range is 1–600 seconds.
        option-refresh-rate {
          packets 3000; # The default is 4800 packets and the range is 1–480000
          # packets.
          seconds 30; # The default is 60 seconds and the range is 1–600.
          flow-active-timeout 60; # The default is 60 seconds and the range is
          # 10–600.
          flow-inactive-timeout 30; # The default is 60 seconds and the range 10–600.
          template-refresh-rate seconds 10; # The default is 60 seconds and the
          # range is 10–600
          option-refresh-rate seconds 10; # The default is 60 seconds and the range
          # is 10–600 seconds.
          mpls-template {
            label-positions [1 | 2 | 3]; # Specifies label position for the MPLS template.
          }
        }
      }
    }
  }
}

```

You can export to multiple templates at a time to a maximum of eight flow servers for AS PICs and one flow server for all other PICs. To assign a template to a flow output, include the **template *template-name*** statement at the [edit forwarding options sampling output flow-server version9] hierarchy level:

```

[edit]
forwarding-options {

```

```

sampling {
  input {
    family mpls {
      rate 1;
      run-length 1;
    }
  }
  output {
    flow-server 10.60.2.1 { # The IP address and port of the flow server.
      port 2055;
      source-address 3.3.3.1;
      version9 { # Records are sent to the flow server using version 9 format.
        template { # Indicates a template will organize records.
          mpls; # Records are sent to the MPLS template.
        }
      }
    }
  }
}

```

### ***Routing Engine-Based Sampling to Multiple Flow Servers***

Routing Engine-based sampling supports up to eight flow servers for both version 5 and version 8 configurations. The total number of collectors is limited to eight, regardless of how many are configured for version 5 or version 8. When you configure sampling, the export packets are replicated to all collectors configured to receive them. If two collectors are configured to receive version 5 records, both collectors will receive records for a specified flow.

The following configuration example allows replication of export packets to two collectors.

```

forwarding-options {
  sampling {
    input {
      family inet {
        rate 1;
      }
    }
    output {
      cflowd 10.10.3.2 {
        port 2055;
        version 5;
        source-address 192.168.164.119;
      }
      cflowd 172.17.20.62 {
        port 2055;
        version 5;
        source-address 192.168.164.119;
      }
    }
  }
}

```

## Replicating Version 9 Flow Aggregation to Multiple Flow Servers

With this feature, you can configure up to eight flow servers to receive packets for a version 9 flow monitoring template. Once a flow server is configured to receive this data, it will also receive the following periodic version 9 flow monitoring updates:

- Options data
- Template definition

With Routing Engine-based sampling, if multiple collectors are configured with version 8 export format, all of them must use the same aggregation-type.

The option and template definition refresh period is configured on a per-template basis at the [edit services flow-monitoring] hierarchy level.

The following configuration example allows replication of version 9 export packets to two flow servers.

```
forwarding-options {
  sampling {
    input {
      family inet {
        rate 1;
      }
    }
    output {
      flow-server 10.10.3.2 {
        port 2055;
        version9 {
          template {
            ipv4;
          }
        }
      }
      flow-server 172.17.20.62 {
        port 2055;
        version9 {
          template {
            ipv4;
          }
        }
      }
      flow-inactive-timeout 30;
      flow-active-timeout 60;
      interface sp-4/0/0 {
        source-address 10.10.3.4;
      }
    }
  }
}
```

### Option: Configuring an Aggregate Export Timer

When you use flow monitoring version 8 records for active flow monitoring, you can configure an aggregate export timer. To configure this timer, include the `aggregate-export-interval` statement at the `[edit forwarding-options sampling output]` hierarchy level. The timer value has a default minimum setting of 90 seconds and a maximum value of 1800 seconds.

```
[edit]
forwarding-options {
  sampling {
    output {
      aggregate-export-interval duration;
    }
  }
}
```

### Option: Configuring Port Mirroring

---

You can copy packets and reroute them to another interface by using port mirroring. To send packet copies to an interface, include the `interface` statement at the `[edit forwarding-options port-mirroring family family-name output]` hierarchy level and specify the interface to receive the traffic.

You can even send port-mirrored traffic to a monitoring services or adaptive services interface. If you choose this option, accepted traffic is copied and the packet copies are sent to the services interface for flow processing.

To configure how often packets are copied from the monitored traffic, include the `rate` statement at the `[edit forwarding-options port-mirroring family family-name input]` hierarchy level. A rate of `1` port-mirrors every packet, while a rate of `10` port-mirrors every tenth packet.

```
[edit]
forwarding-options {
  port-mirroring {
    family (inet | inet6) {
      input {
        rate 1;
      }
      output {
        interface sp-2/0/0.0;
      }
    }
  }
}
```

## Option: Configuring Port Mirroring with Filter-Based Forwarding and a Monitoring Group

For active flow monitoring, you can load-balance traffic across multiple Monitoring Services PICs using the same method as passive flow monitoring. The only difference is that you do not configure the input interface with the `passive-monitor-mode` statement at the `[edit interfaces interface-name]` hierarchy level.

To load-balance traffic for active flow monitoring, port-mirror the incoming packets to a tunnel services interface. Redirect this copy of the traffic to a filter-based forwarding instance by applying a firewall filter to the tunnel services interface. Configure the instance to send the traffic to a group of monitoring services interfaces. Finally, use a monitoring group to send flow records from the monitoring services interfaces to a flow server.



**NOTE:** When you load-balance port-mirrored traffic across several Monitoring Services interfaces, there are some limitations:

- The original Monitoring Services PIC supports this method. You cannot use a Monitoring Services II PIC.
- You must use the suite of `show passive-monitoring` commands to monitor traffic. The `show services accounting` commands are not supported.
- Because load-balanced traffic is routed through the Tunnel Services PIC, the total throughput of the load-balanced traffic coming from the Monitoring Services PICs cannot exceed the bandwidth of the tunnel interface.

For detailed information on this method, see “Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding” on page 22.

## Option: Sending Traffic to Multiple Export Interfaces by Using Next-Hop Groups

To send port-mirrored traffic to multiple flow servers or packet analyzers, you can use the `next-hop-group` statement. The router can make up to 16 copies of traffic per group and send the traffic to the next-hop group members you configure. A maximum of 30 groups can be configured on a router at any given time. The port-mirrored traffic can be sent to any interface, except aggregated SONET/SDH, aggregated Ethernet, loopback (lo0), or administrative (fxp0) interfaces. To configure multiple port mirroring with next-hop groups, include the `next-hop-group` statement at the `[edit forwarding-options]` hierarchy level.

You must port-mirror the initial traffic to a tunnel interface so that it can be filtered and duplicated. Also, you need configure only the interface names for point-to-point interfaces, but you must configure the interface names and a next hop for multipoint interfaces (such as Ethernet).

```
[edit]
forwarding-options {
```

```

port-mirroring {
  family inet {
    input {
      rate 1;
    }
    output {
      interface vt-3/3/0.1;
      no-filter-check;
    }
  }
}
next-hop-group ftp-traffic {
  interface so-4/3/0.0;
  interface so-0/3/0.0;
}
next-hop-group http-traffic {
  interface ge-1/1/0.0 {
    next-hop 10.12.1.2;
  }
  interface ge-1/2/0.0 {
    next-hop 10.13.1.2;
  }
}
next-hop-group default-collect {
  interface so-7/0/0.0;
  interface so-7/0/1.0;
}
}

```



**NOTE:** Next-hop groups are supported on M Series routers only, except the M120 router and the M320 router.

---

### **Option: Using the Flow-Tap Application to Send Packets to a Mediation Device**

---

Dynamic flow capture passively monitors packet flows on the basis of dynamic filtering criteria, using Dynamic Tasking Control Protocol (DTCP) requests.

#### ***Option: Using the Flow-Tap Application to Send Packets to a Mediation Device***

Dynamic flow capture enables you to capture passively monitored packet flows on the basis of dynamic filtering criteria, using Dynamic Tasking Control Protocol (DTCP) requests. The flow-tap application extends the use of DTCP to intercept IPv4 packets in an active flow monitoring station and send a copy of packets that match filter criteria to one or more content destinations. Flow-tap data can be used for lawful intercept purposes and provides flexible trend analysis for detection of new security threats. The flow-tap application is supported on M Series and T Series routers, except M160 routers and TX Matrix platforms.



**NOTE:** For information about dynamic flow capture, see “Using a Dynamic Flow Capture Interface to Monitor Traffic On Demand” on page 34. For information about DTCP, see Internet draft draft-cavuto-dtcp-01.txt at <http://www.ietf.org/internet-drafts>.

For detailed information about the flow-tap application, see the following sections:

- Flow-Tap Architecture on page 84
- Configuring the Flow-Tap Interface on page 85
- Configuring Flow-Tap Security Properties on page 86
- Flow-Tap Application Restrictions on page 86
- Example: Flow-Tap Configuration on page 87

## Flow-Tap Architecture

The flow-tap architecture consists of one or more *mediation devices* that send requests to a Juniper Networks router to monitor incoming data. Any packets that match specific filter criteria are forwarded to a set of one or more *content destinations*:

- **Mediation device**—A client that monitors electronic data or voice transfer over the network. The mediation device sends filter requests to the Juniper Networks router using the DTCP. The clients are not identified for security reasons, but have permissions defined by a set of special login classes.
- **Monitoring platform**—A Juniper Networks M Series or T Series router containing one or more Adaptive Services (AS) PICs, which are configured to support the flow-tap application. The monitoring platform processes the requests from the mediation devices, applies the dynamic filters, monitors incoming data flows, and sends the matched packets to the appropriate content destinations.
- **Content destination**—Recipient of the matched packets from the monitoring platform. Typically the matched packets are sent using an IP Security (IPSec) tunnel from the monitoring platform to another router connected to the content destination. The content destination and the mediation device can be physically located on the same host.
- **Dynamic filters**—The Packet Forwarding Engine automatically generates a firewall filter that is applied to all IPv4 routing instances. Each term in the filter includes a **flow-tap** action that is similar to the existing **sample** or **port-mirroring** actions. As long as one of the filter terms matches an incoming packet, the router copies the packet and forwards it to the AS PIC that is configured for flow-tap service. The AS PIC runs the packet through the client filters and sends a copy to each matching content destination. For security, filters installed by one client are not visible to others and the CLI configuration does not reveal the identity of the monitored target.

Following is a sample filter configuration; note that it is dynamically generated by the router (no user configuration is required):

