



JUNOS® Software

SDK Applications Configuration Guide and Command Reference

Release 10.0

Juniper Networks, Inc.

1194 North Mathilda Avenue
Sunnyvale, California 94089
USA

408-745-2000

www.juniper.net

Published: 2009-10-12

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 SDK Applications Configuration Guide and Command Reference

Release 10.0

Copyright © 2009, Juniper Networks, Inc.

All rights reserved. Printed in USA.

Writing: Joanne McClintock

Editing: Sonia Saruba

Illustration: Faith Bradford

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 NONINFRINGEMENT. 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)).

Abbreviated Table of Contents

About This Guide

xv

Part 1

JUNOS SDK Applications Configuration

Chapter 1	Introduction to Configuring JUNOS SDK Applications	3
Chapter 2	The Multiservices PIC Configuration	5
Chapter 3	Administrator-Defined Resource Limits for SDK Applications	13
Chapter 4	SDK Services Configuration Guidelines	19
Chapter 5	Stateful Firewalls in SDK Applications	23
Chapter 6	Traffic Sampling for JUNOS SDK Applications	27
Chapter 7	High Availability for the Services SDK	31
Chapter 8	Tracing Processes Specific to JUNOS SDK Applications	33
Chapter 9	Configuration Mode Commands for SDK Applications	37

Part 2

Configuration Summaries and Command Reference

Chapter 10	Summary of SDK Configuration Mode Commands	43
Chapter 11	Summary of SDK Configuration Statements	47
Chapter 12	SDK Operational Command Reference	63

Part 3

Index

Index	99
Index of Statements and Commands	103

Table of Contents

	About This Guide	xv
	JUNOS Documentation and Release Notes	xv
	Objectives	xvi
	Audience	xvi
	Supported Routing Platforms	xvi
	Using the Indexes	xvii
	Using the Examples in This Manual	xvii
	Merging a Full Example	xvii
	Merging a Snippet	xviii
	Documentation Conventions	xix
	Documentation Feedback	xx
	Requesting Technical Support	xxi
	Self-Help Online Tools and Resources	xxi
	Opening a Case with JTAC	xxi
Part 1	JUNOS SDK Applications Configuration	
Chapter 1	Introduction to Configuring JUNOS SDK Applications	3
	JUNOS SDK Applications Overview	3
	Enabling the SSD and SDK Application Deployment	4
Chapter 2	The Multiservices PIC Configuration	5
	Configuring Control and Data Cores	5
	Configuring Flow Affinity on the Data Plane	6
	Configuring Object Cache, Policy Database, and Forwarding Database	6
	Configuring Packages on the PIC	8
	Configuring System Log Messages	9
	Configuring Wired Process Memory	10

Chapter 3	Administrator-Defined Resource Limits for SDK Applications	13
	Overview of Administrator-Defined Resource Limits for SDK Applications	13
	Configuring Resource Limits	14
	Displaying Resource Limits	15
	Example: Configuring Separate Resource Limits for a Process	16
Chapter 4	SDK Services Configuration Guidelines	19
	Configuring Service Sets for JUNOS SDK Applications	19
	Configuring the Service Order for JUNOS SDK Service Sets	20
	Configuring the Sampling Service Set	20
	Example: JUNOS SDK Service Set Configuration	21
Chapter 5	Stateful Firewalls in SDK Applications	23
	Loading the Stateful Firewall Plug-In	23
	Configuring Memory for the Stateful Firewall Plug-In	24
	Configuring rsh, rlogin, rexec for Stateful Firewall	25
	Using Stateful Firewall Operational Commands	26
Chapter 6	Traffic Sampling for JUNOS SDK Applications	27
	Traffic Sampling for JUNOS SDK Applications Overview	27
	Limitations and Constraints of SDK Traffic Sampling	27
	Configuring Traffic Sampling for JUNOS SDK Applications	28
	Example: Traffic Sampling on a Multiservices PIC	29
Chapter 7	High Availability for the Services SDK	31
	Configuring High Availability for the Services SDK	31
	Commands Supporting High Availability for Multiservices PICs	32
Chapter 8	Tracing Processes Specific to JUNOS SDK Applications	33
	Tracing Process Monitoring Operations for JUNOS SDK Applications	33
	Tracing System Resource Cleanup Operations for JUNOS SDK Applications	34

Chapter 9	Configuration Mode Commands for SDK Applications	37
	Displaying the Package Name for JUNOS SDK Application	
	Configurations	37
	Displaying and Deleting the Configuration for JUNOS SDK Applications	38
	How the extension show Command Matches Package Names	38
	Using the extension show Command to Display SDK Configuration by	
	Package Name	39
	Using the extension delete Command to Delete SDK Configuration by Package	
	Name	40
Part 2	Configuration Summaries and Command Reference	
Chapter 10	Summary of SDK Configuration Mode Commands	43
	extension package-name (show delete)	44
	show display detail (SDK)	45
Chapter 11	Summary of SDK Configuration Statements	47
	extension-provider	48
	extension-service	49
	extensions	50
	process-monitor	51
	resource-cleanup	52
	resource-limits	53
	resources	55
	service-order	56
	syslog	57
	traceoptions	58
	traceoptions (Process Monitor)	59
	traceoptions (Resource Cleanup)	61
Chapter 12	SDK Operational Command Reference	63
	clear interfaces statistics (SDK)	64
	clear services stateful-firewall flows (SDK)	65
	request interface (revert switchover) (SDK)	66
	show chassis pic (SDK)	67
	show extension-provider system connections	68
	show extension-provider system packages	71
	show extension-provider system processes	73
	show extension-provider system uptime	78
	show extension-provider system virtual-memory	79
	show interfaces redundancy (SDK)	82

show interfaces statistics (SDK)	83
show services stateful-firewall flows (SDK)	86
show services stateful-firewall statistics (SDK)	87
show system processes (SDK)	88
show system processes health	89
show system processes providers	92
show system processes resource-limits process-name	93
show system resource-cleanup processes	95
show version (SDK)	96

Part 3

Index

Index	99
Index of Statements and Commands	103

List of Tables

	About This Guide	xv
	Table 1: Notice Icons	xix
	Table 2: Text and Syntax Conventions	xix
Part 1	JUNOS SDK Applications Configuration	
Chapter 2	The Multiservices PIC Configuration	5
	Table 3: Severity Levels for SDK Syslog Messages	9
Part 2	Configuration Summaries and Command Reference	
Chapter 12	SDK Operational Command Reference	63
	Table 4: show extension-provider system virtual-memory Output Fields	79
	Table 5: show services stateful-firewall flows Output Fields	86
	Table 6: show services stateful-firewall statistics Output Fields	87
	Table 7: show system processes health Output Fields	89
	Table 8: show system processes resource-limits process-name Output Fields	93
	Table 9: show system resource-cleanup processes Output Fields	95
	Table 10: show version Output Fields	96

About This Guide

This preface provides the following guidelines for using the *JUNOS® Software SDK Applications Configuration Guide and Command Reference*:

- JUNOS Documentation and Release Notes on page xv
- Objectives on page xvi
- Audience on page xvi
- Supported Routing Platforms on page xvi
- Using the Indexes on page xvii
- Using the Examples in This Manual on page xvii
- Documentation Conventions on page xix
- Documentation Feedback on page xx
- Requesting Technical Support on page xxi

JUNOS Documentation and Release Notes

For a list of related JUNOS documentation, see <http://www.juniper.net/techpubs/software/junos/>.

If the information in the latest release notes differs from the information in the documentation, follow the *JUNOS Software Release Notes*.

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

Juniper Networks supports a technical book program to publish books by Juniper Networks engineers and subject matter experts with book publishers around the world. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration using JUNOS Software and Juniper Networks devices. In addition, the Juniper Networks Technical Library, published in conjunction with O'Reilly Media, explores improving network security, reliability, and availability using JUNOS configuration techniques. All the books are for sale at technical bookstores and book outlets around the world. The current list can be viewed at <http://www.juniper.net/books>.

Objectives

This guide provides an overview of the configuration features and operational commands Juniper Networks supports for JUNOS SDK applications.



NOTE: For additional information about JUNOS Software—either corrections to or information that might have been omitted from this guide—see the software release notes at <http://www.juniper.net/>.

Audience

This guide is designed for network administrators who are configuring and monitoring a Juniper Networks M Series, MX Series, T Series, EX Series, or J Series router or switch.

To use this guide, you need a broad understanding of networks in general, the Internet in particular, networking principles, and network configuration. You must also be familiar with one or more of the following Internet routing protocols:

- Border Gateway Protocol (BGP)
- Distance Vector Multicast Routing Protocol (DVMRP)
- Intermediate System-to-Intermediate System (IS-IS)
- Internet Control Message Protocol (ICMP) router discovery
- Internet Group Management Protocol (IGMP)
- Multiprotocol Label Switching (MPLS)
- Open Shortest Path First (OSPF)
- Protocol-Independent Multicast (PIM)
- Resource Reservation Protocol (RSVP)
- Routing Information Protocol (RIP)
- Simple Network Management Protocol (SNMP)

Personnel operating the equipment must be trained and competent; must not conduct themselves in a careless, willfully negligent, or hostile manner; and must abide by the instructions provided by the documentation.

Supported Routing Platforms

For the features described in this manual, the JUNOS SDK is supported in its entirety on the following routers:

- Juniper Networks M Series Multiservice Edge Routers, except for dynamic firewall filters on the M40e router.

Among the M7i and M10i routers, the `junos_dfw` APIs (implicit firewall filters) are supported on the following systems:

- M7i router with Enhanced Forwarding Engine
- M10i router with Enhanced Forwarding Engine
- M7i router with integrated Multiservices 100 PIC (available only with Enhanced Forwarding Engine)
- Juniper Networks T Series Core Routers

In addition, the RE SDK module of the SDK is supported on the Juniper Networks SRX210, SRX240, and SRX650 Service Gateways.

Using the Indexes

This guide contains two indexes: a complete index of all index entries, and an index of statements and commands only.

The complete index points to pages in the statement summary chapters. The index entry for each configuration statement contains at least two entries:

- The first entry points to the statement summary section.
- The second entry, *usage guidelines*, points to the section in a configuration guidelines chapter that describes how to use the statement.

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the `load merge` or the `load merge relative` command. These commands cause the software to merge the incoming configuration into the current candidate configuration. If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the `load merge` command.

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

Merging a Full Example

To merge a full example, follow these steps:

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

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

```
system {
```

```

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

```

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

```

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

```

Merging a Snippet

To merge a snippet, follow these steps:

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

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

```

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

```

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

```

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

```

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

```

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

```

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

Documentation Conventions

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

Table 1: Notice Icons





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

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

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the <code>configure</code> command: user@host> configure
Fixed-width text like this	Represents output that appears on the terminal screen.	user@host> show chassis alarms No alarms currently active
<i>Italic text like this</i>	<ul style="list-style-type: none"> Introduces important new terms. Identifies book names. Identifies RFC and Internet draft titles. 	<ul style="list-style-type: none"> A policy <i>term</i> is a named structure that defines match conditions and actions. <i>JUNOS System Basics Configuration Guide</i> RFC 1997, <i>BGP Communities Attribute</i>
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# set system domain-name <i>domain-name</i>

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

Convention	Description	Examples
Plain text like this	Represents names of configuration statements, commands, files, and directories; IP addresses; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none">■ To configure a stub area, include the stub statement at the [edit protocols ospf area area-id] hierarchy level.■ The console port is labeled CONSOLE.
< > (angle brackets)	Enclose optional keywords or variables.	stub <default-metric <i>metric</i> >;
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	broadcast multicast (<i>string1</i> <i>string2</i> <i>string3</i>)
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	rsvp { # Required for dynamic MPLS only
[] (square brackets)	Enclose a variable for which you can substitute one or more values.	community name members [<i>community-ids</i>]
Indentation and braces ({ })	Identify a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
J-Web GUI Conventions		
Bold text like this	Represents J-Web graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none">■ In the Logical Interfaces box, select All Interfaces.■ To cancel the configuration, click Cancel.
> (bold right angle bracket)	Separates levels in a hierarchy of J-Web selections.	In the configuration editor hierarchy, select Protocols > Ospf .

Documentation Feedback

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

- Document or topic name
- URL or page number

- Software release version (if applicable)

Requesting Technical Support

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

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the JTAC User Guide located at <http://www.juniper.net/customers/support/downloads/710059.pdf>.
- Product warranties—For product warranty information, visit <http://www.juniper.net/support/warranty/>.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

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

- Find CSC offerings: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes: <http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://www.juniper.net/alerts/>
- Join and participate in the Juniper Networks Community Forum: <http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

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

Opening a Case with JTAC

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

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

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

Part 1

JUNOS SDK Applications Configuration

- Introduction to Configuring JUNOS SDK Applications on page 3
- The Multiservices PIC Configuration on page 5
- Administrator-Defined Resource Limits for SDK Applications on page 13
- SDK Services Configuration Guidelines on page 19
- Stateful Firewalls in SDK Applications on page 23
- Traffic Sampling for JUNOS SDK Applications on page 27
- High Availability for the Services SDK on page 31
- Tracing Processes Specific to JUNOS SDK Applications on page 33
- Configuration Mode Commands for SDK Applications on page 37

Chapter 1

Introduction to Configuring JUNOS SDK Applications

This chapter includes the following topics:

- JUNOS SDK Applications Overview on page 3
- Enabling the SSD and SDK Application Deployment on page 4

JUNOS SDK Applications Overview

The JUNOS Software Development Kit (SDK) allows members of the Juniper Open IP Solution Development Program (OSDP) to build custom applications that run on the Juniper Networks JUNOS operating system and extend the functionality of JUNOS systems. Such an application may run on the Routing Engine or, to perform a specific service, on the Multiservices PIC. In JUNOS Software user documentation, these third-party applications are called *SDK applications*. These applications are installed in one or more packages.