```
filter combined_LEA_filter {
  term LEA1_filter {
    from {
```

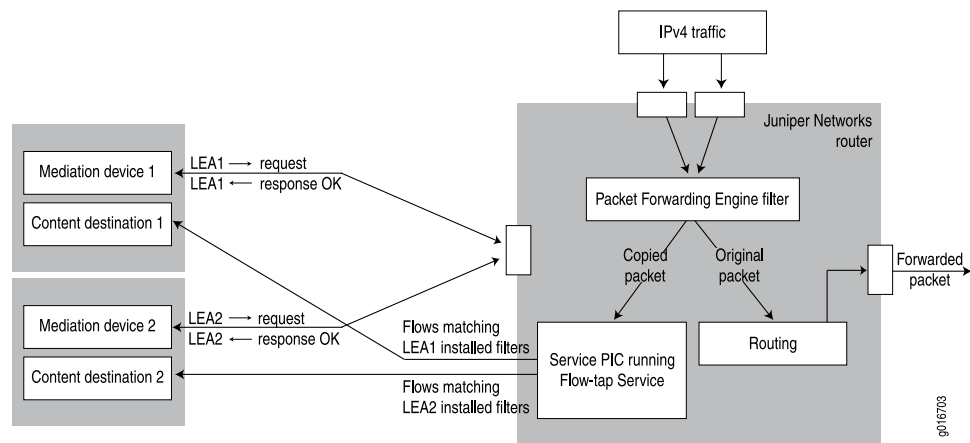
```

        source-address 1.2.3.4;
        destination-address 3.4.5.6;
    }
    then {
        flow-tap;
    }
}
term LEA2_filter {
    from {
        source-address 10.1.1.1;
        source-port 23;
    }
    then {
        flow-tap;
    }
}
}

```

Figure 5 on page 85 shows a sample topology that uses two mediation devices and two content destinations.

**Figure 5: Flow-Tap Topology Diagram**



## Configuring the Flow-Tap Interface

To configure an AS PIC interface for the flow-tap service, include the `interface` statement at the `[edit services flow-tap]` hierarchy level:

```
interface sp-fpc/pic/port.unit-number;
```

You can assign any AS PIC in the active monitoring station for flow-tap service, and use any logical unit on the PIC.



**NOTE:** You cannot configure dynamic flow capture and flow-tap features on the same router simultaneously.

You must also configure the logical interface at the `[edit interfaces]` hierarchy level:

```
interface sp-fpc/pic/port {
  unit logical-unit-number {
    family inet;
  }
}
```

## Configuring Flow-Tap Security Properties

You can add an extra level of security to DTCP transactions between the mediation device and the router by enabling DTCP sessions on top of the SSH layer. To configure, include the `flow-tap-dtcp` statement at the `[edit system services]` hierarchy level:

```
flow-tap-dtcp {
  ssh {
    connection-limit value;
    rate-limit value;
  }
}
```

To configure client permissions for viewing and modifying flow-tap configurations and for receiving tapped traffic, include the `permissions` statement at the `[edit system login class class-name]` hierarchy level:

```
permissions [ permissions ];
```

The permissions needed to use flow-tap features are as follows:

- `flow-tap`—Can view flow-tap configuration.
- `flow-tap-control`—Can modify flow-tap configuration.
- `flow-tap-operation`—Can tap flows.

You can also specify user permissions on a RADIUS server, for example:

```
Bob Auth-Type := Local, User-Password = "abc123"
Juniper-User-Permissions = "flow-tap-operation"
```

For details on `[edit system]` and RADIUS configuration, see the *JUNOS System Basics Configuration Guide*.

## Flow-Tap Application Restrictions

The following restrictions apply to flow-tap services:

- You cannot configure dynamic flow capture and flow-tap services on the same router simultaneously.
- When the dynamic flow capture process or an AS PIC configured for flow-tap processing restarts, all filters are deleted and the mediation devices are disconnected.
- Only the first fragment of an IPv4 fragmented packet stream is sent to the content destination.

- Port mirroring might not work in conjunction with flow-tap processing.
- If the flow-tap application is configured, you cannot configure the filter action then syslog for any firewall filter running on the same platform.

### **Example: Flow-Tap Configuration**

The following example shows all the parts of a complete flow-tap configuration.

```

services {
  flow-tap {
    interface sp-1/2/0.100;
  }
}
interfaces {
  sp-1/2/0 {
    unit 100 {
      family inet;
    }
  }
}
system {
  services {
    flow-tap-dtcp {
      ssh {
        connection-limit 5;
        rate-limit 5;
      }
    }
  }
  login {
    class ft-class {
      permissions flow-tap-operation;
    }
    user ft-user1 {
      class ft-class;
      authentication {
        encrypted-password "xxxx";
      }
    }
  }
}

```



## Chapter 5

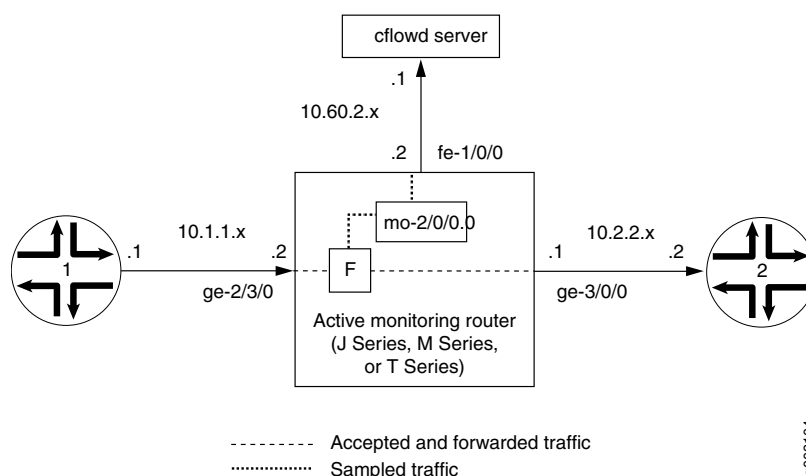
# Active Flow Monitoring Configuration Examples

This section contains configuration examples and commands you can issue to verify an active flow monitoring configuration:

- Example: Sampling Configuration on page 89
- Example: Sampling and Discard Accounting Configuration on page 93
- Example: Multiple Port Mirroring with Next-Hop Groups Configuration on page 97
- Example: Sampling Instance Configuration on page 100

## Example: Sampling Configuration

**Figure 6: Active Flow Monitoring—Sampling Configuration Topology Diagram**



In Figure 6 on page 89, traffic from Router 1 arrives on the monitoring router's Gigabit Ethernet **ge-2/3/0** interface. The exit interface on the monitoring router that leads to destination Router 2 is **ge-3/0/0**. In active flow monitoring, both the input interface and exit interface can be any interface type (such as SONET/SDH, Gigabit Ethernet, and so on). The export interface leading to the flow server is **fe-1/0/0**.

Configure a firewall filter to sample, count, and accept all traffic. Apply the filter to the input interface, and configure the exit interface (for traffic forwarding), the

adaptive services interface (for flow processing), and the export interface (for exporting flow records).

Configure sampling at the [edit forwarding-options] hierarchy level. Include the IP address and port of the flow server with the **flow-server** statement and specify the adaptive services interface to be used for flow record processing with the **interface** statement at the [edit forwarding-options sampling] hierarchy level.

```

Router 1 [edit]
interfaces {
  sp-2/0/0 { # This adaptive services interface creates the flow records.
    unit 0 {
      family inet {
        address 10.5.5.1/32 {
          destination 10.5.5.2;
        }
      }
    }
  }
  fe-1/0/0 { # This is the interface where records are sent to the flow server.
    unit 0 {
      family inet {
        address 10.60.2.2/30;
      }
    }
  }
  ge-2/3/0 { # This is the input interface where all traffic enters the router.
    unit 0 {
      family inet {
        filter {
          input catch_all; # This is where the firewall filter is applied.
        }
        address 10.1.1.1/20;
      }
    }
  }
  ge-3/0/0 { # This is the interface where the original traffic is forwarded.
    unit 0 {
      family inet {
        address 10.2.2.1/24;
      }
    }
  }
}
forwarding-options {
  sampling { # Traffic is sampled and sent to a flow server.
    input {
      rate 1; # Samples 1 out of x packets (here, a rate of 1 sample per packet).
    }
    family inet {
      output {
        flow-server 10.60.2.1 { # The IP address and port of the flow server.
          port 2055;
          version 5; # Records are sent to the flow server using version 5 format.
        }
      }
    }
  }
}

```



The active flow monitoring commands can be used with most active flow monitoring applications, including sampling, discard accounting, port mirroring, and multiple port mirroring. However, you can use the passive flow monitoring commands only with configurations that contain a monitoring group at the [edit forwarding-options monitoring] hierarchy level.

The following shows the output of the **show** commands used with the configuration example:

```

user@router> show services accounting errors
Service Accounting interface: sp-2/0/0, Local interface index: 542
Service name: (default sampling)
Error information
  Packets dropped (no memory): 0, Packets dropped (not IP): 0
  Packets dropped (not IPv4): 0, Packets dropped (header too small): 0
  Memory allocation failures: 0, Memory free failures: 0
  Memory free list failures: 0
  Memory overload: No, PPS overload: No, BPS overload: Yes

user@router> show services accounting flow-detail limit 10
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: (default sampling)

```

Protocol	Source Address	Source Port	Destination Address	Destination Port	Packet count	Byte count
udp(17)	10.1.1.2	53	10.0.0.1	53	4329	3386035
ip(0)	10.1.1.2	0	10.0.0.2	0	4785	3719654
ip(0)	10.1.1.2	0	10.0.1.2	0	4530	3518769
udp(17)	10.1.1.2	0	10.0.7.1	0	5011	3916767
tcp(6)	10.1.1.2	20	10.3.0.1	20	1	1494
tcp(6)	10.1.1.2	20	10.168.80.1	20	1	677
tcp(6)	10.1.1.2	20	10.69.192.1	20	1	446
tcp(6)	10.1.1.2	20	10.239.240.1	20	1	1426
tcp(6)	10.1.1.2	20	10.126.160.1	20	1	889
tcp(6)	10.1.1.2	20	10.71.224.1	20	1	1046

```

user@router> show services accounting memory
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: (default sampling)
Memory utilization
  Allocation count: 437340, Free count: 430681, Maximum allocated: 6782
  Allocations per second: 3366, Frees per second: 6412
  Total memory used (in bytes): 133416928, Total memory free (in bytes):
133961744

user@router> show services accounting packet-size-distribution
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: (default sampling)

```

Range start	Range end	Number of packets	Percentage packets
64	96	1705156	100

```

user@router> show services accounting status
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: (default sampling)
Interface state: Monitoring
Group index: 0
Export interval: 60 secs, Export format: cflowd v5
Protocol: IPv4, Engine type: 55, Engine ID: 5
Route record count: 13, IFL to SNMP index count: 30, AS count: 1
Time set: Yes, Configuration set: Yes
Route record set: Yes, IFL SNMP map set: Yes

```

```

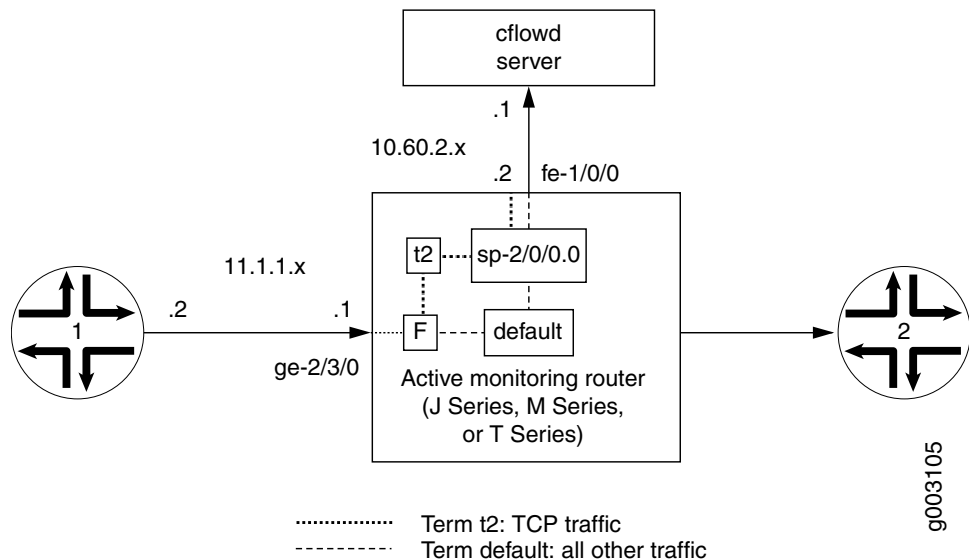
user@router> show services accounting usage
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: (default sampling)
CPU utilization
  Uptime: 4790345 milliseconds, Interrupt time: 1668537848 microseconds
  Load (5 second): 71%, Load (1 minute): 63%

```

## Example: Sampling and Discard Accounting Configuration

Discard accounting allows you to sample traffic, send it to a flow server for analysis, and discard all packets without forwarding them to their intended destination. Discard accounting is enabled with the discard accounting *group-name* statement in a firewall filter at the [edit firewall family inet filter *filter-name* term *term-name* then] hierarchy level. Then, the filter is applied to an interface with the filter statement at the [edit interfaces *interface-name* unit *unit-number* family inet] hierarchy level and processed with the output statement at the [edit forwarding-options accounting *group-name*] hierarchy level.

**Figure 7: Active Flow Monitoring—Sampling and Discard Accounting Topology Diagram**



In Figure 7 on page 93, traffic from Router 1 arrives on the monitoring router's Gigabit Ethernet **ge-2/3/0** interface. The export interface leading to the flow server is **fe-1/0/0** and there is no exit interface.

In this example, TCP traffic is sent to one accounting group and all other traffic is diverted to a second group. After being sampled and counted, the two types of traffic are acted upon by the sampling and accounting processes. These processes create flow records and send the records to the version 8 flow server for analysis. Because multiple types of traffic are sent to the same server, we recommend that you configure the **engine-id**, **engine-type**, and **source-address** statements manually in your accounting and sampling hierarchies. This way, you can differentiate between traffic types when they arrive at the flow server.

```

[edit]
interfaces {
  sp-2/0/0 { # This adaptive services interface creates the flow records.
    unit 0 {
      family inet {
        address 10.5.5.1/32 {
          destination 10.5.5.2;
        }
      }
    }
  }
  fe-1/0/0 { # This is the interface where records are sent to the flow server.
    unit 0 {
      family inet {
        address 10.60.2.2/30;
      }
    }
  }
  ge-2/3/0 { # This is the input interface where traffic enters the router.
    unit 0 {
      family inet {
        filter {
          input catch_all;
        }
        address 10.1.1.1/20;
      }
    }
  }
}
forwarding-options {
  sampling { # The router samples the traffic.
    input {
      rate 100; # One out of every 100 packets is sampled.
    }
    family inet {
      output { # The sampling process creates and exports flow records.
        flow-server 10.60.2.1 { # You can configure a variety of settings.
          port 2055;
          version 8;
          aggregation { # Aggregation is unique to flow version 8.
            protocol-port;
            source-destination-prefix;
          }
        }
        aggregate-export-interval 90;
        flow-inactive-timeout 60;
        flow-active-timeout 60;
        interface sp-2/0/0 { # This statement enables PIC-based sampling.
          engine-id 5; # Engine statements are dynamic, but can be configured.
          engine-type 55;
          source-address 10.60.2.2; # You must configure this statement.
        }
      }
    }
  }
  accounting counter1 { # This discard accounting process handles default traffic.

```

```

output { # This process creates and exports flow records.
    flow-inactive-timeout 65;
    flow-active-timeout 65;
    flow-server 10.60.2.1 { # You can configure a variety of settings.
        port 2055;
        version 8;
        aggregation { # Aggregation is unique to version 8.
            protocol-port;
            source-destination-prefix;
        }
    }
}

interface sp-2/0/0 { # This statement enables PIC-based discard accounting.
    engine-id 1; # Engine statements are dynamic, but can be configured.
    engine-type 11;
    source-address 10.60.2.3; # You must configure this statement.
}

}

accounting t2 { # The second discard accounting process handles the TCP traffic.
    output { # This process creates and exports flow records.
        aggregate-export-interval 90;
        flow-inactive-timeout 65;
        flow-active-timeout 65;
        flow-server 10.60.2.1 { # You can configure a variety of settings for the server.
            port 2055;
            version 8;
            aggregation { # Aggregation is unique to version 8.
                protocol-port;
                source-destination-prefix;
            }
        }
    }

    interface sp-2/0/0 { # This statement enables PIC-based discard accounting.
        engine-id 2; # Engine statements are dynamic, but can be configured.
        engine-type 22;
        source-address 10.60.2.4; # You must configure this statement.
    }
}