A JUNOS SDK application may already be on your router if it was provided to you by a third party, or you may need to install the SDK application if you acquired it separately from the router.

To install an SDK application, please consult the application-specific documentation supplied by the provider. The application-specific documentation may also have information about configuring the SDK application on the router that is in addition to the generic information in this collection of topics.

In the configuration itself, the SDK application is called an *extension*. The third-party creator of that application is called a *provider*. Also specific to SDK applications is the SDK service process, or *ssd*. This process, which runs on the Routing Engine, is responsible for communications between the SDK application and the regular JUNOS Software. Although *ssd* is present on the router, it does not run unless specifically enabled.

For security, an SDK application comes with a certificate that authenticates it as a product of a specific provider. Part of this certificate, the *provider ID*, must be activated on the router to allow the SDK application to be deployed on the router and run.

The *Services SDK* (previously called the MP SDK) is the JUNOS SDK module that supports the development and running of SDK applications on the Multiservices PIC. The Multiservices PIC is based on a multicore chip. For SDK applications that are

installed to run on the Multiservices PIC, you can designate the number of cores used for control versus data handling. Your application provider may recommend values for this core allocation, or you may choose these values yourself.

For more information about the Juniper OSDP and the JUNOS SDK, please contact your account team or visit <http://www.juniper.net/partners/osdp.html>.

- Related Topics**
- Configuring Control and Data Cores on page 5
 - Enabling the SSD and SDK Application Deployment on page 4

Enabling the SSD and SDK Application Deployment

By default, the SDK service process (ssd) does not run. To enable **ssd** and prepare the router for installation of a JUNOS SDK application, include the following statements at the **[edit system]** hierarchy level:

```
[edit]
system {
  extensions {
    providers {
      provider-id;
    }
  }
}
```

The provider ID is a prefix that is part of the certificate name used by the provider in building an SDK application. Before installing your SDK application, you must enable its provider ID using the **extensions** statement. Enabling a provider ID allows you to install the SDK application package that was built using that certificate name. Multiple provider IDs can be enabled on a router.

The following example shows how to use **extensions** statement to specify provider IDs in the configuration.

If **abc** and **xyz** are provider IDs issued to two providers, then the following configuration enables the router for the SDK applications built by either provider:

```
[edit]
system {
  extensions {
    providers {
      abc;
      xyz;
    }
  }
}
```

- Related Topics**
- Configuring Packages on the PIC on page 8

Chapter 2

The Multiservices PIC Configuration

To configure an SDK application, include the following statements at the [edit chassis fpc *slot-number* pic *pic-number* adaptive-services service-package extension-provider] hierarchy level:

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
  control-cores control-number;
  data-cores data-number;
  data-flow-affinity;
  forwarding-db-size size;
  object-cache-size value;
  package package-name;
  policy-db-size size;
  wired-process-mem-size size;
  syslog {
    facility {
      severity;
      destination destination;
    }
  }
}
```

For details on configuring these statements, see the following topics:

- Configuring Control and Data Cores on page 5
- Configuring Flow Affinity on the Data Plane on page 6
- Configuring Object Cache, Policy Database, and Forwarding Database on page 6
- Configuring Packages on the PIC on page 8
- Configuring System Log Messages on page 9
- Configuring Wired Process Memory on page 10

Configuring Control and Data Cores

There are eight cores in a PIC. Some cores, called *control cores*, are dedicated to running control functionality for the application. Cores dedicated to processing data for the application are called *data cores*.

To configure control and data cores, use the `control-cores` and `data-cores` statements, respectively, at the `[edit chassis fpc slot-number pic pic-number adaptive-services service-package extension-provider]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
    control-cores control-number;
    data-cores data-number;
}
```

You must designate at least one core as a control core. Although it is not mandatory to designate any cores as data cores, it is advisable to designate a minimum of five, depending on the nature of the application, to achieve good performance. The total number of cores, both control and data cores, that you can dedicate using the `extension-provider` statement ranges from one through eight. Any cores not configured as control or data cores are treated as *user cores*.



NOTE: For help with architecting their JUNOS SDK applications, providers should consult with JUNOS SDK Developer Support.

Configuring Flow Affinity on the Data Plane

As of JUNOS Release 9.5, the Services SDK (a module of the JUNOS SDK) supports flow affinity behavior for the data CPUs. Flow affinity distribution is based on a hash distribution. Flow affinity is already the default behavior for the control CPUs, but the default behavior for distributing data packets over data cores is in a round-robin fashion.

You can change the default behavior from round-robin to flow affinity for the data cores by adding the `data-flow-affinity` statement at the `[edit chassis fpc slot-number pic slot-number adaptive-services service-package extension-provider]` hierarchy level.



NOTE: Either adding the `data-flow-affinity` statement or removing it will cause the PIC to reboot.

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
    data-flow-affinity;
}
```

Configuring Object Cache, Policy Database, and Forwarding Database

To tune SDK application performance, use the `object-cache-size`, `forwarding-db-size`, and `policy-db-size` statements at the `[edit chassis fpc slot-number pic pic-number adaptive-services service-package extension-provider]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
```

```

forwarding-db-size size;
object-cache-size value;
policy-db-size size;
wired-process-mem-size size;
}

```

Both the forwarding database (FDB) and the policy database (PDB) are carved out of object cache ($PDB + FDB \leq \text{object cache}$).

The **policy-db-size** statement defines the size of policies that providers expect to be present in their system. It is configured in megabytes. The size should be less than that set for the **object-cache-size** statement.

The FDB provides access to the route information.



NOTE: You need to enable the **forwarding-options sampling** statement for the FDB to be created. For information on configuring this statement, see “Configuring Traffic Sampling for JUNOS SDK Applications” on page 28.

For object cache, specify a value that is a multiple of 128 megabytes (MB) and up to 512 MB for the Multiservices 100 PIC or up to 1280 MB for the Multiservices 400 PIC. However, if you set wired process memory as well, the maximum value for the object cache on the Multiservices 100 PIC is 128 MB and 768 MB on the Multiservices 400 PIC.



NOTE: Changing the object cache size on a running system causes the PIC to reboot.

For the policy database, the current recommendations when configuring Multiservices PICs are:

- Do not exceed a policy database size of 64 MB,
- Stay with one rule per term.
- Keep the object cache size high (1280 MB on Multiservices 400 PICs and DPCs and 512 MB on Multiservices 100 PICs).
- Do not configure anything for forwarding database.
- Keep the number of service sets per Multiservices PIC below 1000. (For more on service sets, see “Configuring Service Sets for JUNOS SDK Applications” on page 19.)

When configuring the stateful firewall internal plug-in, some questions remain regarding the upper limit to specify for the `policy-db-size`, `object-cache-size`, and `forwarding-db-size` statements when the application will use a large number of rules, causing the total memory required to approach the size of the object cache configured. The following limits, which are specific to the stateful firewall configuration, await additional review:

- Maximum number of terms (with one rule per term) per service set: 1200
- Maximum number of service sets per Multiservices PIC: 4000 (M Series and T Series routers), 6000 (MX Series and M120 routers)
- Maximum object cache size: 1280 MB (Multiservices 400 PICs and DPCs), 512 MB (Multiservices 100 PICs)
- Maximum policy database size: Still to be determined.

If the policy database is set too small, an error message will be logged in the router message file even though the commit may appear to be successful. You need to check the logs, and not find any message file errors there, to be sure that the stateful firewall commit was indeed successful. The remedial action is to increase the size of the policy database.

Configuring Packages on the PIC

JUNOS SDK applications are installed on the Multiservices PIC in one or more packages. To designate which SDK application package to install on a given PIC, include the `package package-name` option at the `[edit chassis fpc slot-number pic pic-number adaptive-services service-package extension-provider]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
    package package-name;
}
```

Up to eight packages can be installed on a PIC; however, only one data package is allowed per PIC.

As of JUNOS Release 9.5, a stateful firewall plug-in is provided as part of jbundle. To load this plug-in on the PIC, include the `package jservices-sfw` statement at the `[edit chassis fpc slot-number pic slot-number adaptive-services service-package extension-provider]` hierarchy level.

```
user@host# show chassis
fpc 0;
  pic 0;
    adaptive-services;
      service-package;
        extension-provider;
          control-cores 1;
          data-cores 4;
          object-cache-size 128;
          package jservices-sfw;
          policy-db-size 64;
        }
```

```
    }  
  }  
}
```



NOTE: You cannot install both a JUNOS service package and an SDK application package on the same PIC. However, you can load both the jservices-sfw package and a JUNOS SDK application package on the same PIC.

Related Topics ■ show chassis pic

Configuring System Log Messages

To record or view system log messages on a specific PIC, include the **syslog** statement at the [edit chassis fpc slot-number pic pic-number adaptive-services service-package extension-provider] hierarchy level:

```
[edit chassis fpc slot-number pic slot-number adaptive-services service-package]
extension-provider {
  syslog {
    facility {
      severity;
      destination destination;
    }
  }
}
```

Each system log message belongs to a facility, which is a group of messages that are either generated by the same software process or concern a similar condition or activity. Each message is also preassigned a severity level, which indicates how seriously the triggering event affects router functions.

For the JUNOS SDK, there are four values for facility that log either actions performed or errors encountered by the following entities:

- **daemon**—Various system processes.
- **external**—Local external applications.
- **kernel**—The PIC kernel.
- **pfe**—The Packet Forwarding Engine.

The severity option has the same values as it does in the native JUNOS Software. See for possible values.

Table 3: Severity Levels for SDK Syslog Messages

Level	Description
any	Include all severity levels.

Table 3: Severity Levels for SDK Syslog Messages *(continued)*

Level	Description
none	Disable logging of the associated facility to a destination.
emergency	System panic or other condition that causes the routing platform to stop functioning.
alert	Conditions that require immediate correction, such as a corrupted system database.
critical	Critical conditions, such as hard errors.
error	Error conditions that generally have less serious consequences than errors in the emergency, alert, and critical levels.
warning	Conditions that warrant monitoring.
notice	Conditions that are not errors but might warrant special handling.
info	Events or nonerror conditions of interest.

Enhancements to the existing infrastructure make debugging on the Multiservices PIC easier by giving the user the option of redirecting the log messages to either the Routing Engine (**routing-engine**) or to the console of the PIC (**pic-console**). The user does not have to specify a destination for the messages; by default all messages go to **/var/log/messages** on the Routing Engine. When the syslog destination is configured to redirect the log messages to the Routing Engine, using the CLI **set system syslog** command, available in the native JUNOS Software, overrides the syslog settings made on the Multiservices PIC.

To record or view system log messages on a specific PIC, include the **syslog** statement. System log information is passed to the Routing Engine and put in the **/var/log/messages** directory. Four facilities are supported: **daemon**, **external**, **kernel**, and **pfe**. These facilities log actions performed or errors encountered by various system processes, the local external applications, the PIC kernel, or the Packet Forwarding Engine, respectively. Severity is the same as for the JUNOS Software; see the **syslog** configuration statement summary for the severity levels that you can specify.

Configuring Wired Process Memory

Wired process memory is memory used by the operating system that is generally “off limits” to another application. To configure wired process memory size, specify 512 MB for the **wired-process-mem-size** statement at the **[edit chassis fpc slot-number pic pic-number adaptive-services service-package extension-provider]** hierarchy level.. In addition, you can also configure the object cache.

```
[edit chassis fpc slot-number pic pic-number adaptive-services service-package]
extension-provider {
    object-cache-size value;
    wired-process-mem-size size;
```



```
}
```

Currently, 512 MB is the default size of wired process memory and the one size of wired process memory available.

Chapter 3

Administrator-Defined Resource Limits for SDK Applications

The following topics describe the configuration statements and operational commands available to the network administrator for setting resource limits for SDK applications:

- Overview of Administrator-Defined Resource Limits for SDK Applications on page 13
- Configuring Resource Limits on page 14
- Displaying Resource Limits on page 15
- Example: Configuring Separate Resource Limits for a Process on page 16

Overview of Administrator-Defined Resource Limits for SDK Applications

The JUNOS SDK has four levels of policies to ensure that SDK applications have the minimal impact on the native JUNOS operating system and system operations. These policies impose limits on the system resources the SDK application uses. Each succeeding policy level overrides the previous level's settings, provided the constraints are within the previous level's settings.

The Level I policy is the default policy generated by Juniper Networks. Level II is a per-provider level policy that has not yet been implemented in the SDK (it is the same for all providers and the policy file is a part of the JUNOS Software). A Level III policy is implemented in the policy file that SDK developers write for each SDK package.

In building an application package, the SDK provider creates a manifest and specifies roles. The manifest file specifies the contents of the application package, and the provider specifies a role name for each entry in the package manifest file. Before packaging the application, the provider has the option to create a policy file that describes the constraints to be applied to the Provider_Daemon role. This role can be defined in either a Level II or Level III policy as specified in the manifest. This policy file limits the system resources the application uses in addition to the limits imposed in the default policy generated by Juniper Networks (Level I policy).

Configuring Resource Limits

The Level IV policy is set by the administrative user using the `resource-limits` statement at the `[edit system extensions]` hierarchy level. The limits imposed by this policy can be configured either by package or by individual processes in the package.