}

}

firewall {
    family inet {
        filter catch_all { # Apply the firewall filter on the input interface.
            term t2 { # This places TCP traffic into one group for sampling and
                from { # discard accounting.
                    protocol tcp;
                }
                then {
                    count c2; # The count action counts traffic as it enters the router.
                    sample; # The sample action sends the traffic to the sampling process.
                    discard accounting t2; # The discard accounting discards traffic.
                }
            }

            term default { # Performs sampling and discard accounting on all other traffic.
                then {
                    count counter; # The count action counts traffic as it enters the router.
                    sample; # The sample action sends the traffic to the sampling process.
                }
            }
        }
    }
}

```

```

        discard accounting counter1; # This activates discard accounting.
    }
}
}
}
}

```

## Verifying Your Work

To verify that your configuration is correct, use the following commands on the monitoring station that is configured for active flow monitoring:

- `show services accounting aggregation` (for version 8 flows only)
- `show services accounting errors`
- `show services accounting (flow | flow-detail)`
- `show services accounting memory`
- `show services accounting packet-size-distribution`
- `show services accounting status`
- `show services accounting usage`

The following shows the output of the `show` commands used with the configuration example:

```

user@router> show services accounting flow name t2
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: t2
  Flow information
    Flow packets: 56130820, Flow bytes: 3592372480
    Flow packets 10-second rate: 13024, Flow bytes 10-second rate: 833573
    Active flows: 600, Total flows: 600
    Flows exported: 28848, Flows packets exported: 960
    Flows inactive timed out: 0, Flows active timed out: 35400

user@router> show services accounting
Service Name:
  (default sampling)
  counter1
  t2

user@router> show services accounting aggregation protocol-port detail name t2
Service Accounting interface: sp-2/0/0, Local interface index: 468
Service name: t2

  Protocol: 6, Source port: 20, Destination port: 20
  Start time: 442794, End time: 6436260
  Flow count: 1, Packet count: 4294693925, Byte count: 4277471552

user@router> show services accounting aggregation source-destination-prefix name
t2 limit 10 order packets
Service Accounting interface: sp-2/0/0, Local interface index: 542
Service name: t2
  Source      Destination  Input SNMP  Output SNMP  Flow  Packet  Byte

```

Prefix	Prefix	Index	Index	count	count	count
10.1.1.2/20	10.225.0.1/0	24	26	0	13	9650
10.1.1.2/20	10.143.80.1/0	24	26	0	13	10061
10.1.1.2/20	10.59.176.1/0	24	26	0	13	10426
10.1.1.2/20	10.5.32.1/0	24	26	0	13	12225
10.1.1.2/20	10.36.16.1/0	24	26	0	13	9116
10.1.1.2/20	10.1.96.1/0	24	26	0	12	11050
10.1.1.2/20	10.14.48.1/0	24	26	0	13	10812
10.1.1.2/20	10.31.192.1/0	24	26	0	13	11473
10.1.1.2/20	10.129.144.1/0	24	26	0	13	7647
10.1.1.2/20	10.188.160.1/0	24	26	0	13	10056

```
user@router> show services accounting aggregation source-destination-prefix name
t2 extensive limit 3
Service Accounting interface: sp-2/0/0, Local interface index: 542
Service name: t2
```

```
Source address: 10.1.1.2, Source prefix length: 20
Destination address: 10.200.176.1, Destination prefix length: 0
Input SNMP interface index: 24, Output SNMP interface index: 26
Source-AS: 69, Destination-AS: 69
Start time: Fri Feb 21 14:16:57 2003, End time: Fri Feb 21 14:22:50 2003
Flow count: 0, Packet count: 6, Byte count: 5340
```

```
Source address: 10.1.1.2, Source prefix length: 20
Destination address: 10.243.160.1, Destination prefix length: 0
Input SNMP interface index: 24, Output SNMP interface index: 26
Source-AS: 69, Destination-AS: 69
Start time: Fri Feb 21 14:16:57 2003, End time: Fri Feb 21 14:22:50 2003
Flow count: 0, Packet count: 6, Byte count: 5490
```

```
Source address: 10.1.1.2, Source prefix length: 20
Destination address: 10.162.160.1, Destination prefix length: 0
Input SNMP interface index: 24, Output SNMP interface index: 26
Source-AS: 69, Destination-AS: 69
Start time: Fri Feb 21 14:16:57 2003, End time: Fri Feb 21 14:22:50 2003
Flow count: 0, Packet count: 6, Byte count: 4079
```

## Example: Multiple Port Mirroring with Next-Hop Groups Configuration

When you need to analyze traffic containing more than one packet type, or you wish to perform multiple types of analysis on a single type of traffic, you can implement multiple port mirroring and next-hop groups. You can make up to 16 copies of traffic per group and send the traffic to next-hop group members. A maximum of 30 groups can be configured on a router at any given time. The port-mirrored traffic can be sent to any interface, except aggregated SONET/SDH, aggregated Ethernet, loopback (lo0), or administrative (fxp0) interfaces. To send port-mirrored traffic to multiple flow servers or packet analyzers, you can use the `next-hop-group` statement at the `[edit forwarding-options]` hierarchy level.

**Figure 8: Active Flow Monitoring—Multiple Port Mirroring with Next-Hop Groups Topology Diagram**

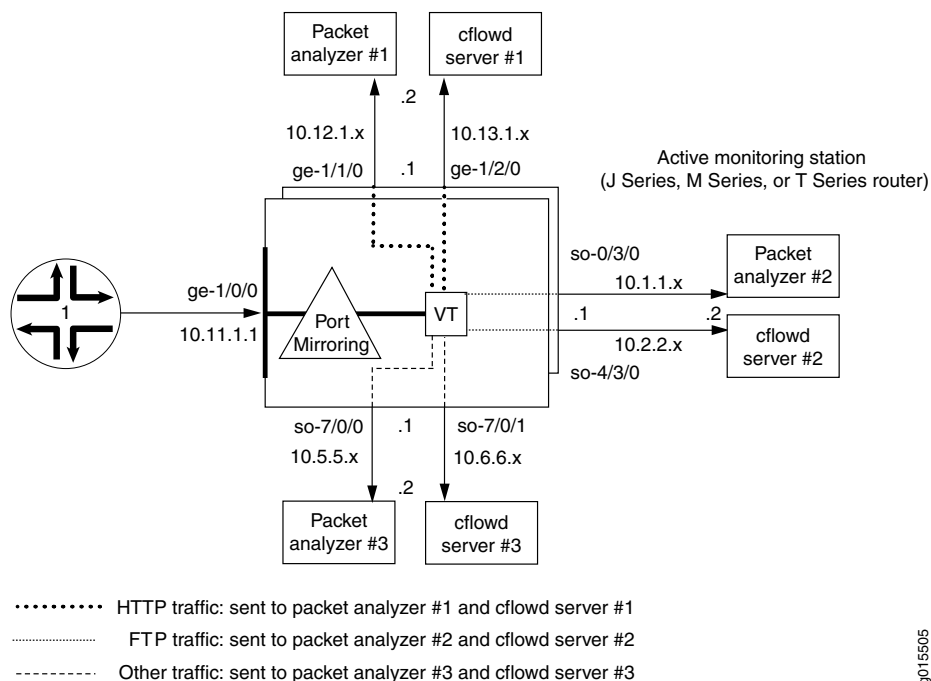


Figure 8 on page 98 shows an example of how to configure multiple port mirroring with next-hop groups. All traffic enters the monitoring router at interface **ge-1/0/0**. A firewall filter counts and port-mirrors all incoming packets to a Tunnel Services PIC. A second filter is applied to the tunnel interface and splits the traffic into three categories: HTTP traffic, FTP traffic, and all other traffic. The three types of traffic are assigned to three separate next-hop groups. Each next-hop group contains a unique pair of exit interfaces that lead to different groups of packet analyzers and flow servers.

```
[edit]
interfaces {
  ge-1/0/0 { # This is the input interface where packets enter the router.
    unit 0 {
      family inet {
        filter {
          input mirror_pkts; # Here is where you apply the first filter.
        }
        address 10.11.1.1/24;
      }
    }
  }
  ge-1/1/0 { # This is an exit interface for HTTP packets.
    unit 0 {
      family inet {
        address 10.12.1.1/24;
      }
    }
  }
}
```

```

ge-1/2/0 { # This is an exit interface for HTTP packets.
    unit 0 {
        family inet {
            address 10.13.1.1/24;
        }
    }
}
so-0/3/0 { # This is an exit interface for FTP packets.
    unit 0 {
        family inet {
            address 10.1.1.1/30;
        }
    }
}
so-4/3/0 { # This is an exit interface for FTP packets.
    unit 0 {
        family inet {
            address 10.2.2.1/30;
        }
    }
}
so-7/0/0 { # This is an exit interface for all remaining packets.
    unit 0 {
        family inet {
            address 10.5.5.1/30;
        }
    }
}
so-7/0/1 { # This is an exit interface for all remaining packets.
    unit 0 {
        family inet {
            address 10.6.6.1/30;
        }
    }
}
vt-3/3/0 { # The tunnel interface is where you send the port-mirrored traffic.
    unit 0 {
        family inet;
    }
    unit 1 {
        family inet {
            filter {
                input collect_pkts; # This is where you apply the second firewall filter.
            }
        }
    }
}
forwarding-options {
    port-mirroring { # This is required when you configure next-hop groups.
        family inet {
            input {
                rate 1; # This port-mirrors all packets (one copy for every packet received).
            }
            output { # Sends traffic to a tunnel interface to enable multiport mirroring.
                interface vt-3/3/0.1;
            }
        }
    }
}

```

```

        no-filter-check;
    }
}
next-hop-group ftp-traffic { # Point-to-point interfaces require you to specify the
    interface so-4/3/0.0; # interface name.
    interface so-0/3/0.0;
}
next-hop-group http-traffic { # Configure a next hop for all multipoint interfaces.
    interface ge-1/1/0.0 {
        next-hop 10.12.1.2;
    }
    interface ge-1/2/0.0 {
        next-hop 10.13.1.2;
    }
}
next-hop-group default-collect {
    interface so-7/0/0.0;
    interface so-7/0/1.0;
}
}
firewall {
    family inet {
        filter mirror_pkts { # Apply this filter to the input interface.
            term catch_all {
                then {
                    count input_mirror_pkts;
                    port-mirror; # This action sends traffic to be copied and port-mirrored.
                }
            }
        }
        filter collect_pkts { # Apply this filter to the tunnel interface.
            term ftp-term { # This term sends FTP traffic to an FTP next-hop group.
                from {
                    protocol ftp;
                }
                then next-hop-group ftp-traffic;
            }
            term http-term { # This term sends HTTP traffic to an HTTP next-hop group.
                from {
                    protocol http;
                }
                then next-hop-group http-traffic;
            }
            term default { # This sends all remaining traffic to a final next-hop group.
                then next-hop-group default-collectors;
            }
        }
    }
}
}

```

## Example: Sampling Instance Configuration

---

You can configure active sampling using a sampling instance and associate that sampling instance to a particular Packet Forwarding Engine. In addition, you can

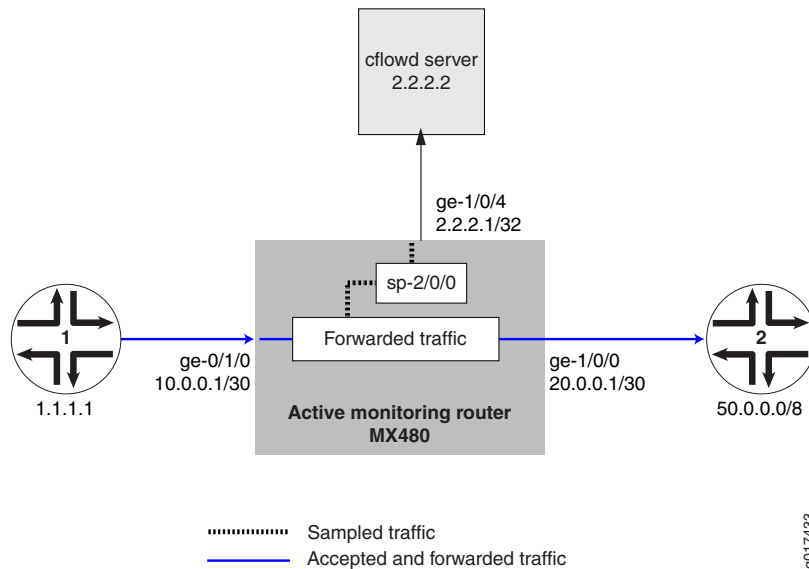
define multiple sampling instances associated with multiple destinations (as many as the number of Packet Forwarding Engines in the chassis), with multiple protocol families per each sampling instance destination. This configuration example shows one sampling instance configured and associated with one Packet Forward Engine. Sampling instance configuration is supported on MX Series, M120, M320, and T Series routers.

- Example Network Details on page 101
- Example Router Configuration on page 102
- Configuration Commands Used for the Configuration Example on page 103
- Verifying Your Work on page 104

### Example Network Details

The following example shows the configuration of one sampling instance on an MX480 router running JUNOS Release 9.6.

**Figure 9: Active Flow Monitoring—Sampling Instance Configuration Topology Diagram**



In Figure 9 on page 101, packets from Router 1 arrive on the monitoring router's Gigabit Ethernet `ge-0/1/0` interface, the packets are sampled by the services interface `sp-2/0/0` and sent to the cflowd server by the export interface `ge-1/0/4`. Normal traffic flow from `ge-0/1/0` to `ge-1/0/0` and on to Router 2 continues undisturbed during the sampling process. In active flow monitoring, both the input interface and exit interface can be any interface type (such as SONET/SDH, Gigabit Ethernet, and so on).

The sampling configuration for this example includes the following:

- A sampling instance configured to collect sampling data at the [edit forwarding-options] hierarchy level. The `flow-server` statement includes the IP address, port, and template of the flow server. The `interface` statement includes the services interface `sp-2/0/0` for flow record processing.

- The sampling instance associated with the Packet Forwarding Engine using the `sampling-instance` statement at the `[edit chassis fpc slot]` hierarchy level.
- Sampling activated on the input interface `ge-0/1/0` using the `sampling` statement at the `[edit interfaces interface-name unit unit-number family family]` hierarchy level.

In this example, the `ping` command is issued on Router 1 to Router 2 via the MX480 router to generate traffic. After the packets are generated, `show` commands are issued to verify that the sampling configuration is working as expected.

## Example Router Configuration

The following output shows the configuration of an MX480 router with a sampling instance.

```
user@MX480-router> show configuration
[...Output Truncated...]
}
chassis {
  fpc 0 { # The fpc number is associated with the interface on which sampling
is enabled, ge-0/1/0 in this example.
    sampling-instance s0;
  }
}
interfaces {
  ge-0/1/0 { # This interfaces has sampling activated.
    unit 0 {
      family inet {
        sampling { # Here sampling is activated.
          input;
        }
        address 10.0.0.1/30;
      }
    }
  }
  ge-1/0/0 { # The interface on which packets are exiting the router.
    unit 0 {
      family inet {
        address 20.0.0.1/30;
      }
    }
  }
  ge-1/0/4 { # The interface connected to the cflowd server.
    unit 0 {
      family inet {
        address 2.2.2.1/32;
      }
    }
  }
  sp-2/0/0 { # The service interface that samples the packets.
    unit 0 {
      family inet;
    }
  }
}
forwarding-options {
  sampling {
    instance {
```

```

s0 {
    input {
        rate 1;
        run-length 0;
    }
    family inet {
        output {
            flow-server 2.2.2.2 { # The address of the external server.

                port 2055;
                version9 {
                    template {
                        v4;
                    }
                }
            }
        }
        interface sp-2/0/0 {
            source-address 1.1.1.1; # Source address of the sampled
packets
        }
    }
}

routing-options {
    static {
        route 50.0.0.0/8 next-hop 20.0.0.2;
    }
}

services {
    flow-monitoring {
        version9 {
            template v4 {
                flow-active-timeout 30;
                flow-inactive-timeout 30;
                ipv4-template;
            }
        }
    }
}

```

### Configuration Commands Used for the Configuration Example

The following **set** commands are used for the configuration of the sampling instance in this example. Replace the values in these commands with values relevant to your own network.

- **set chassis fpc 0 sampling-instance s0**
- **set interfaces ge-0/1/0 unit 0 family inet sampling input**
- **set interfaces ge-0/1/0 unit 0 family inet address**
- **set interfaces ge-1/0/0 unit 0 family inet address**
- **set interfaces sp-2/0/0 unit 0 family inet**
- **set forwarding-options sampling instance s0 input rate 1**

- set forwarding-options sampling instance s0 input run-length 0
- set forwarding-options sampling instance s0 family inet output flow-server 2.2.2.2 port 2055
- set forwarding-options sampling instance s0 family inet output flow-server 2.2.2.2 version9 template v4;
- set forwarding-options sampling instance s0 family inet output interface sp-2/0/0 source-address 1.1.1.1
- set routing-options static route 50.0.0.0/8 next-hop 20.0.0.2
- set services flow-monitoring version9 template v4 flow-active-timeout 30
- set services flow-monitoring version9 template v4 flow-inactive-timeout 30
- set services flow-monitoring version9 template v4 ipv4-template

## Verifying Your Work

To verify that your configuration is working as expected, use the following commands on the router that is configured with the sampling instance:

- show services accounting aggregation template template-name *template-name*
- show services accounting flow

The following shows the output of the **show** commands issued on the MX480 router used in this configuration example:

```
user@MX480-router> show services accounting aggregation template template-name v4
```

Source Address	Destination Address	Src Port/ ICMP Type	Dst Port/ ICMP Code	Proto	TOS	Packet Count
10.0.0.6	50.0.0.3	100	1000	17	8	14
10.0.0.5	50.0.0.2	100	1000	17	8	15
10.0.0.3	50.0.0.3	100	1000	17	8	15
10.0.0.2	50.0.0.3	100	1000	17	8	15
10.0.0.4	50.0.0.2	100	1000	17	8	15
10.0.0.6	50.0.0.2	100	1000	17	8	15
10.0.0.4	50.0.0.3	100	1000	17	8	15
10.0.0.2	50.0.0.2	100	1000	17	8	16
10.0.0.3	50.0.0.2	100	1000	17	8	15
10.0.0.5	50.0.0.3	100	1000	17	8	15