```
[edit system extensions]
resource-limits {
  package package-name {
    resources {
      cpu {
        priority number;
        time seconds;
      }
      file {
        core-size bytes;
        open number;
        size bytes;
      }
      memory {
        data-size bytes;
        locked-in bytes;
        resident-set-size bytes;
        socket-buffers bytes;
        stack-size bytes;
      }
    }
  }
  process process-ui-name {
    resources {
      cpu {
        priority number;
        time seconds;
      }
      file {
        core-size bytes;
        open number;
        size bytes;
      }
      memory {
        data-size bytes;
        locked-in bytes;
        resident-set-size bytes;
        socket-buffers bytes;
        stack-size bytes;
      }
    }
  }
}
```

Limits defined for individual processes override the limits defined for an entire package. Any limits not set as a Level IV limits inherit the limits from Level III if they exist or from Level II.

Level IV policies can be more restrictive than previous policy levels, but they cannot raise the limits set by the other levels.

If an SDK application exceeds any of the imposed limits, the router logs it. For example, if a process tries to exceed its stack size, the process will get killed and the system will generate a core file.

Level IV policies can be applied either during runtime of the application or before the application gets started by `init()`. However, if the SDK application was already running, it must be restarted manually in order to allow for the new limits to take effect.

If you try to commit a resource limit that is higher (less stringent) than the inherited value, the commit will fail with the following error message:

```
[edit system extensions resource-limits]
  'process jnx-example-service'
    Limit validation failed for program 'jnx-example-service', resource 'file'
    limit 'open': raising limits defined in role 'Provider_Daemon' is not allowed.
  commit complete

[edit system extensions resource-limits]
```

If you delete a resource configuration, the setting goes back to the limits from the assigned role in the manifest file (Level II or Level III).

Displaying Resource Limits

To display the applied policies, use the `show system processes resource-limits process-name process-ui-name` operational command. The following example configuration, when committed, applies resource limits for an SDK package `jnx-example` and overrides it with process-level settings for the process `jnx-example-service`:

```
[edit system extensions]
user@router# show
resource-limits {
  package jnx-example {
    resources {
      memory {
        stack-size 4m;
      }
    }
  }
  process jnx-example-service {
    resources {
      file {
        size 4m;
      }
    }
  }
}
```

Using the `show system processes resource-limits process-name` command, the output for `jnx-example-foo-binary`, part of package `jnx-example`, looks like the following because the package-level settings are applied on it:

```
user@router> show system processes resource-limits process-name
jnx-example-foo-binary
Resource Limits:
  Area                      Max. allowed  Max. configurable
  memory/stack-size         4MB           8MB
  memory/data-size          32MB          32MB
  memory/resident-set-size  24MB          24MB
  memory/locked-in          16MB          16MB
  cpu/priority               10            10
  file/open                  64            64
```

The output for `jnx-example-service` looks like this:

```
user@router> show system processes resource-limits process-name jnx-example-service
Resource Limits:
  Area                      Max. allowed  Max. configurable
  file/size                 4MB           unlimited
  file/open                 64            64
  cpu/priority               10            10
  memory/stack-size         8MB           8MB
  memory/data-size          32MB          32MB
  memory/resident-set-size  24MB          24MB
  memory/locked-in          16MB          16MB
```

For more detail on the `show system processes resource-limits process-name` *process-ui-name* operational command, see its command summary.

Related Topics ■ `show system processes resource-limits process-name`

Example: Configuring Separate Resource Limits for a Process

In this example there are 10 processes in the SDK application package, and the administrative user wants to limit the number of open files and the stack size but does not want to apply the same limits to the `jnx-example-service` process. To do this, the user must specify limits for the `jnx-example` package and for the `jnx-example-service` process. Any limits not set inherit the Level III limits (if any) or Level II limits (if no Level III limits exist).

```
user@router# show
resource-limits {
  package jnx-example {
    resources {
      file {
        open 8;
      }
      memory {
        stack-size 4m;
      }
    }
  }
  process jnx-example-service {
```

```

resources {
  memory {
    resident-set-size 8m;
  }
}
}

[edit]
user@router# commit

```



NOTE: The `show system processes resource-limits process-name` command output shows a maximum configurable column, which has been omitted in this example to better show where the maximum allowed values come from.

```

user@router> show system processes resource-limits process-name jnx-foo-service
Resource Limits
Area                               Max. allowed
cpu/priority                       10  <-- inherited from Level II
file/open                          8   <-- defined in config
memory/data-size                   32MB <-- inherited from Level II
memory/locked-in                   16MB <-- inherited from Level II
memory/resident-set-size           24MB <-- inherited from Level II
memory/stack-size                  4MB  <-- defined in config

```

The above output will be the same for all other programs in the `jnx-example` package except the `jnx-example-service` process for which there is a special configuration. The process-level configuration overrides the package-level configuration.

```

user@router> show system processes resource-limits process-name jnx-example-service
Resource Limits
Area                               Max. allowed
cpu/priority                       10  <-- inherited from Level II
file/open                          64  <-- inherited from Level II
memory/data-size                   32MB <-- inherited from Level II
memory/locked-in                   16MB <-- inherited from Level II
memory/resident-set-size           8MB  <-- defined in config
memory/stack-size                  8MB  <-- inherited from Level II

```


Chapter 4

SDK Services Configuration Guidelines

This chapter includes the following topics:

- Configuring Service Sets for JUNOS SDK Applications on page 19
- Configuring the Service Order for JUNOS SDK Service Sets on page 20
- Configuring the Sampling Service Set on page 20
- Example: JUNOS SDK Service Set Configuration on page 21

Configuring Service Sets for JUNOS SDK Applications

A *service set* is a collection of policies taken from multiple services that can be applied as a unit to traffic coming to the PIC.

For SDK applications, to include a defined service in a service set, you must reference it using the `extension-service` statement at the `[edit services service-set service-set-name]` hierarchy level.

```
[edit]
services {
  service-set service-set-name {
    extension-service service-name1;
    extension-service service-name2;
  }
}
```

Up to two SDK plug-ins are supported per PIC and can be chained together in one service set.



NOTE: If the `extension-service` statement is used, the `service-order` statement is mandatory.

To specify the order of the policies within a service set, configure the `service-order` statement at the `[edit services extension-service]` hierarchy level.

- Related Topics**
- Configuring the Service Order for JUNOS SDK Service Sets on page 20
 - Example: JUNOS SDK Service Set Configuration on page 21

Configuring the Service Order for JUNOS SDK Service Sets

The service order is the order in which services are applied for a given service set.

To configure the service order, include the `service-order` statement at the `[edit services service-set service-set-name extension-service]` hierarchy level.

```
[edit]
services {
  service-set service-set-name {
    extension-service service-name1 {
      extension-service service-name2 {
        service-order {
          forward-flow [ service-name1 service-name2 ];
          reverse-flow [ service-name1 service-name2 ];
        }
      }
    }
  }
}
```



NOTE: If the `extension-service` statement is specified, the `service-order` statement is mandatory. Service order should not be configured for native JUNOS internal services. For the internal services, there is a default service order.

The `service-order` statement must include all services defined in the service set. It is mandatory to specify the forward-flow service order and the reverse-flow service order. If the reverse-flow service order is not specified, the reverse-flow order is the reverse of the forward-flow service order. The exception to this is for the sampling service set type. .

To change the service order, delete the service order elements and then add them again in the new order.

- Related Topics**
- Configuring Service Sets for JUNOS SDK Applications on page 19
 - Configuring the Sampling Service Set on page 20

Configuring the Sampling Service Set

In addition to next-hop and interface service sets there is also a *sampling service set*. Interface and next-hop service sets are explained in the *JUNOS Services Interfaces Configuration Guide*.

The sampling service set is configured using the `sampling-service` statement at the `[edit services service-set service-set-name]` hierarchy level.

```
[edit]
services {
  service-set service-set-name {
    sampling-service {
      service-interface interface-name;
    }
  }
}
```

```

    }
  }
}

```

The *service interface* is the interface the sampling is taken from. In the case of a sampling service set, the service interface must be a Multiservices PIC interface with a subunit number of 0. If the subunit is not specified, 0 will be assumed. The **reverse-flow** statement is not mandatory in this case. If no reverse flow is configured, all the sampled traffic is considered to be forward traffic. The next example makes sure that any sampled packet to the ms-6/1/0.0 interface will have the service plug-ins plugin1 and plugin2 applied to it.

```

[edit]
services {
  service-set sset1 {
    sampling-service {
      service-interface ms-6/1/0;
    }
    extension-service plugin1;
    extension-service plugin2;
    service-order {
      forward-flow [ plugin1 plugin2 ];
      reverse-flow [ plugin1 plugin2 ];
    }
  }
}

```

As with the native JUNOS Software, to configure traffic sampling for an JUNOS SDK application, you must also create a firewall filter, apply it to the logical interface on which you want to sample traffic, and enable sampling.

- Related Topics**
- Traffic Sampling for JUNOS SDK Applications Overview on page 27
 - Configuring Service Sets for JUNOS SDK Applications on page 19
 - Configuring the Service Order for JUNOS SDK Service Sets on page 20
 - Example: JUNOS SDK Service Set Configuration on page 21

Example: JUNOS SDK Service Set Configuration

In the following configuration example, the **acme-svc1** service is defined by three rules (their content is unspecified) and the **acme-svc2** service is defined by a rule set made up of an unspecified number of rules. In this case, these services are defined at the [edit acme services] hierarchy level.

```

[edit]
acme {
  services {
    acme-svc1 { #Provider-defined service
      svc1-rule1 { # First rule's name
        . . . # First rule defined
      }
      svc1-rule2 { # Second rule's name
        . . . # Second rule defined
      }
    }
  }
}

```

```

    }
    svc1-rule3 { # Third rule's name
        . . . # Third rule defined
    }
}
acme-svc2 { # Provider-defined service
    rule-set svc2-rule-set { # Rule-set name
        [ rules rule-names ]; # Rules definitions start here
    }
}
}
}

```

At the [edit services] hierarchy level (no intervening “acme” level here), the **service-set sset1** is defined by referencing the three rule names for **acme-svc1** and the one rule set name for **acme-svc2** using the **service-set service-set-name extension-service** statement at the [edit services service-set service-set-name] hierarchy level.

The service order is also configured at the [edit services service-set service-set-name] hierarchy level.

The following is an example of configuring service sets, extension service rules, and the service order:

```

[edit]
services {
    service-set sset1 {
        extension-service acme-svc1 {
            svc1-rule1;
            svc1-rule2;
            svc1-rule3;
        }
        extension-service acme-svc2 {
            rule-set svc2-rule-set;
        }
        /* Now define the order */
        service-order {
            forward-flow [acme-svc1 acme-svc2];
            reverse-flow [acme-svc1 acme-svc2];
        }
    }
}

```

Chapter 5

Stateful Firewalls in SDK Applications

This topic covers the following configuration and management of stateful firewalls for the JUNOS SDK.

- Loading the Stateful Firewall Plug-In on page 23
- Configuring Memory for the Stateful Firewall Plug-In on page 24
- Configuring rsh, rlogin, rexec for Stateful Firewall on page 25
- Using Stateful Firewall Operational Commands on page 26

Loading the Stateful Firewall Plug-In

As of JUNOS Release 9.5, a stateful firewall plug-in is provided as part of jbundle. To load this plug-in on the PIC, include the `package jservices-sfw` statement at the `[edit chassis fpc slot-number pic slot-number adaptive-services service-package extension-provider]` hierarchy level. For example:

```
user@host# show chassis
fpc 0 {
  pic 2 {
    adaptive-services {
      service-package {
        extension-provider {
          control-cores 1;
          data-cores 4;
          object-cache-size 128;
          package jservices-sfw; #Loads stateful firewall plug-in.
          policy-db-size 64;
        }
      }
    }
  }
}
```

You can load both the `jservices-sfw` package and a JUNOS SDK application package on the same PIC.

The following example demonstrates the stateful firewall plug-in coexisting with a provider's plug-in:

```
[edit services]
service-set sset {
  interface-service {
```

```

    service-interface ms-0/0/0.0;
  }
  stateful-firewall-rules rule1;
  extension-service customer-plugin;
  service-order [stateful-firewall customer-plugin];
}
stateful-firewall {
  rule rule1 {
    match-direction input-output;
    term term1 {
      from {
        applications junos-ftp;
      }
      then {
        accept;
      }
    }
  }
  rule rule2 {
    match-direction input;
    term term1 {
      from {
        source-address {
          192.1.1.2/32;
        }
      }
      then {
        reject;
        syslog;
      }
    }
  }
}
}

```

Related Topics ■ Configuring Packages on the PIC on page 8

Configuring Memory for the Stateful Firewall Plug-In

When configuring the stateful firewall internal plug-in, some questions remain regarding the upper limit to specify for the **policy-db-size**, **object-cache-size**, and **forwarding-db-size** statements when the application will use a large number of rules, causing the total memory required to approach the size of the object cache configured. The following limits, which are specific to the stateful firewall configuration, await additional review:

- Maximum number of terms (with one rule per term) per service set: 1200
- Maximum number of service sets per Multiservices PIC: 4000 (Juniper Networks M Series Multiservice Edge Routers and T Series Core Routers), 6000 (Juniper Networks MX Series Ethernet Services Routers and M120 Multiservice Edge Routers)

- Maximum object cache size: 1280 MB (Multiservices 400 PICs and DPCs), 512 MB (Multiservices 100 PICs)
- Maximum policy database size: Still to be determined.

If the policy database is set too small, an error message will be logged in the router message file even though the commit may appear to be successful. It is necessary to check the logs to make sure that no message file error is found to be sure that the stateful firewall commit was indeed successful. The remedial action is to increase the size of the policy database.

Related Topics ■ Configuring Object Cache, Policy Database, and Forwarding Database on page 6

Configuring rsh, rlogin, rexec for Stateful Firewall

Some implementations of the rsh, rlogin, rexec mechanism require the remote host to authenticate the request by opening a separate TCP session to port 113 on the client host. By default, the stateful firewall does not allow this authentication flow to go through.

To open the authentication flow, include the `applications junos-ident` statement at the `[edit services stateful-firewall rule rule-name term term-name from]` hierarchy level:

```
[edit]
services {
  stateful-firewall {
    rule rule1 {
      term term1 {
        from {
          (source-address | destination-address);
          applications junos-ident;
        }
        then {
          accept;
        }
      }
    }
  }
}
```

To allow Kerberos-enabled rsh, rlogin, rexec through the stateful firewall, configure the following additional applications and include them in the stateful firewall terms:

```
[edit]
applications {
  application test-kerberos-kshell {
    Protocol tcp;
    destination-port kshell;
  }
  application test-kerberos-klogin {
    protocol tcp;
    destination-port klogin;
  }
}
```

```

}

services {
  stateful-firewall {
    rule rule1 {
      term term1 {
        from {
          applications [kerberos-klogin kerberos-kshell];
        }
        then {
          accept;
        }
      }
    }
  }
}

```

Using Stateful Firewall Operational Commands

The following stateful firewall operational commands support the **ms-** interface:

- **show services stateful-firewall flows**—Display stateful firewall flow table entries.
- **show services stateful-firewall statistics**—Display stateful firewall statistics. For this command, only rule and ALG statistics are given. In the extensive option, other statistics appear but do not populate correctly; those values are all zeroes.
- **clear services stateful-firewall flows**—Remove established flows from the flow table.

For more information on these commands, see their command summaries.

Chapter 6

Traffic Sampling for JUNOS SDK Applications

You can configure traffic sampling on a Multiservices PIC for JUNOS SDK applications. See the following topics:

- Traffic Sampling for JUNOS SDK Applications Overview on page 27
- Configuring Traffic Sampling for JUNOS SDK Applications on page 28
- Example: Traffic Sampling on a Multiservices PIC on page 29

Traffic Sampling for JUNOS SDK Applications Overview

For routers containing a Multiservices PIC, the existing **interface** statement at the [edit forwarding-options sampling family *family* output] hierarchy level is extended to allow you to use an interface of the **ms-** type (**ms-fpc/pic/port**) to sample traffic passing through the router.

As with the regular JUNOS Software, to configure traffic sampling for a JUNOS SDK application, you must complete three tasks: create a firewall filter, apply it to the logical interface on which you want to sample traffic, and enable sampling. There is also a sampling service set you should be aware of. For information on this new service set type, see “Configuring the Sampling Service Set” on page 20.

Limitations and Constraints of SDK Traffic Sampling

The following are limitations of traffic monitoring as supported for JUNOS SDK applications:

- Traffic monitoring is supported for IPv4 only.
- Firewall filters are used on ingress interfaces to direct packets for sampling. Sampling and port mirroring cannot be enabled on the same traffic.
- Only one **extension-service** statement can be used.
- Only one interface can be configured for sampling.

Related Topics

- Configuring Traffic Sampling for JUNOS SDK Applications on page 28
- Example: Traffic Sampling on a Multiservices PIC on page 29
- Configuring the Sampling Service Set on page 20

Configuring Traffic Sampling for JUNOS SDK Applications

To enable sampling on a Multiservices PIC, include the **input** and **family** statements at the [edit forwarding-options sampling] hierarchy level:

```
[edit]
forwarding-options {
  sampling {
    input {
      rate number;
    }
    family family {
      output {
        extension-service service-name {
          provider-specific rules;
        }
        interface ms-fpc/pic/port;
      }
    }
  }
}
```



NOTE: You must enable the **forwarding-options sampling** statement for the forwarding database to be created.

The **extension-service** statement provides a section of the configuration hierarchy in which the provider of your SDK application may have added its own traffic monitoring configuration statements. To enable sampling for SDK applications and be able to use the configuration statements the SDK application provider may have added, you must include the **extension-service** statement at the [edit forwarding-options sampling output] hierarchy level. For application-specific configuration guidelines, see the documentation provided with your application.



NOTE: If you use the **extension-service** statement, the only other statement you can include at the [edit forwarding-options sampling output] hierarchy level is the **interface** statement. In this case, you must set the **interface** statement to an interface with an **ms-** prefix.

- Related Topics**
- Traffic Sampling for JUNOS SDK Applications Overview on page 27
 - Configuring the Sampling Service Set on page 20
 - Example: Traffic Sampling on a Multiservices PIC on page 29

Example: Traffic Sampling on a Multiservices PIC

The following example shows a firewall filter **sample-monitor**, which, when attached to an interface, ensures that all traffic entering that interface with a source address matching address 10.1.1.1 is sampled and sent to the output interface **ms-2/0/0**.

```
[edit]
firewall {
  family inet {
    filter sample-monitor {
      term sample-term {
        from {
          source-address {
            10.1.1.1/32;
          }
        }
        then {
          sample;
          accept;
        }
      }
    }
  }
}
forwarding-options {
  sampling {
    input {
      family inet {
        rate 1;
      }
    }
    output {
      extension-service abc-sample {
        provider-specific rules;
      }
      interface ms-2/0/0;
    }
  }
}
```

You can attach the firewall filter created in the above sample to any interface in your network.

- Related Topics**
- Traffic Sampling for JUNOS SDK Applications Overview on page 27
 - Configuring Traffic Sampling for JUNOS SDK Applications on page 28

Chapter 7

High Availability for the Services SDK

This chapter includes the following topics

- Configuring High Availability for the Services SDK on page 31
- Commands Supporting High Availability for Multiservices PICs on page 32

Configuring High Availability for the Services SDK

You can pair two Multiservices PICs in high availability mode using the virtual **rms** interface. As part of the configuration, one of the PICs is set as the *primary* and the other as the *secondary* PIC. When the **rms** interface comes up, the primary PIC comes up as active and the secondary PIC as standby.

On failure of the active PIC, the standby PIC becomes active and traffic is switched over to the new active PIC. When the failed primary PIC is restored, it comes up as standby. The network administrator must explicitly request a reversion to the primary PIC to make it active again. Otherwise, it continues in standby mode until the active PIC fails.

To create the **rms** interface, include the **redundancy-options** statement at the [edit **interfaces** *interface-name*] hierarchy level:

```
[edit interfaces interface-name]  
redundancy-options {  
  primary ms-fpc/pic/port;  
  secondary ms-fpc/pic/port;  
  hot-standby;  
}  
unit logical-unit-number family inet ip-address;
```

For the *interface-name* variable, specify the value *rmsnumber*, where *number* is in the range from 0 to 127.



NOTE: To bring up the **rms** interface, you must include the unit *logical-unit-number* family **inet** statement.

When a new **rms** interface is created, PICs designated as primary and secondary will reboot automatically. If either the primary or secondary PIC, or both, are changed, the corresponding old and new **ms-** interfaces will reboot automatically.



NOTE: The rms interface is also supported for service sets, class of service (CoS), and routes.

Commands Supporting High Availability for Multiservices PICs

The following operational commands support Multiservices PIC redundancy:

- **clear interfaces statistics**—Clear interface statistics. When the **show interfaces statistics** command is next issued, the output will show the packets received since the last time the **clear interfaces statistics** command was issued. However, after a graceful Routing Engine switchover (GRES), issuing the **show interfaces statistics** command will show cumulative statistics.
- **request interface (revert | switchover)**—Manually change over to the other PIC. Use **revert** to revert to the primary interface; use **switchover** to change, or switch over, to the secondary interface. After the switchover, it is recommended you place the old active PIC offline temporarily so it comes up fresh as a new backup.
- **show interfaces redundancy**—Display general information about the rms and ms-interfaces redundancy. You can use this command to determine which PIC is currently active.
- **show interfaces statistics**—Display interface statistics, such as packets received, packets transmitted, and errors. This command has output in three different levels of detail.

For detail on these commands as used in support of the rms interface, including sample output, see their command summaries. For information on these commands in other contexts, see the *JUNOS Interfaces Command Reference*.

Chapter 8

Tracing Processes Specific to JUNOS SDK Applications

This chapter includes the following topics:

- Tracing Process Monitoring Operations for JUNOS SDK Applications on page 33
- Tracing System Resource Cleanup Operations for JUNOS SDK Applications on page 34

Tracing Process Monitoring Operations for JUNOS SDK Applications

The *process health monitor* process (pmond) is the central monitoring process for JUNOS SDK applications. It tracks resource usage and performs actions on processes when they trigger certain events (for example, when there is a runaway process event or when low-water or high-water resource marks are exceeded). Process monitoring ensures that SDK applications are operating appropriately and provides an interface for operators to monitor the impact of their SDK applications on the Routing Engine.

To trace process monitoring operations, include the `process-monitor` statement at the `[edit system processes]` hierarchy level:

```
[edit]
system {
  processes {
    process-monitor {
      disable;
      traceoptions {
        file;
        flag flag;
        level level;
        no-remote-trace;
      }
    }
  }
}
```

The **traceoptions** statement is the only container statement at the [edit system processes process-monitor] hierarchy level. The available flags for the **traceoptions** statement include:

- **all**—Enable all trace options flags.
- **process-tracking**—Display process life-cycle events and parent/child pedigree changes.
- **heartbeat**—Display heartbeat updates from applications.
- **ui**—Display tracing messages for UI operational commands.

Tracing System Resource Cleanup Operations for JUNOS SDK Applications

Using the JUNOS SDK, providers can have their SDK applications request and manage system resources. Some of this resource utilization is persistent across, for example, reboots or the restart of the application. Some system tasks such as deleting a package, disabling an application, or accessing shared resources require that resources be cleaned up by entities other than the application itself. Resources that are known to need cleaning up include the following:

- GENCFG blobs
- SYSV shared memory segments
- SYSV semaphores
- Temporary files

Currently, the **traceoptions** statement is the only CLI statement available for configuring resource cleanup.

To configure tracing operations for resource cleanup operations, include the **traceoptions flag** option for selectively turning the debugging of trace messages on or off:

```
[edit]
system {
  processes {
    resource-cleanup {
      traceoptions {
        flag (all | events | gencfg | sysvsem | sysvshm | ui);
        level level;
      }
    }
  }
}
```

The available flags for the **traceoptions** statement include:

- **all**—Enable all trace option flags.
- **events**—Display process state change and cleanup events.
- **gencfg**—Display GENCFG blobs recorded for cleanup.
- **sysvsem**—Display SYSV semaphores recorded for cleanup.

- `sysvshm`—Display SYSV shared memory segments recorded for cleanup.
- `ui`—Display tracing messages for UI operational commands.

Chapter 9

Configuration Mode Commands for SDK Applications

Configuration mode commands are commands you enter in configuration mode that perform general configuration functions such as copying, navigating, and managing configuration files. This chapter covers the following configuration mode command topics:

- Displaying the Package Name for JUNOS SDK Application Configurations on page 37
- Displaying and Deleting the Configuration for JUNOS SDK Applications on page 38
- How the extension show Command Matches Package Names on page 38
- Using the extension show Command to Display SDK Configuration by Package Name on page 39
- Using the extension delete Command to Delete SDK Configuration by Package Name on page 40

Displaying the Package Name for JUNOS SDK Application Configurations

To determine if a configuration has settings contributed by an SDK application package, use the following configuration mode command:

```
user@host# show <statement-path> | display detail
```

This command displays the configuration schema as piped through to the **display detail** command. The **show** command displays the entire user-defined configuration unless it is limited by the optional **statement-path** variable to that branch of the configuration schema. The **show | display detail** command displays the characteristics, descriptions, and constraints of each configuration statement in the JUNOS Software configuration schema using comment lines.

Generally, the information displayed is help strings and the permission bits required to add or modify the configuration statement. For configuration statements that are defined by an SDK application package, the name of the package that defines the statement is also displayed. If a JUNOS statement is redefined by an SDK application package, the package name is listed. However, if a configuration statement is defined by the native JUNOS Software only, no package name is displayed.

Displaying and Deleting the Configuration for JUNOS SDK Applications

To display or delete the configuration for a specific SDK application package, use the `extension package-name (show | delete)` configuration mode command:

```
user@host# extension package-name (show | delete) <section>
```

The `extension` command filters for SDK application configuration statements based on the package named in the `extension package-name (show | delete)` command starting at the top of the configuration hierarchy, or, if you use the `section` option, starting at the hierarchy level (statement path) specified by `section`.



NOTE: Remove all the soon-to-be invalid configurations before removing the SDK application package.

Before removing SDK application packages, use the `extension package-name show` command to display the entire configuration contributed by the named package. Then use the `extension package-name delete` command to remove all configuration statements for that package.

Use the `extension show` and `extension delete` commands to select for, or filter, configurations differently, as explained in the following topics.

How the `extension show` Command Matches Package Names

Without the `extension package-name` filter, the `show` command displays the entire user-defined configuration hierarchy. The `extension package-name show` command, however, displays only the configuration statements contributed by packages whose names match those in the command. The subset of matches includes packages whose package names exactly match the value of `package-name` as well as those whose names have the same root but may have longer names, similar to a wildcard situation. The output displays the settings relating to all matches. The following example shows simplified output illustrating how the `extension show` filter works.

Suppose a router has `packageA`, `packageB`, and `packageAB` installed. When you issue the `show` command, you see the following output:

```
user@host# show
system {
  packageA {
    ....
  }
  packageB {
    ....
  }
  packageAB {
    ....
  }
}
```

If you issue the command `extension packageA show`, you see a subset of the previous `show` command output. The output displays settings for packages with names that not only exactly match `packageA` but also have roots that match (in this case, `packageAB`):

```
user@host# extension packageA show
system {
  packageA {
    ....
  }
  packageAB {
    ....
  }
}
```

Using the extension show Command to Display SDK Configuration by Package Name

The extension `package-name show` command selects configurations contributed by and leading to the package named in the command.

In the following example, only the `sdk-backup-server` statement at the `[edit system radius-server 10.1.1.1]` hierarchy level and the `sdk-proto1` statement at the `[edit system protocols]` hierarchy level are contributed by the SDK application package `sdk-pkg1`:

```
user@host# show
system {
  radius-server {
    10.1.1.1 {
      timeout 10;
      sdk-backup-server 10.1.1.2; # Contributed by sdk-pkg1
    }
  }
}
protocols {
  ospf {
    ....
  }
  sdk-proto1 { # Contributed by sdk-pkg1
    ....
  }
}
```

Following is the output from the extension `sdk-pkg1 show` command:

```
user@host# extension sdk-pkg1 show
system {
  radius-server {
    10.1.1.1 {
      sdk-backup-server 10.1.1.2;
    }
  }
}
protocols {
```

```

    sdk-proto1 {
        ....
    }
}

```

The extension `sdk-pkg1 show` command filters out the `timeout` and `ospf` commands from the displayed output, which were not contributed by the `sdk-pkg1` package.

Using the extension `delete` Command to Delete SDK Configuration by Package Name

The extension `package-name delete` command treats the value of *package-name* as a literal. It deletes only the settings contributed by the package whose package name exactly matches the given value. Using this command, you can delete all user-defined configuration statements related to the named package.



NOTE: A configuration schema defined in any native JUNOS package is never deleted by the removal of any third-party package.

Continuing with the preceding example, notice how the configuration changes when the extension `sdk-pkg1 delete` command is used:

```

user@host# extension sdk-pkg1 delete
[edit]
user@host# show
system {
    radius-server {
        10.1.1.1 {
            timeout 10;
        }
    }
}
protocols {
    ospf {
        ....
    }
}

```

You could accomplish the same thing by issuing the following two commands:

- `delete system radius-server 10.1.1.1 sdk-backup-server`
- `delete protocols sdk-proto1`

Part 2

Configuration Summaries and Command Reference

- Summary of SDK Configuration Mode Commands on page 43
- Summary of SDK Configuration Statements on page 47
- SDK Operational Command Reference on page 63

Chapter 10

Summary of SDK Configuration Mode Commands

The configuration mode commands described in this section are specific to the configuration of SDK applications either in whole or in part. The commands are listed in alphabetical order.

extension package-name (show | delete)

Syntax extension *package-name* (show | delete) <*section*>

Release Information Command introduced in JUNOS Release 8.5.

Description Manage configurations that are contributed by SDK application packages. Use **show** to display user-defined configuration contributed by the named package. If you specify **delete**, all subordinate statements and identifiers contained within the specified statement path (see the *section* option) are deleted as well.



NOTE: A configuration defined in any of the native JUNOS packages is never deleted by the extension *package-name* delete command.

Options *package-name*—Name of the SDK application package.

section—(Optional) Statement path from which you want the command to act.

Required Privilege Level configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.

show | display detail (SDK)

Syntax	show <hierarchy-level> display detail
Release Information	Support for JUNOS SDK applications added in JUNOS Release 8.5.
Description	Display additional information about the configuration. The additional information generally includes the help string that explains each configuration statement and the permission bits required to add and modify the configuration statement. If a statement is contributed by an SDK application package, the name of the package is given with the output field ## package: package-name .
Options	<i>hierarchy-level</i> —(Optional) Statement hierarchy path for which you want the configuration displayed.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.

Chapter 11

Summary of SDK Configuration Statements

This chapter provides a reference for each of the SDK configuration statements. The statements are organized alphabetically.

extension-provider

Syntax

```
extension-provider {
  control-cores control-number;
  data-cores data-number;
  data-flow-affinity;
  forwarding-db-size size;
  object-cache-size size;
  package package-name;
  policy-db-size size;
  wired-process-mem-size size;
  syslog {
    facility severity;
  }
}
```

Hierarchy Level [edit chassis fpc *slot-number* pic *slot-number* adaptive-services service-package]

Release Information Statement introduced in JUNOS Release 9.0.
 object-cache-size option introduced in JUNOS Release 9.1.
 forwarding-db-size, syslog, facility, and severity options introduced in JUNOS Release 9.2.
 policy-db-size and wired-process-mem-size options introduced in JUNOS Release 9.3.

Description Configure an SDK application on a PIC.

Options control-cores *control-number*—Number of cores you want to specify as control cores, which are cores dedicated to run control functionality (application control). At least one core must be a control core.

Range: 1 through 8

data-cores *data-number*—Number of cores you want to specify as data cores, which are cores dedicated to processing data for the application (application data).

data-flow-affinity—Enable flow affinity distribution for packets over data CPUs on the PIC. Once enabled, the default behavior distributing data packets changes from a round-robin distribution to a flow affinity distribution based on a hash distribution. Adding or deleting this statement will cause the PIC to reboot.

Range: 0 through 7 (recommended minimum is 5).

forwarding-db-size *size*—Size of the forwarding database (FDB), in megabytes (MB).

Range: 0 through 1280

object-cache-size *size*—Size of the object cache, in MB. Only values in increments of 128 MB are allowed.

Range: For Multiservices 100 PIC, 128 through 512. If the PIC is configured with a wired-process-mem-size of 512, the maximum object-cache-size is 128 MB.

Range: For Multiservices 400 PIC, 128 through 1280. If the PIC is configured with a wired-process-mem-size of 512, the maximum object-cache-size is 512 MB.

package *package-name*—Name of the SDK application package to be installed on the PIC. Up to eight packages can be installed on a PIC.

`policy-db-size size`—Size of the policy database, in MB.

Range: Range: 0 through 1280

`wired-process-mem-size size`—Size of the wired process memory, in MB.

Range: For Multiservices 100 PIC and Multiservices 400 PIC, 0 through 512

The remaining statements are explained separately in this collection of topics.

Usage Guidelines See “Configuring Object Cache, Policy Database, and Forwarding Database” on page 6.

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

Related Topics `syslog`

extension-service

Syntax `extension-service service-name {
 provider-specific rules;
}`

Hierarchy Level `[edit forwarding-options sampling output]`,
`[edit services service-set service-set-name]`

Release Information Statement introduced in JUNOS Release 9.0.

Description Define a service set or traffic monitoring for JUNOS SDK applications using application-specific configuration guidelines.

Options `provider-specific rules`—Provider-specific subhierarchy for services and service sets. See the application-specific documentation for details.

Usage Guidelines See “SDK Services Configuration Guidelines” on page 19.

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

extensions

Syntax	<pre>extensions { providers { <i>provider-id</i>; } }</pre>
Hierarchy Level	[edit system]
Release Information	Statement introduced in JUNOS Release 9.0.
Description	Turn on the SDK service process (ssd) and configure the provider-id statement to enable SDK application packages to be deployed and run on the router.
Options	providers <i>provider-id</i> —Provider ID for the SDK application package. See the application-specific documentation.
Usage Guidelines	See “Enabling the SSD and SDK Application Deployment” on page 4.
Required Privilege Level	admin—To view this statement in the configuration. admin-control—To add this statement to the configuration.

process-monitor

Syntax

```
process-monitor {
  disable;
  traceoptions {
    file;
    flag flag;
    level level;
    no-remote-trace;
  }
}
```

Hierarchy Level [edit system processes]

Release Information Statement introduced in JUNOS Release 9.0.

Description Configure tracing options for the process health monitor (pmond).

Options disable—Disable the health monitor.

The remaining statements are explained separately in this collection of topics.

Usage Guidelines See “Tracing Process Monitoring Operations for JUNOS SDK Applications” on page 33.

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

Related Topics traceoptions (Process Monitor)

resource-cleanup

Syntax

```
resource-cleanup {
  disable;
  traceoptions {
    file;
    flag flag;
    level level;
    no-remote-trace;
  }
}
```

Hierarchy Level [edit system processes]

Release Information Statement introduced in JUNOS Release 9.3.

Description Selectively turn on or off the debugging of trace messages for the resource cleanup process.

Options **disable**—Disable the resource cleanup process.

The remaining statements are explained separately in this collection of topics.

Usage Guidelines See “Tracing System Resource Cleanup Operations for JUNOS SDK Applications” on page 34.

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

Related Topics traceoptions (Resource Cleanup)

resource-limits

Syntax

```
resource-limits {
  package package-name {
    resources {
      cpu {
        priority number;
        time seconds;
      }
      file {
        core-size bytes;
        open number;
        size bytes;
      }
      memory {
        data-size bytes;
        locked-in bytes;
        resident-set-size bytes;
        socket-buffers bytes;
        stack-size bytes;
      }
    }
  }
  process process-ui-name {
    resources {
      cpu {
        priority number;
        time seconds;
      }
      file {
        core-size bytes;
        open number;
        size bytes;
      }
      memory {
        data-size bytes;
        locked-in bytes;
        resident-set-size bytes;
        socket-buffers bytes;
        stack-size bytes;
      }
    }
  }
}
```

Hierarchy Level [edit system extensions]

Release Information Statement introduced in JUNOS Release 9.6.

Description Set resource limits for JUNOS SDK applications using the command-line interface (CLI). You can set limits for all SDK applications listed in the SDK application package's manifest file or define limits for individual processes in the package.

Options `package` *package-name*—Name of the JUNOS SDK application package.

`process` *process-ui-name*—Name of the process.

The remainder of the statements are explained separately.

Usage Guidelines See “Configuring Resource Limits” on page 14.

Required Privilege Level `admin`—To view this statement in the configuration.
 `admin`—To view this statement in the configuration.

Related Topics `resources`

resources

Syntax

```
resources {
  cpu {
    priority number;
    time seconds;
  }
  file {
    core-size bytes;
    open number;
    size bytes;
  }
  memory {
    data-size bytes;
    locked-in bytes;
    resident-set-size bytes;
    socket-buffers bytes;
    stack-size bytes;
  }
}
```

Hierarchy Level [edit system extensions resource-limits package package-name],
[edit system extensions resource-limits process process-ui-name]

Release Information Statement introduced in JUNOS Release 9.6.

Description Set resource limits for JUNOS SDK applications.

Options *bytes*—Maximum size of each file, in kilobytes (KB) or megabytes (MB).
Syntax: Where x is some number, use xk to specify KB or xm to specify MB.

core-size bytes—Maximum size of a core file that can be created.

data-size bytes—Maximum size of the data segment.

file—File system resources.

file-size bytes—Maximum size of a file that can be created.

locked-in bytes—Maximum number of bytes that can be locked into memory.

memory—Memory resources.

open number—Maximum number of simultaneous open files.

priority number—Highest priority number (nice level) at which the process can run.

resident-set-size bytes—Maximum amount of private or shared memory at any given moment.

socket-buffers bytes—Maximum amount of physical memory that may be dedicated to the socket buffers.

`stack-size bytes`—Maximum size of the stack segment.

`time seconds`—Maximum amount of CPU time that can be accumulated.

Usage Guidelines See Setting Administrator-Defined Resource Limits for SDK Applications.

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

Related Topics `resource-limits`

service-order

Syntax `service-order {
 forward-flow [service-name1 service-name2];
 reverse-flow [service-name1 service-name2];
 }`

Hierarchy Level `[edit services service-set service-set-name]`

Release Information Statement introduced in JUNOS Release 9.3.

Description Define the order of services in service set to be applied to traffic coming to the PIC.

Options `forward-flow`—Order of services in service set to be applied in forward flow.

`reverse-flow`—Order of services in service set to be applied in reverse flow. If you want the order to be the reverse of that specified for forward flow, this is optional. However, if, for example, you want the order to be the same regardless of direction of flow, you must include this statement.

Usage Guidelines See “Configuring the Service Order for JUNOS SDK Service Sets” on page 20.

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

Related Topics `extension-service`

syslog

Syntax	syslog { <i>facility severity</i> ; }
Hierarchy Level	[edit chassis fpc <i>slot-number</i> pic <i>slot-number</i> adaptive-services service-package extension-provider]
Release Information	Statement introduced in JUNOS Release 9.2.
Description	Enable PIC system logging to record or view system log messages on a specific PIC. The system log information is passed to the kernel for logging in the /var/log/messages directory.
Options	<p><i>facility</i>—Group of messages that are either generated by the same software process or concern a similar condition or activity. Possible values include the following: daemon, external, kernel, and pfe.</p> <p><i>severity</i>—Classification of effect on functioning. Possible values are the same as for the native JUNOS Software and include the following:</p> <ul style="list-style-type: none"> ■ any—Include all severity levels. ■ none—Disable logging of the associated facility to a destination. ■ emergency—System panic or other condition that causes the routing platform to stop functioning. ■ alert—Conditions that require immediate correction, such as a corrupted system database. ■ critical—Critical conditions, such as hard errors. ■ error—Error conditions that generally have less serious consequences than errors in the emergency, alert, and critical levels. ■ warning—Conditions that warrant monitoring. ■ notice—Conditions that are not errors but might warrant special handling. ■ info—Events or nonerror conditions of interest.
Usage Guidelines	See “Configuring System Log Messages” on page 9.
Required Privilege Level	<p>system—To view this statement in the configuration.</p> <p>system-control—To add this statement to the configuration.</p>
Related Topics	extension-provider

traceoptions

See the following sections:

- [traceoptions \(Process Monitor\) on page 59](#)
- [traceoptions \(Resource Cleanup\) on page 61](#)

traceoptions (Process Monitor)

Syntax