```
user@MX480-router> show services accounting aggregation template template-name v4
```

Source Address	Destination Address	Src Port/ ICMP Type	Dst Port/ ICMP Code	Proto	TOS	Packet Count
10.0.0.6	50.0.0.3	100	1000	17	8	16
10.0.0.5	50.0.0.2	100	1000	17	8	17
10.0.0.3	50.0.0.3	100	1000	17	8	16
10.0.0.2	50.0.0.3	100	1000	17	8	16
10.0.0.4	50.0.0.2	100	1000	17	8	17

10.0.0.6	50.0.0.2	100	1000	17	8	17
10.0.0.4	50.0.0.3	100	1000	17	8	16
10.0.0.2	50.0.0.2	100	1000	17	8	17
10.0.0.3	50.0.0.2	100	1000	17	8	17
10.0.0.5	50.0.0.3	100	1000	17	8	16

```
user@MX480-router> show services accounting flow
```

```
Flow information
```

```
Interface name: sp-2/0/0, Local interface index: 152
```

```
Flow packets: 884, Flow bytes: 56576
```

```
Flow packets 10-second rate: 0, Flow bytes 10-second rate: 628
```

```
Active flows: 10, Total flows: 35
```

```
Flows exported: 75, Flows packets exported: 14
```

```
Flows inactive timed out: 25, Flows active timed out: 75
```

```
user@MX480-router> show services accounting flow
```

```
Flow information
```

```
Interface name: sp-2/0/0, Local interface index: 152
```

```
Flow packets: 898, Flow bytes: 57472
```

```
Flow packets 10-second rate: 0, Flow bytes 10-second rate: 628
```

```
Active flows: 10, Total flows: 35
```

```
Flows exported: 75, Flows packets exported: 14
```

```
Flows inactive timed out: 25, Flows active timed out: 75
```



## Chapter 6

# Formats for Flow Monitoring Output

- Flow Monitoring Output Formats on page 107
- Version 5 Formats and Fields on page 108
- Version 8 Formats and Fields on page 111
- Version 9 Formats and Fields on page 117
- More Information About Passive and Active Flow Monitoring on page 123

### Flow Monitoring Output Formats

---

When you implement passive flow monitoring and active flow monitoring, you should be familiar with flow monitoring formats and fields. Version 5 and version 8 export data into specified fields. Version 9 exports data into templates.

The flow monitoring station monitors the traffic flow and exports the data in flow format to an external server. The JUNOS Software collects information about the following fields:

- Source and destination IP address
- Total number of bytes and packets sent
- Start and end times of the data flow
- Source and destination port numbers
- TCP flags
- IP protocol and IP type of service
- Originating AS of source and destination address
- Source and destination address prefix mask lengths
- Next-hop router's IP address
- MPLS label (version 9 only)
- ICMP (version 9 only)

Detailed descriptions of the formats are available as follows:

- Version 5 Formats and Fields on page 108
- Version 8 Formats and Fields on page 111
- Version 9 Formats and Fields on page 117

## Version 5 Formats and Fields

A detailed explanation of version 5 packet formats and fields is shown in the following figures and tables:

- Figure 10 on page 108
- Table 17 on page 108
- Figure 11 on page 109
- Table 18 on page 109

**Figure 10: Version 5 Packet Header Format**

Byte 3	Byte 2	Byte 1	Byte 0
Version		Count	
sysUptime			
UNIX seconds			
UNIX nanoseconds			
Flow sequence number			
Engine type	Engine ID	Reserved	

0003132

g003132

**Table 17: Export Version 5 Packet Header Fields**

Field	Description	Comments
Version	5	–
Count	The number of records in the Protocol Data Unit (PDU) or packet	–
sysUptime	Current time elapsed, in milliseconds, since the router started	–
UNIX seconds	Current seconds since 0000 UTC 1970	NTP synchronized time; the clock on each services PIC is autonomous (200–400 msec jitter) across PICs in a chassis
UNIX nanoseconds	Residual nanoseconds since 0000 UTC 1970	See Comments above for UNIX seconds
Flow sequence number	Sequence number of total flows received	–
Engine type	User-configured 8-bit value	Also known as VIP type on other vendors' equipment
Engine ID	User-configured 8-bit value	–

**Figure 11: Version 5 Flow-Export Flow Header Format**

Byte 3	Byte 2	Byte 1	Byte 0
Source IP address			
Destination IP address			
Next-hop IP address			
Input ifIndex		Output ifIndex	
Packets			
Bytes			
Start time of flow			
End time of flow			
Source port		Destination port	
Padding	TCP flags	IP protocol	TOS
Source AS		Destination AS	
Source mask length	Dest. mask length	Padding	

g003133

**Table 18: Export Version 5 Flow-Export Flow Header Fields**

Field	Description	Comments
Source IP address	Source IP address of the flow	–
Destination IP address	Destination IP address of the flow	–
Next-hop IP address	IP address of the router where flows are forwarded	–
Input ifIndex	SNMP index value for the input interface where the router receives flows	JUNOS Release 5.7 and later—Dynamically inserted, but overridden by manual configuration  JUNOS Release 5.5—Manually set  JUNOS Release 5.4—Set to zero
Output ifIndex	SNMP index value for the output interface where the router forwards flows	JUNOS Release 5.7 and later—Dynamically inserted, but overridden by manual configuration  JUNOS Release 5.5—Manually set  JUNOS Release 5.4—Set to zero
Packets	Total number of packets received in a flow	–
Bytes	Total number of bytes received in a flow	–
Start time of flow	System up time, in seconds, at the start of the flow	System up time for the services PIC accepting flows

**Table 18: Export Version 5 Flow-Export Flow Header Fields** (continued)

Field	Description	Comments
End time of flow	System up time, in seconds, at the end of the flow	System up time for the services PIC accepting flows
Source port	Source application port	–
Destination port	Destination application port	The ICMP type is placed in the high-order byte and the ICMP type code is placed in the low-order byte of this field
TCP flags	TCP flags set in the flow	–
IP protocol	IP protocol number	–
TOS	IP type of service	–
Source AS	AS number of the source address	JUNOS Release 5.7 and later—Dynamically inserted if AS information is available
Destination AS	AS number of the destination address	JUNOS Release 5.7 and later—Dynamically inserted if AS information is available
Source mask length	Source address network mask length	–
Dest. mask length	Destination address network mask length	–
Padding	Bytes available to ensure a minimum packet length	–

Useful formulas for flow monitoring are:

- start flow timestamp absolute =  $unixTime \times 1000 - (sysUptime - \text{start flow timestamp})$
- end flow timestamp absolute =  $unixTime \times 1000 - (sysUptime - \text{end flow timestamp})$



**NOTE:** In the 2-byte destination port field of the export version 5 flow-export flow format, the following information can be derived:

- High-order byte—ICMP type
- Low-order byte—ICMP type code

For example, if the ICMP type is 3 (00000011 in binary) and the ICMP type code is network unreachable (Type Code 0, or 00000000 in binary), the resulting destination port field value is 00000011 00000000 (768 in decimal).

For more information on ICMP type and type code, see RFC 792 at <http://www.ietf.org>.

## Version 8 Formats and Fields

A detailed explanation of version 8 packet formats and fields is shown as follows:

- Figure 12 on page 111
- Table 19 on page 112
- Figure 13 on page 112
- Table 20 on page 112
- Figure 14 on page 113
- Table 21 on page 113
- Figure 15 on page 114
- Table 22 on page 114
- Figure 16 on page 115
- Table 23 on page 115
- Figure 17 on page 116
- Table 24 on page 116

**Figure 12: Version 8 Template Flow Format**

Byte 3	Byte 2	Byte 1	Byte 0
Version		Count	
sysUptime			
UNIX seconds			
UNIX nanoseconds			
Flow Sequence Number			
Engine type	Engine ID	Aggregation method	Aggregation version
Reserved			

9003076

Byte 3	Byte 2	Byte 1	Byte 0
Version		Count	
sysUptime			
UNIX seconds			
UNIX nanoseconds			
Flow Sequence Number			
Engine type	Engine ID	Aggregation method	Aggregation version
Reserved			

9003076

**Table 19: Version 8 Flow Template Fields**

Field	Description
Version	8
Count	The number of records in the protocol data unit (PDU) or packet
sysUptime	Current time elapsed, in milliseconds, since the router started
UNIX seconds	Current seconds since 0000 UTC 1970
UNIX nanoseconds	Residual nanoseconds since 0000 UTC 1970
Flow sequence number	Sequence counter of total flows received
Engine type	Type of flow switching engine
Engine ID	ID number of the flow switching engine
Aggregation method	Aggregation method used
Aggregation version	Version of the aggregation export
Reserved	Empty field reserved for future usage

**Figure 13: Version 8 AS Aggregation Flow Entry Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flows			
Packets			
Bytes			
Start Time of Flow			
End Time of Flow			
Source AS		Destination AS	
Input interface		Output interface	

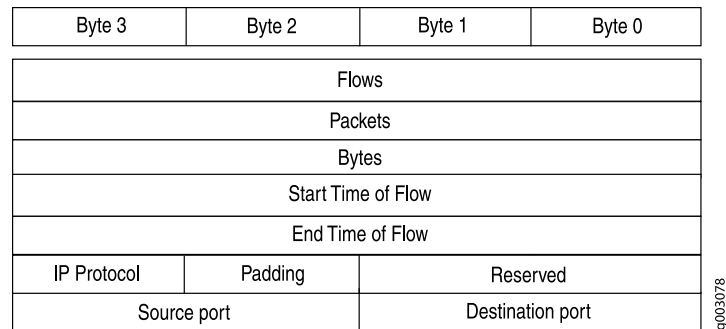
9003077

**Table 20: Version 8 AS Aggregation Flow Entry Fields**

Field	Description
Flows	Total number of flows
Packets	Total number of packets received in a flow