```

traceoptions {
    file filename {
        files number;
        match regex;
        size size;
        world-readable | no-world-readable;
    }
    flag flag;
    level level;
    no-remote-trace;
}

```

Hierarchy Level [edit system processes process-monitor]

Release Information Statement introduced in JUNOS Release 9.0.

Description Enable tracing options for the process health monitor (pmond).

Options file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name within quotation marks. To include the **file** statement, you must specify a filename.

files *number*—(Optional) Maximum number of trace files. When a trace file named **trace-file** reaches its maximum size, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

flag *flag*—Specify which tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements. You can include the following flags:

- all—Enable all trace options flags.
- events—Trace process state change and cleanup events.
- gencfg—Trace GENCFG blobs recorded for cleanup.
- module—Trace module code.
- sysvsem—Trace SYSV semaphores recorded for cleanup.
- sysvshm—Trace SYSV shared memory segments recorded for cleanup.
- tracking—Trace tracking code.
- ui—Trace user interface operations.

level *level*—Specify the level of debugging output:

- all—Match all levels.
- error—Match error conditions.

- **info**—Match informational messages.
- **notice**—Match conditions that warrant special handling (but are not errors).
- **verbose**—Match verbose messages.
- **warning**—Match warning messages.

match *regex*—(Optional) Refine the output to include lines that contain the regular expression.

no-remote-trace—Disable remote tracing.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). If you specify a maximum file size, you also must specify a maximum number of trace files with the **files *number*** option.

Syntax: **xk** to specify KB, **xm** to specify MB, or **xg** to specify GB

Range: 10 KB through 1 GB

Default: 128 KB

world-readable | no-world-readable—(Optional). Grant all users permission to read log files, or restrict the permission only to the **root** user and users who have the **JUNOS maintenance** permission.

Usage Guidelines See “Tracing Process Monitoring Operations for JUNOS SDK Applications” on page 33.

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

traceoptions (Resource Cleanup)

Syntax

```

traceoptions {
    file filename {
        files number;
        match regex;
        size size;
        world-readable | no-world-readable;
    }
    flag flag;
    level level;
    no-remote-trace;
}

```

Hierarchy Level [edit system processes resource-cleanup]

Release Information Statement introduced in JUNOS Release 9.3.

Description Enable debugging tracing for resource cleanup process.

Options **file *filename***—Name of the file to receive the output of the tracing operation. Enclose the name within quotation marks. To include the **file** statement, you must specify a filename.

files *number*—(Optional) Maximum number of trace files. When a trace file named **trace-file** reaches its maximum size, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

flag *flag*—Specify which tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements. You can include the following flags:

- **all**—Enable all trace options flags.
- **events**—Trace process state change and cleanup events.
- **gencfg**—Trace GENCFG blobs recorded for cleanup.
- **module**—Trace module code.
- **sysvsem**—Trace SYSV semaphores recorded for cleanup.
- **sysvshm**—Trace SYSV shared memory segments recorded for cleanup.
- **tracking**—Trace tracking code.
- **ui**—Trace user interface operations.

level *level*—Specify the level of debugging output:

- **all**—Match all levels.
- **error**—Match error conditions.

- **info**—Match informational messages.
- **notice**—Match conditions that warrant special handling (but are not errors).
- **verbose**—Match verbose messages.
- **warning**—Match warning messages.

match *regex*—(Optional) Refine the output to include lines that contain the regular expression.

no-remote-trace—Disable remote tracing.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). If you specify a maximum file size, you also must specify a maximum number of trace files with the **files *number*** option.

Syntax: **xk** to specify KB, **xm** to specify MB, or **xg** to specify GB

Range: 10 KB through 1 GB

Default: 128 KB

world-readable | no-world-readable—(Optional). Grant all users permission to read log files, or restrict the permission only to the **root** user and users who have the JUNOS **maintenance** permission.

Usage Guidelines See “Tracing System Resource Cleanup Operations for JUNOS SDK Applications” on page 34.

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

Related Topics **resource-cleanup**

Chapter 12

SDK Operational Command Reference

Operational mode commands show you the current status of the router interfaces, chassis, protocols, and system information and are used to monitor and troubleshoot configurations.

The operational commands described here are relevant to the configuration of SDK applications. For JUNOS operational commands, see the JUNOS command references.

clear interfaces statistics (SDK)

Syntax	clear interfaces statistics (all <i>interface-name</i>)
Release Information	For routers running JUNOS SDK applications, support for rms interfaces added in JUNOS Release 9.5.
Description	Clear interface statistics. When the show interfaces statistics command is next issued, the output will show the packets received since the last time the clear interfaces statistics command was issued. However, after a graceful Routing Engine switchover (GRES), issuing the show interfaces statistics command will show culmulative statistics.
Options	<p>all—Clear statistics on all interfaces.</p> <p><i>interface-name</i>—Clear statistics on the named interface.</p>
Required Privilege Level	clear
Output Fields	There is no output for this command.
Sample Output	user@host> clear interfaces statistics rms0

clear services stateful-firewall flows (SDK)

Syntax	clear services stateful-firewall flows <interface interface-name>
Release Information	For routers running JUNOS SDK applications, support for ms- interfaces added in JUNOS Release 9.5.
Description	<p>Remove established flows from the flow table.</p> <p>For this commands's usage in other contexts, see the <i>JUNOS System Basics and Services Command Reference</i>.</p>
Options	<p>none—Clear all stateful firewall flows.</p> <p>interface interface-name—(Optional) Clear stateful firewall flows for the named interface.</p>
Required Privilege Level	clear
Output Fields	There is no output for this command.
Sample Output	user@host> clear services stateful-firewall flows

request interface (revert | switchover) (SDK)

Syntax	<code>request interface (revert switchover) rmsnumber</code>
Release Information	For routers running JUNOS SDK applications, support for rms interfaces added in JUNOS Release 9.5.
Description	<p>Manually revert to the primary interface or switch over to the secondary interface. After the switchover, it is recommended to take the old active PIC offline so it comes up fresh as a new backup.</p> <p>For this command's usage in other contexts, see the <i>JUNOS Interfaces Command Reference</i>.</p>
Options	<p>revert—Restore active processing to the primary Multiservices PIC.</p> <p>switchover—Transfer active processing to the secondary (backup) Multiservices PIC.</p> <p>rmsnumber—Number identifying the rms interface.</p>
Required Privilege Level	view
Output Fields	<p>One of two outputs are possible:</p> <ul style="list-style-type: none"> ■ request succeeded ■ error: command invalid for current state
request interface switchover	<pre>user@host> request interface switchover rms0 request succeeded</pre>
request interface revert	<pre>user@host> request interface revert rms0 request succeeded</pre>
request interface revert repeated	<pre>user@host> request interface revert rms0 error: command invalid for current state</pre>

show chassis pic (SDK)

Syntax	show chassis pic fpc-slot <i>slot-number</i> pic-slot <i>slot-number</i>
Release Information	Support for JUNOS SDK applications added in JUNOS Release 8.4.
Description	For PICs with SDK application packages installed, the show chassis pic command displays Extension-provider in the Package field. For more information about this command, see the <i>JUNOS System Basics and Services Command Reference</i> .
Options	See the show chassis pic command in the <i>JUNOS System Basics and Services Command Reference</i> .
Required Privilege Level	view
List of Sample Output	show chassis pic on page 67
Output Fields	For a description of the output fields, see the output fields table for the show chassis pic command in the <i>JUNOS System Basics and Services Command Reference</i> .
show chassis pic	<pre> user@host> show chassis pic pic-slot 1 fpc-slot 1 FPC slot 1, PIC slot 1 information: Type MultiServices 100 State Online PIC version 1.5 Uptime 5 hours, 59 minutes, 34 seconds Package Extension-provider </pre>

show extension-provider system connections

Syntax	show extension-provider system connections <extensive> <inet inet6> <interface> <show-routing-instances>
Release Information	Command introduced in JUNOS Release 9.1.
Description	Show connection activity on the extension provider PIC. This command functions similarly to <code>show system connections</code> command.
Options	<p><code>extensive</code>—(Optional) Display exhaustive system process information.</p> <p><code>inet inet6</code>—(Optional) Display IPv4 connections or IPv6 connections, respectively.</p> <p><code>interface</code>—(Optional) Display the name of the extension provider interface.</p> <p><code>show-routing-instances</code>—(Optional) Display routing instances.</p>
Required Privilege Level	view
List of Sample Output	<p>show extension-provider system connections on page 68</p> <p>show extension-provider system connections extensive interface on page 69</p> <p>show extension-provider system connections inet on page 69</p> <p>show extension-provider system connections inet6 on page 69</p> <p>show extension-provider system connections interface on page 69</p> <p>show extension-provider system connections show-routing-instances on page 70</p> <p>show extension-provider system connections show-routing-instances interface inet6 on page 70</p>
Output Fields	For a description of the output fields, see the output fields table for the <code>show system connections</code> command in the <i>JUNOS System Basics and Services Command Reference</i> .

```

user@host> show extension-provider system connections
Interface: ms-0/0/0
Active Internet connections (including servers)
Proto Recv-Q Send-Q Local Address          Foreign Address         (state)
tcp4      0      0 20.0.0.16.32004        20.0.0.1.64292         ESTABLISHED
tcp4      0      0 20.0.0.16.3000         20.0.0.1.58036         ESTABLISHED
tcp4      0      0 20.0.0.16.59517        20.0.0.1.3000          ESTABLISHED
tcp4      0      0 *.3000                 *.*                     LISTEN
tcp4      0      0 *.32004                 *.*                     LISTEN
tcp4    66312  0 20.0.0.16.59592        20.0.0.1.32003         ESTABLISHED
tcp4      0      0 *.23                    *.*                     LISTEN
tcp4      0      0 *.33005                 *.*                     LISTEN
tcp4      0      0 128.0.1.16.49900       128.0.0.1.6234         ESTABLISHED
udp4      0      0 *.842                   *.*                     LISTEN
udp4      0      0 127.0.0.1.123          *.*                     LISTEN
udp4      0      0 *.123                   *.*                     LISTEN
udp46     0      0 *.514                   *.*                     LISTEN
udp4      0      0 *.514                   *.*                     LISTEN
Interface: ms-0/2/0
Active Internet connections (including servers)

```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	(state)
tcp4	0	0	20.0.0.18.32004	20.0.0.1.62402	ESTABLISHED
tcp4	0	0	*.33005	*.*	LISTEN
tcp4	0	0	128.0.3.16.59592	128.0.0.1.6234	ESTABLISHED
tcp4	0	0	*.23	*.*	LISTEN
tcp4	0	0	*.32004	*.*	LISTEN
tcp4	0	0	20.0.0.18.49900	20.0.0.1.32003	ESTABLISHED
udp4	0	0	*.789	*.*	
udp4	0	0	127.0.0.1.123	*.*	
udp4	0	0	*.123	*.*	
udp46	0	0	*.514	*.*	
udp4	0	0	*.514	*.*	

show extension-provider system connections extensive interface user@host> **show extension-provider system connections extensive interface ms-0/2/0**
Interface: ms-0/2/0
Active Internet connections (including servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	(state)
tcp4	0	0	20.0.0.18.32004	20.0.0.1.49792	ESTABLISHED
sndsbcc:		0	sndsmbcnt:	0	sdsbmbmax: 265248
sndsblowat:		2048	sndsbiwat:	33156	
rcvsbcc:		0	rcvsbmbcnt:	0	rcvsbmbmax: 530496
rcvsblowat:		1	rcvsbiwat:	66312	
proc id:		1	proc name:		
iss:	4025626166		sndup:	4025626167	
snduna:	4025626167		sndnxt:	4025626167	sndwnd: 66312
sndmax:	4025626167		sndcwnd:	131070	sndssthresh: 1073725440
irs:	3544420903		rcvup:	3544420904	
rcvnxt:	3544421176		rcvadv:	3544487488	rcvwnd: 66312
rtt:	0		srtt:	64	rttv: 16
rxtcur:	1200		rxtshift:	0	rtseq: 0
rttmin:	1000		mss:	1228	
Flags:	REQ_SCALE RCVD_SCALE REQ_TSTMP RCVD_TSTMP SACK_PERMIT [0x20003e0]				

...

show extension-provider system connections inet The output for the show extension-provider system connections inet command is identical to that for the show extension-provider system connections command. For sample output, see show extension-provider system connections on page 68.

show extension-provider system connections inet6 user@host> **show extension-provider system connections inet6**
Interface: ms-0/0/0
Active Internet connections (including servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	(state)
udp6	0	0	*.123	*.*	
udp46	0	0	*.514	*.*	

Interface: ms-0/2/0
Active Internet connections (including servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	(state)
udp6	0	0	*.123	*.*	
udp46	0	0	*.514	*.*	

show extension-provider system connections interface user@host> **show extension-provider system connections interface ms-0/2/0**
Interface: ms-0/2/0
Active Internet connections (including servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	(state)
tcp4	0	0	20.0.0.18.32004	20.0.0.1.59521	ESTABLISHED
tcp4	0	0	*.33005	*.*	LISTEN
tcp4	0	0	128.0.3.16.51534	128.0.0.1.6234	ESTABLISHED
tcp4	0	0	*.23	*.*	LISTEN