Bytes	Total number of bytes received in a flow
Start time of flow	System up time, in seconds, at the start of the flow

**Table 20: Version 8 AS Aggregation Flow Entry Fields** *(continued)*

Field	Description
End time of flow	System up time, in seconds, at the end of the flow
Source AS	AS number of the source address
Destination AS	AS number of the destination address
Input interface	SNMP index value for the input interface where the router receives flows
Output interface	SNMP index value for the output interface where the router forwards flows

**Figure 14: Version 8 Protocol/Port Aggregation Flow Entry Format****Table 21: Version 8 Protocol/Port Aggregation Flow Entry Fields**

Field	Description
Flows	Total number of flows
Packets	Total number of packets received in a flow
Bytes	Total number of bytes received in a flow
Start time of flow	System up time, in seconds, at the start of the flow
End time of flow	System up time, in seconds, at the end of the flow
IP protocol	IP protocol number
Padding	Bytes available to ensure a minimum packet length
Reserved	Empty field reserved for future usage
Source port	Source application port
Destination port	Destination application port

**Figure 15: Version 8 Prefix Aggregation Flow Entry Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flows			
Packets			
Bytes			
Start Time of Flow			
End Time of Flow			
Source prefix			
Destination prefix			
Source Mask Length	Dest. Mask Length	Reserved	
Source AS		Destination AS	
Input interface		Output interface	

9003079

Byte 3	Byte 2	Byte 1	Byte 0
Flows			
Packets			
Bytes			
Start Time of Flow			
End Time of Flow			
Source prefix			
Destination prefix			
Source Mask Length	Dest. Mask Length	Reserved	
Source AS		Destination AS	
Input interface		Output interface	

9003079

**Table 22: Version 8 Prefix Aggregation Flow Entry Fields**

Field	Description
Flows	Total number of flows
Packets	Total number of packets received in a flow
Bytes	Total number of bytes received in a flow
Start time of flow	System up time, in seconds, at the start of the flow
End time of flow	System up time, in seconds, at the end of the flow
Source prefix	Source IP address prefix
Destination prefix	Destination IP address prefix
Source mask length	Source address network mask length

**Table 22: Version 8 Prefix Aggregation Flow Entry Fields** (*continued*)

Field	Description
Dest. mask length	Destination address network mask length
Reserved	Empty field reserved for future usage
Source AS	AS number of the source address
Destination AS	AS number of the destination address
Input interface	SNMP index value for the input interface where the router receives flows
Output interface	SNMP index value for the output interface where the router forwards flows

**Figure 16: Version 8 Source Prefix Aggregation Flow Entry Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flows			
Packets			
Bytes			
Start Time of Flow			
End Time of Flow			
Source prefix			
Source Mask Length	Padding	Source AS	
Input interface		Reserved	

08030005

**Table 23: Version 8 Source Prefix Aggregation Flow Entry Fields**

Field	Description
Flows	Total number of flows
Packets	Total number of packets received in a flow
Bytes	Total number of bytes received in a flow
Start time of flow	System up time, in seconds, at the start of the flow
End time of flow	System up time, in seconds, at the end of the flow
Source prefix	Source IP address prefix
Source mask length	Source address network mask length
Padding	Bytes available to ensure a minimum packet length
Source AS	AS number of the source address

**Table 23: Version 8 Source Prefix Aggregation Flow Entry Fields** *(continued)*

Field	Description
Input interface	SNMP index value for the input interface where the router receives flows
Reserved	Empty field reserved for future usage

**Figure 17: Version 8 Destination Prefix Aggregation Flow Entry Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flows			
Packets			
Bytes			
Start Time of Flow			
End Time of Flow			
Destination prefix			
Dest. Mask Length	Padding	Destination AS	
Output interface		Reserved	

1803081

**Table 24: Version 8 Destination Prefix Aggregation Flow Entry Fields**

Field	Description
Flows	Total number of flows
Packets	Total number of packets received in a flow
Bytes	Total number of bytes received in a flow
Start time of flow	System up time, in seconds, at the start of the flow
End time of flow	System up time, in seconds, at the end of the flow
Destination prefix	Destination IP address prefix
Dest. mask length	Destination address network mask length
Padding	Bytes available to ensure a minimum packet length
Destination AS	AS number of the destination address
Output interface	SNMP index value for the output interface where the router forwards flows
Reserved	Empty field reserved for future usage

For more information about version 5 and version 8 packet formats and fields, see <http://www.caida.org>.

## Version 9 Formats and Fields

A detailed explanation of active flow monitoring version 9 packet formats and fields is shown as follows:

- Table 25 on page 117
- Figure 18 on page 119
- Table 26 on page 119
- Figure 20 on page 121
- Table 26 on page 119
- Figure 21 on page 122
- Table 30 on page 122
- Figure 22 on page 123
- Table 31 on page 123

The JUNOS Software supports the following version 9 template formats:

**Table 25: Flow Monitoring Version 9 Template Formats**

Template	Fields
IPv4	<p>Flow selectors:</p> <ul style="list-style-type: none"> <li>■ Source and destination IP address</li> <li>■ Source and destination address prefix mask lengths</li> <li>■ Source and destination port numbers</li> <li>■ IP protocol and IP type of service</li> <li>■ ICMP type</li> </ul> <p>Flow nonselectors:</p> <ul style="list-style-type: none"> <li>■ TCP flags</li> <li>■ Input and output SNMP</li> <li>■ Input bytes</li> <li>■ Input packets</li> <li>■ Start time</li> <li>■ End time</li> </ul>

**Table 25: Flow Monitoring Version 9 Template Formats** *(continued)*

Template	Fields
MPLS	<p>Flow selectors:</p> <ul style="list-style-type: none"> <li>■ MPLS label 1</li> <li>■ MPLS label 2</li> <li>■ MPLS label 3</li> </ul> <p>Flow nonselectors:</p> <ul style="list-style-type: none"> <li>■ Input and output SNMP</li> <li>■ Input bytes</li> <li>■ Input packets</li> <li>■ Start time</li> <li>■ End time</li> </ul>
MPLS_IPv4	<p>Flow selectors:</p> <ul style="list-style-type: none"> <li>■ MPLS label 1</li> <li>■ MPLS label 2</li> <li>■ MPLS label 3</li> </ul> <p>Flow nonselectors:</p> <ul style="list-style-type: none"> <li>■ Input and output SNMP</li> <li>■ Input bytes</li> <li>■ Input packets</li> <li>■ Start time</li> <li>■ End time</li> </ul>
IPv6	<p>Flow selectors:</p> <ul style="list-style-type: none"> <li>■ IP protocol and IP type of service</li> <li>■ Source and destination port numbers</li> <li>■ Input SNMP</li> <li>■ Source and destination IPv6 address</li> <li>■ ICMP type</li> </ul> <p>Flow nonselectors:</p> <ul style="list-style-type: none"> <li>■ Input bytes</li> <li>■ Input packets</li> <li>■ TCP flags</li> <li>■ Output SNMP</li> <li>■ Source and destination autonomous system</li> <li>■ Last and first switched</li> <li>■ IPv6 source and destination mask</li> <li>■ IP protocol version</li> <li>■ IPv6 next hop</li> </ul>

**Figure 18: Version 9 Flow Header Format**

Byte 3	Byte 2	Byte 1	Byte 0
Version		Count	
sysUptime			
UNIX seconds			
Flow Sequence Number			
Source ID			

9016785

**Table 26: Version 9 Flow Header Fields**

Field	Description
Version	9
Count	Total number of records in the protocol data unit (PDU) or packet. This number includes all of the options FlowSet records, template FlowSet records, and data FlowSet records.
sysUptime	Current time elapsed, in milliseconds, since the router started.
UNIX seconds	Current seconds since 0000 UTC 1970.
Flow sequence number	Sequence counter of total flows received.
Source ID	32-bit value that identifies the data exporter. Version 9 uses the integrated field diagnostics (IFD) SNMP index of the PIC or device that is exporting the data flow. This field is equivalent to engine type and engine ID fields found in versions 5 and 8.

**Figure 19: Version 9 Template FlowSet Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flowset ID = 0		Length	
Template ID 256		Field Count	
Field Type 1		Field Length 1	
Field Type 2		Field Length 2	
...		...	
Field Type N		Field Type N	
Template ID 257		Field Count	
Field Type 1		Field Length 1	

9016786

**Table 27: Version 9 Template FlowSet Fields**

Field	Description
FlowSet ID	FlowSet type. FlowSet ID 0 is reserved for the Template FlowSet.
Length	FlowSet length. Individual template FlowSets might contain multiple template records, which means that the length of template FlowSets varies.
Template ID	Unique template ID assigned to each newly generated template. Templates numbered 256 and higher define data formats. Templates numbered 0 through 255 define FlowSet IDs.
Field Count	Fields in the template record. This field allows the collector to determine the end of the current template record and the start of the next.
Field Type	Field type. These are defined in Table 28 on page 120.
Field Length	Length, in bytes, of the corresponding field type.

**Table 28: Field Type Definitions Supported in the JUNOS Software**

Field Type	Description
1	IN_BYTES: The number of bytes associated with an IP flow. By default, the length is 4 bytes.
2	IN_PKTS: The number of packets associated with an IP flow. By default, the length is 4 packets.
4	PROTOCOL: The IP protocol byte.
5	TOS: The type of service byte setting of an incoming packet.
6	TCP_FLAGS: The cumulative TCP flags associated with a flow.
7	L4_SRC_PORT: The TCP/UDP source port.
8	IPv4_SRC_ADDR: The IPv4 source address.
9	SRC_MASK: The number of contiguous bits in the source subnet mask.
10	INPUT_SNMP: The IFD SNMP input interface index. By default, the length is 2.
11	L4_DST_PORT: The TCP/UDP destination port number.
12	IPv4_DST_ADDR: The IPv4 destination address.
13	DST_MASK: The number of contiguous bits in the destination subnet mask.
14	OUTPUT_SNMP: The IFD SNMP output interface index. By default, the length is 2.

**Table 28: Field Type Definitions Supported in the JUNOS Software** *(continued)*

Field Type	Description
16	SRC_AS: The source autonomous system number. This is always set to zero.
17	DST_AS: The destination autonomous system number. This is always set to zero.
21	LAST_SWITCHED: The uptime of the device (in milliseconds) at which the last packet of the flow was switched.
22	FIRST_SWITCHED: The uptime of the device (in milliseconds) at which the first packet of the flow was switched.
29	IPV6_SRC_MASK: The length of the IPv6 source mask in contiguous bits.
30	IPV6_DST_MASK: The length of the IPv6 destination mask in contiguous bits.
32	ICMP_TYPE: The ICMP type.
34	SAMPLING_INTERVAL: The rate at which packets are sampled. As an example, a rate of 100 means that one packet is sampled for every 100 packets in the data flow.
35	SAMPLING_ALGORITHM: The type of algorithm being used. 0x01 indicates deterministic sampling and 0x02 indicates random sampling.
60	IP_PROTOCOL_VERSION: The IP protocol version being used.
62	IPV6_NEXT_HOP: The IPv6 address of the next-hop router.
70	MPLS_LABEL_1: The first MPLS label in the stack.
71	MPLS_LABEL_2: The second MPLS label in the stack.
72	MPLS_LABEL_3: The third MPLS label in the stack.

**Figure 20: Version 9 Data FlowSet Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flowset ID = Template ID		Length	
Record 1 - Field Value 1		Record 1 - Field Value 2	
Record 1 - Field Value 3		...	
Record 2 - Field Value 1		Record 2 - Field Value 2	
Record 2 - Field Value 3		Record 2 - Field Value 2	
Record 3 - Field Value 1		...	
...		Padding	

129106

**Table 29: Version 9 Data FlowSet Format**

Field	Description
FlowSet ID = Template ID	Data FlowSet that associated with a FlowSet ID. The FlowSet ID maps to a previously generated template ID. The flow collector must use the FlowSet ID to find the corresponding template record and decode the flow records from the FlowSet.
Length	FlowSet length. Data FlowSets are fixed in length.
Record Number - Field Value Number	Flow data records, each containing a set of field values. The template record identified by the FlowSet ID dictates the type and length of the field values.
Padding	Bytes (in zeros) that the exporter inserts so that the subsequent FlowSet starts at a 4-byte aligned boundary.

**Figure 21: Version 9 Options Template Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flowset ID = 1		Length	
Template ID		Option Scope Length	
Option Length		Scope 1 Field Type	
Scope 1 Field Length		...	
Scope N Field Length		Option 1 Field Type	
Option 1 Field Length		...	
Option M Field Length		Padding	

9016780

**Table 30: Version 9 Options Template Format**

Field	Description
FlowSet ID	FlowSet type. FlowSet ID 1 is reserved for the options template.
Length	FlowSet length. Option template FlowSets are fixed in length.
Template ID	Template ID of the options template. Options template values are greater than 255.
Option Scope Length	Length, in bytes, of any scope field definition that is part of the options template record.
Scope 1 Field Type	Relevant process. The JUNOS Software supports the system process (1).
Scope 1 Field Length	Length, in bytes, of the option field.

**Table 30: Version 9 Options Template Format (continued)**

Field	Description
Padding	Bytes the exporter inserts so that the subsequent FlowSet starts at a 4-byte aligned boundary.

**Figure 22: Active Flow Monitoring Version 9 Options Data Record Format**

Byte 3	Byte 2	Byte 1	Byte 0
Flowset ID = Template ID		Length	
Record 1 - Scope 1 Value		Record 1 - Option Field 1 Value	
Record 1 - Option Field 2 Value		...	
Record 2 - Option Field 2 Value		...	
Record 3 - Scope 1 Value		Record 3 - Option Field 1 Value	
...		Padding	

682910

**Table 31: Active Flow Monitoring Version 9 Options Data Record Format**

Field	Description
FlowSet ID = Template ID	ID that precedes each options data flow record. The FlowSet ID maps to a previously generated template ID. The collector must use the FlowSet ID to find the corresponding template record and decode the options data flow records from the FlowSet.
Length	FlowSet length. Option FlowSets are fixed in length.
Number of Flow Data Records	Remainder of the options data FlowSet is a collection of flow data records, each containing a set of field values. The template record identified by the FlowSet ID dictates the type and length of the field values.
Padding	Bytes (in zeros) the exporter inserts so that the subsequent FlowSet starts at a 4-byte aligned boundary.

## More Information About Passive and Active Flow Monitoring

To learn more about passive flow monitoring, active flow monitoring, cflowd versions 5 and 8, and flow monitoring version 9 see the following:

- Version 9: RFC 3954 at <http://www.faqs.org/rfcs/rfc3954.html>
- Versions 5 and 8: Cooperative Association for Internet Data Analysis (CAIDA) website at <http://www.caida.org>.
- *JUNOS Services Interfaces Configuration Guide*

- *JUNOS Policy Framework Configuration Guide*
- Internet draft draft-cavuto-dtcp-01.txt, *DTCP: Dynamic Tasking Control Protocol* (expires March 2007)

For more information on IPSec and the ES PIC, see the *JUNOS System Basics Configuration Guide*.

## **Part 2**

# **Index**

- Index on page 127



# Index

## A

active flow monitoring  
     destination-based example configuration  
         .....100

## C

configuration  
     flow-tap application.....87  
 content destinations  
     flow-tap.....84

## D

DTCP.....83  
 dynamic flow capture  
     configuration procedure.....35  
     example configuration.....65  
     operational mode commands.....67  
     options  
         configuring system logging.....38  
         configuring thresholds.....38  
         monitoring with SNMP.....39  
     overview.....34  
 Dynamic Tasking Control Protocol *See* DTCP

## E

example configurations  
     active flow monitoring, destination-based.....100

## F

flow collector interface *See* flow monitoring  
 flow monitoring.....82  
     configuration procedure  
         active flow monitoring.....69  
         dynamic flow capture.....35  
         flow collector interface.....28  
         passive flow monitoring.....13  
     destination-based example configuration.....100  
     example configuration  
         active flow monitoring.....89  
         discard accounting.....93

dynamic flow capture.....65  
 multiple port mirroring.....97  
 next-hop groups.....97  
 operational mode commands  
     active flow monitoring.....91  
     discard accounting.....96  
     dynamic flow capture.....67  
     flow collector interface.....60  
     passive flow monitoring.....48  
 options  
     applying a firewall filter to an output  
         interface.....28  
     configuring an aggregate export timer.....81  
     ES PIC.....26  
     next-hop groups.....82  
     port mirroring.....81  
     port mirroring with filter-based  
         forwarding.....82  
     stripping MPLS labels.....21  
     templates.....78  
 overview  
     active flow monitoring.....5  
     dynamic flow capture.....34  
     flow collector interface.....28  
     general.....3  
     passive flow monitoring.....4  
     system requirements.....6  
 flow-tap  
     application.....83  
     architecture.....84  
     permissions statement.....86  
     RADIUS configuration.....86  
     restrictions.....86  
     security.....86  
 flow-tap application  
     example configuration.....87  
 flow-tap-dtcp statement.....86

## L

lawful intercept architecture.....84

## M

mediation devices  
     flow-tap.....84

**N**

next-hop groups.....82

**S**

system requirements  
    flow monitoring.....6