```

tcp4      0      0 *.32004          *.*             LISTEN
tcp4      0      0 20.0.0.18.61044  20.0.0.1.32003 ESTABLISHED
udp4      0      0 *.914            *.*
udp4      0      0 127.0.0.1.123    *.*
udp4      0      0 *.123            *.*
udp46     0      0 *.514            *.*
udp4      0      0 *.514            *.*

```

show extension-provider system connections show-routing-instances

```

user@host> show extension-provider system connections show-routing-instances

Interface: ms-0/0/0
Active Internet connections (including servers) (including routing-instances)
Proto Recv-Q Send-Q Local Address Foreign Address Routing Instance (state)
tcp4    0      0 20.0.0.16.32004  20.0.0.1.52590 __juniper_private2__ ESTABLISHED
tcp4    0      0 20.0.0.16.3000   20.0.0.1.58036 __juniper_private2__ ESTABLISHED
tcp4    0      0 20.0.0.16.59517  20.0.0.1.3000   __juniper_private2__ ESTABLISHED
tcp4    0      0 *.3000           *.*             __juniper_private2__ LISTEN
tcp4    0      0 *.32004          *.*             __juniper_private2__ LISTEN
tcp4  66312    0 20.0.0.16.59592  20.0.0.1.32003 __juniper_private2__ ESTABLISHED
tcp4    0      0 *.23             *.*             __juniper_private1__ LISTEN
tcp4    0      0 *.33005          *.*             __juniper_private2__ LISTEN
tcp4    0      0 128.0.1.16.49900 128.0.0.1.6234 __juniper_private1__ ESTABLISHED
udp4    0      0 *.842            *.*             __juniper_private1__
udp4    0      0 127.0.0.1.123    *.*             default
udp4    0      0 *.123            *.*             __juniper_private1__
udp46   0      0 *.514            *.*             default
udp4    0      0 *.514            *.*             __juniper_private1__

Interface: ms-0/2/0
Active Internet connections (including servers) (including routing-instances)
Proto Recv-Q Send-Q Local Address Foreign Address Routing Instance (state)
tcp4    0      0 20.0.0.18.32004  20.0.0.1.54602 __juniper_private2__ ESTABLISHED
tcp4    0      0 *.33005          *.*             __juniper_private2__ LISTEN
tcp4    0      0 128.0.3.16.59592 128.0.0.1.6234 __juniper_private1__ ESTABLISHED
tcp4    0      0 *.23             *.*             __juniper_private1__ LISTEN
tcp4    0      0 *.32004          *.*             __juniper_private2__ LISTEN
tcp4    0      0 20.0.0.18.49900  20.0.0.1.32003 __juniper_private2__ ESTABLISHED
udp4    0      0 *.789            *.*             __juniper_private1__
udp4    0      0 127.0.0.1.123    *.*             default
udp4    0      0 *.123            *.*             __juniper_private1__
udp46   0      0 *.514            *.*             default
udp4    0      0 *.514            *.*             __juniper_private1__

```

show extension-provider system connections show-routing-instances interface ms-0/0/0 inet6

```

user@host> show extension-provider system connections show-routing-instances
interface ms-0/0/0 inet6

Interface: ms-0/0/0
Active Internet connections (including servers) (including routing-instances)
Proto Recv-Q Send-Q Local Address Foreign Address Routing Instance (state)
udp6      0      0 *.123          *.*             default
udp46     0      0 *.514          *.*             default

```

show extension-provider system packages

Syntax	show extension-provider system packages <detail> <interface>
Release Information	Command introduced in JUNOS Release 9.1.
Description	Show packages loaded on the extension provider PIC. This command functions similarly to <code>show system software</code> command.
Options	detail—(Optional) Display detailed output. interface—(Optional) Display the name of the extension provider interface.
Required Privilege Level	view
List of Sample Output	show extension-provider system packages on page 71 show extension-provider system packages detail on page 71 show extension-provider system packages interface on page 72
Output Fields	Output field descriptions to be provided.
show extension-provider system packages	<pre> user@host> show extension-provider system packages Interface: ms-0/0/0 jmpsdk JUNOS MPSDK Base OS boot [9.2R1.3] jnx-flow-data-pic JUNOS SDK JNX-FLOW Dataplane Component [9.2I20080801_1059] Interface: ms-0/2/0 jmpsdk JUNOS MPSDK Base OS boot [9.2R1.3] </pre>
show extension-provider system packages detail	<pre> user@host> show extension-provider system packages detail Interface: ms-0/0/0 Information for jmpsdk: Comment: JUNOS MPSDK Base OS boot [9.2R1.3] Description: JUNOS MPSDK Base OS Copyright (c) 1996-2008, Juniper Networks, Inc. All rights reserved. Software version: 9.2R1.3 This package contains the MPSDK base operating system components. Information for jnx-flow-data-pic: Comment: JUNOS SDK JNX-FLOW Dataplane Component [9.2I20080801_1059] Description: JUNOS SDK JNX-FLOW Data Component Package Copyright (c) 1996-2008, Juniper Networks, Inc. All rights reserved. Software version: 9.2I20080801_1059 This package contains SDK JNX-FLOW dataplane component Interface: ms-0/2/0 Information for jmpsdk: Comment: JUNOS MPSDK Base OS boot [9.2R1.3] Description: JUNOS MPSDK Base OS </pre>

Copyright (c) 1996-2008, Juniper Networks, Inc.
All rights reserved.
Software version: 9.2R1.3
This package contains the MPSDK base operating system components.

show extension-provider	user@host> show extension-provider system packages interface ms-0/2/0
system packages	Interface: ms-0/2/0
interface	jmpsdk JUNOS MPSDK Base OS boot [9.2R1.3]

show extension-provider system processes

Syntax	show extension-provider system processes <brief detail extensive> <interface> <wide>
Release Information	Command introduced in JUNOS Release 9.1.
Description	Show system process table on the extension provider PIC.
Options	brief detail extensive—(Optional) Display the specified level of output. interface—(Optional) Name of the extension provider interface. wide—(Optional) Display information even if it is wider than 80 columns.
Required Privilege Level	view
List of Sample Output	show extension-provider system processes on page 73 show extension-provider system processes brief on page 74 show extension-provider system processes detail on page 74 show extension-provider system processes extensive on page 75 show extension-provider system processes interface on page 75 show extension-provider system processes wide on page 76
Output Fields	For a description of the output fields, see the output fields table for the show system processes command in the <i>JUNOS System Basics and Services Command Reference</i> .

show extension-provider system processes user@host> **show extension-provider system processes**

```
Interface: ms-0/0/0
  PID  TT  STAT      TIME  COMMAND
   0  ??  WLS      0:00.00 [swapper]
   1  ??  SLs      0:00.75 /sbin/init --
   2  ??  DL       0:05.91 [g_event]
   3  ??  DL       0:03.96 [g_up]
   4  ??  DL       0:04.24 [g_down]
   5  ??  DL       0:00.00 [kqueue taskq]
   6  ??  DL       0:00.00 [thread taskq]
   9  ??  DL       0:00.15 [pagedaemon]
  10  ??  DL       0:00.00 [ktrace]
  11  ??  RL       0:00.00 [idle: cpu31]
  12  ??  RL       0:00.00 [idle: cpu30]
  13  ??  RL       0:00.00 [idle: cpu29]
  14  ??  RL       0:00.13 [idle: cpu28]
  15  ??  RL       0:00.00 [idle: cpu27]
  16  ??  RL       0:00.00 [idle: cpu26]
  17  ??  RL       0:00.00 [idle: cpu25]
  18  ??  RL       0:00.13 [idle: cpu24]
  19  ??  RL       0:00.00 [idle: cpu23]
  20  ??  RL       0:00.00 [idle: cpu22]
  21  ??  RL       0:00.00 [idle: cpu21]
  22  ??  RL       0:00.13 [idle: cpu20]
  23  ??  RL       0:00.00 [idle: cpu19]
  24  ??  RL       0:00.00 [idle: cpu18]
```

```

25 ?? RL 0:00.00 [idle: cpu17]
26 ?? RL 0:00.13 [idle: cpu16]
27 ?? RL 0:00.00 [idle: cpu15]
28 ?? RL 0:00.00 [idle: cpu14]
29 ?? RL 0:29.15 [idle: cpu13]
30 ?? RL 443:15.66 [idle: cpu12]
31 ?? RL 0:29.15 [idle: cpu11]
32 ?? RL 0:29.14 [idle: cpu10]
. . .

```

show extension-provider system processes brief

```
user@host> show extension-provider system processes brief
```

```
Interface: ms-0/0/0
```

```
last pid: 20238; load averages: 11.64, 11.71, 11.74 up 0+07:24:28 22:23:18
91 processes: 45 running, 34 sleeping, 12 waiting
```

```
Mem: 8924K Active, 1768K Inact, 152M Wired, 156K Cache, 77M Buf, 200M Free
```

```
Swap:
```

```
Interface: ms-0/2/0
```

```
last pid: 13025; load averages: 4.16, 4.08, 4.02 up 0+04:43:20 22:23:18
88 processes: 37 running, 32 sleeping, 19 waiting
```

```
Mem: 6488K Active, 1212K Inact, 156M Wired, 24K Cache, 75M Buf, 460M Free
```

```
Swap:
```

show extension-provider system processes detail

```
user@host> show extension-provider system processes detail
```

```
Interface: ms-0/0/0
```

PID	UID	PPID	CPU	PRI	NI	RSS	WCHAN	STARTED	TT	STAT	TIME	COMMAND
20	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
21	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
22	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.13	[idle: cp
28	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
15	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
19	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
11	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
12	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
13	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
16	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
17	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
18	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.13	[idle: cp
26	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.13	[idle: cp
68	0	0	0	-16	0	8	-	2:59PM	??	RL	0:00.00	[pot: cpu
70	0	0	0	-16	0	8	-	2:59PM	??	RL	0:00.00	[poller:
14	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.13	[idle: cp
24	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
25	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
27	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
69	0	0	0	-16	0	8	-	2:59PM	??	RL	0:00.00	[poller:
23	0	0	0	171	0	8	-	2:59PM	??	RL	0:00.00	[idle: cp
30	0	0	0	171	0	8	-	2:59PM	??	RL	444:58.30	[idle: cp
259	0	1	0	139	0	840	-	2:59PM	??	R	3111:46.43	/opt/sdk/
120	0	1	0	139	0	652	select	2:59PM	??	Ss	154:48.02	syslogd -
142	0	1	0	8	0	1132	wait	2:59PM	??	S	122:57.26	/usr/sbin
20287	0	142	0	109	0	1100	-	10:24PM	??	R	0:00.08	/bin/ps -
0	0	0	0	-16	0	0	-	2:59PM	??	Wls	0:00.00	[swapper]
1	0	0	0	8	0	1152	wait	2:59PM	??	SLs	0:00.75	/sbin/ini
. . .												

**show extension-provider
system processes
extensive**

```

user@host> show extension-provider system processes extensive
Interface: ms-0/0/0
last pid: 20384; load averages: 11.51, 11.63, 11.69 up 0+07:27:42 22:26:32
91 processes: 45 running, 34 sleeping, 12 waiting
Mem: 8924K Active, 1784K Inact, 152M Wired, 156K Cache, 77M Buf, 200M Free
Swap:
  PID USERNAME  THR PRI NICE   SIZE    RES STATE  C  TIME  WCPU COMMAND
  259 root         8 139    0   643M    840K CPU13  d  52.1H 658.40% jnx-flow-data

    22 root         1 171   52    OK     8K CPU20  14   0:00 99.22% idle: cpu20
    21 root         1 171   52    OK     8K CPU21   0   0:00 99.22% idle: cpu21
    20 root         1 171   52    OK     8K CPU22   0   0:00 99.22% idle: cpu22
    15 root         1 171   52    OK     8K CPU27   0   0:00 98.49% idle: cpu27
    19 root         1 171   52    OK     8K CPU23   0   0:00 97.71% idle: cpu23
    28 root         1 171   52    OK     8K CPU14   0   0:00 97.71% idle: cpu14
    18 root         1 171   52    OK     8K CPU24  18   0:00 96.97% idle: cpu24
    26 root         1 171   52    OK     8K CPU16  10   0:00 96.97% idle: cpu16
    70 root         1 -16    0    OK     8K CPU8    0   0:00 96.97% poller: cpu8
    68 root         1 -16    0    OK     8K CPU1    0   0:00 96.97% pot: cpu1
    25 root         1 171   52    OK     8K CPU17   0   0:00 96.97% idle: cpu17
    17 root         1 171   52    OK     8K CPU25   0   0:00 96.97% idle: cpu25
    13 root         1 171   52    OK     8K CPU29   0   0:00 96.97% idle: cpu29
    12 root         1 171   52    OK     8K CPU30   0   0:00 96.97% idle: cpu30
    11 root         1 171   52    OK     8K CPU31   0   0:00 96.97% idle: cpu31
    16 root         1 171   52    OK     8K CPU26   0   0:00 96.97% idle: cpu26
    14 root         1 171   52    OK     8K CPU28  1c   0:00 96.24% idle: cpu28

Interface: ms-0/2/0
last pid: 13175; load averages: 4.00, 4.04, 4.00 up 0+04:46:34 22:26:32
88 processes: 37 running, 32 sleeping, 19 waiting
Mem: 6488K Active, 1228K Inact, 156M Wired, 24K Cache, 75M Buf, 460M Free
Swap:
  PID USERNAME  THR PRI NICE   SIZE    RES STATE  C  TIME  WCPU COMMAND
    12 root         1 171   52    OK     8K CPU30   0   0:00 98.49% idle: cpu30
    23 root         1 171   52    OK     8K CPU19   0   0:00 98.49% idle: cpu19
    20 root         1 171   52    OK     8K CPU22   0   0:00 98.49% idle: cpu22
    21 root         1 171   52    OK     8K CPU21   0   0:00 98.49% idle: cpu21
    75 root         1 -16    0    OK     8K CPU16   4   0:00 97.71% poller: cpu16
    76 root         1 -16    0    OK     8K CPU20   4   0:00 97.71% poller: cpu20
    13 root         1 171   52    OK     8K CPU29   0   0:00 97.71% idle: cpu29
    14 root         1 171   52    OK     8K CPU28  1c   0:00 96.24% idle: cpu28
    77 root         1 -16    0    OK     8K CPU24   4   0:00 96.24% poller: cpu24
    16 root         1 171   52    OK     8K CPU26   0   0:00 96.24% idle: cpu26
    11 root         1 171   52    OK     8K CPU31   0   0:00 96.24% idle: cpu31
    24 root         1 171   52    OK     8K CPU18   0   0:00 96.24% idle: cpu18
    25 root         1 171   52    OK     8K CPU17   0   0:00 96.24% idle: cpu17
    17 root         1 171   52    OK     8K CPU25   0   0:00 96.24% idle: cpu25
    19 root         1 171   52    OK     8K CPU23   0   0:00 96.24% idle: cpu23
    29 root         1 171   52    OK     8K CPU13  d 284:04 93.95% idle: cpu13
    30 root         1 171   52    OK     8K CPU12  c 284:05 92.48% idle: cpu12
    33 root         1 171   52    OK     8K CPU9    9 283:52 92.48% idle: cpu9

```

**show extension-provider
system processes
interface**

The output for the show extension-provider system processes interface command is identical to that for the show extension-provider system processes command except that the output for the former is for the specified interface only and the output for the latter is for all ms interfaces. For sample output, see show extension-provider system processes on page 73.

show extension-provider system processes wide

user@host> show extension-provider system processes wide

Interface: ms-1/0/0

PID	TT	STAT	TIME	PROVIDER	COMMAND
0	??	WLs	0:00.00	(null)	[swapper]
1	??	SLs	0:00.83		/sbin/init --
2	??	DL	0:24.86		[g_event]
3	??	DL	0:24.52		[g_up]
4	??	DL	0:24.38		[g_down]
5	??	DL	0:00.00		[thread taskq]
6	??	DL	0:00.00		[kqueue taskq]
9	??	DL	0:00.53		[pagedaemon]
10	??	DL	0:00.00		[ktrace]
11	??	RL	0:00.00		[idle: cpu31]
12	??	RL	0:00.00		[idle: cpu30]
13	??	RL	0:00.00		[idle: cpu29]
14	??	RL	0:00.17		[idle: cpu28]
15	??	RL	0:00.00		[idle: cpu27]
16	??	RL	0:00.00		[idle: cpu26]
17	??	RL	0:00.00		[idle: cpu25]
18	??	RL	0:00.17		[idle: cpu24]
19	??	RL	0:00.00		[idle: cpu23]
20	??	RL	0:00.00		[idle: cpu22]
21	??	RL	0:00.00		[idle: cpu21]
22	??	RL	0:00.17		[idle: cpu20]
23	??	RL	0:00.00		[idle: cpu19]
24	??	RL	0:00.00		[idle: cpu18]
25	??	RL	0:00.00		[idle: cpu17]
26	??	RL	0:00.17		[idle: cpu16]

...

show extension-provider system processes wide detail

user@host> show extension-provider system processes wide detail

Interface: ms-0/2/0

PID	UID	PPID	CPU	PRI	NI	RSS	WCHAN	STARTED	TT	STAT	TIME	COMMAND	PROVIDER
12	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu30]	
20	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu22]	
23	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu19]	
21	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu21]	
25	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu17]	
11	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu31]	
13	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu29]	
14	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.36	[idle: cpu28]	
16	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu26]	
17	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu25]	
19	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu23]	
24	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu18]	
75	0	0	0	-16	0	8	-	5:40PM	??	RL	0:00.00	[poller: cpu16]	
76	0	0	0	-16	0	8	-	5:40PM	??	RL	0:00.00	[poller: cpu20]	
77	0	0	0	-16	0	8	-	5:40PM	??	RL	0:00.00	[poller: cpu24]	
29	0	0	0	171	0	8	-	5:40PM	??	RL	288:22.84	[idle: cpu13]	
15	0	0	0	171	0	8	-	5:40PM	??	RL	0:00.00	[idle: cpu27]	
74	0	0	0	-16	0	8	-	5:40PM	??	RL	0:00.00	[pot: cpu1]	
33	0	0	0	171	0	8	-	5:40PM	??	RL	288:47.94	[idle: cpu9]	
30	0	0	0	171	0	8	-	5:40PM	??	RL	287:07.79	[idle: cpu12]	
34	0	0	0	171	0	8	-	5:40PM	??	RL	288:57.04	[idle: cpu8]	
37	0	0	0	171	0	8	-	5:40PM	??	RL	288:01.05	[idle: cpu5]	
38	0	0	0	171	0	8	-	5:40PM	??	RL	285:08.89	[idle: cpu4]	
42	0	0	0	171	0	8	-	5:40PM	??	RL	281:47.55	[idle: cpu0]	
0	0	0	0	-16	0	0	-	5:40PM	??	WLs	0:00.00	[swapper]	null)

```
1 0 0 0 8 0 1144 wait 5:40PM ?? SLs 0:00.58 /sbin/init -  
...
```

show extension-provider system uptime

Syntax	show extension-provider system uptime <interface>
Release Information	Command introduced in JUNOS Release 9.1.
Description	Show uptime on the extension provider PIC.
Options	interface—(Optional) Name of the extension provider interface.
Required Privilege Level	view
List of Sample Output	show extension-provider system uptime on page 78 show extension-provider system uptime interface on page 78
Output Fields	For a description of the output fields, see the output fields table for the show system uptime command in the <i>JUNOS System Basics and Services Command Reference</i> .
show extension-provider system uptime	<pre> user@host> show extension-provider system uptime Interface: ms-0/0/0 5:08PM up 26 mins, 0 users, load averages: 0.09, 0.06, 0.04 Interface: ms-0/2/0 5:08PM up 26 mins, 0 users, load averages: 0.15, 0.03, 0.01 </pre>
show extension-provider system uptime interface	<pre> user@host> show extension-provider system uptime interface ms-0/2/0 Interface: ms-0/2/0 5:08PM up 26 mins, 0 users, load averages: 0.15, 0.03, 0.01 </pre>

show extension-provider system virtual-memory

Syntax	show extension-provider system virtual-memory <interface>
Release Information	Command introduced in JUNOS Release 9.1.
Description	Show kernel dynamic memory usage on the extension provider PIC. Display the usage of JUNOS kernel memory listed first by size of allocation and then by type of usage.
Options	interface—(Optional) Name of the extension provider interface.
Required Privilege Level	view
List of Sample Output	show extension-provider system virtual-memory on page 79 show extension-provider system virtual-memory interface on page 80
Output Fields	For a description of the output fields, see Table 4 on page 79. Output fields are listed in the approximate order in which they appear.

Table 4: show extension-provider system virtual-memory Output Fields

Field	Field Description
Size (unlabeled)	Memory block size in bytes.
Type(s) (unlabeled)	Kernel modules that are using the memory blocks. For a definition of each type, see a FreeBSD book.
interrupt	Type of interrupt: <ul style="list-style-type: none"> ■ total—Total number of interrupts for each type. ■ rate—Interrupt rate.
Total	Total of all interrupts.

```

show extension-provider user@host> show extension-provider system virtual-memory
system virtual-memory Interface: ms-0/0/0
                        916976 cpu context switches
193097320 device interrupts
                        43468 software interrupts
                        0 traps
199882 system calls
                        78 kernel threads created
2416 fork() calls
                        0 vfork() calls
                        0 rfork() calls
                        0 swap pager pageins
                        0 swap pager pages paged in
                        0 swap pager pageouts
                        0 swap pager pages paged out
1307 vnode pager pageins

```

```

1307 vnode pager pages paged in
    0 vnode pager pageouts
    0 vnode pager pages paged out
    0 page daemon wakeups
    0 pages examined by the page daemon
454 pages reactivated
54109 copy-on-write faults
    11 copy-on-write optimized faults
65858 zero fill pages zeroed
65190 zero fill pages prezeroed
    10 intransit blocking page faults
206484 total VM faults taken
    0 pages affected by kernel thread creation
102942 pages affected by fork()
    0 pages affected by vfork()
    0 pages affected by rfork()
144325 pages freed
    0 pages freed by daemon
102373 pages freed by exiting processes
1749 pages active
    317 pages inactive
    28 pages in VM cache
38743 pages wired down
51895 pages free
    4096 bytes per page
    0 swap pages used
    0 peak swap pages used
109540 total name lookups
    cache hits (86% pos + 12% neg) system 0% per-directory
    deletions 0%, falsehits 0%, toolong 0%
interrupt          total      rate
clock              177515903    59689
Total              177515903    59689

```

**show extension-provider
system virtual-memory
interface**

```

user@host> show extension-provider system virtual-memory interface ms-0/2/0
Interface: ms-0/2/0
6971866 cpu context switches
757808764 device interrupts
101858 software interrupts
    0 traps
1129382 system calls
    80 kernel threads created
15764 fork() calls
    0 vfork() calls
    0 rfork() calls
    0 swap pager pageins
    0 swap pager pages paged in
    0 swap pager pageouts
    0 swap pager pages paged out
1212 vnode pager pageins
1212 vnode pager pages paged in
    0 vnode pager pageouts
    0 vnode pager pages paged out
    0 page daemon wakeups
    0 pages examined by the page daemon
420 pages reactivated
354621 copy-on-write faults
    0 copy-on-write optimized faults
430183 zero fill pages zeroed
424776 zero fill pages prezeroed
1434 intransit blocking page faults

```

```

1332189 total VM faults taken
    0 pages affected by kernel thread creation
648103 pages affected by fork()
    0 pages affected by vfork()
    0 pages affected by rfork()
892030 pages freed
    0 pages freed by daemon
673820 pages freed by exiting processes
1604 pages active
309 pages inactive
6 pages in VM cache
39994 pages wired down
117802 pages free
4096 bytes per page
0 swap pages used
0 peak swap pages used
702497 total name lookups
    cache hits (86% pos + 13% neg) system 0% per-directory
    deletions 0%, falsehits 0%, toolong 0%
interrupt          total      rate
clock              987886798    47811
Total              987886798    47811

```

show interfaces redundancy (SDK)

Syntax show interfaces redundancy

Release Information For routers running JUNOS SDK applications, support for rms interfaces added in JUNOS Release 9.5.

Description Display general information about the Multiservices interfaces redundancy (rms interface).

For this command's usage in other contexts, see the *JUNOS Interfaces Command Reference*.

Required Privilege Level view

Output Fields For a description of the output fields, see the output fields table for the show interfaces redundancy command in the *JUNOS Interfaces Command Reference*.

show interfaces redundancy both up

```
user@host> show interfaces redundancy
Interface State      Last change Primary   Secondary Current status
rms0      On primary    01:02:45  ms-0/0/0  ms-0/2/0  both up
```

show interfaces redundancy one or both down

```
user@host> show interfaces redundancy
Interface State      Last change Primary   Secondary Current status
rms0      Not present                ms-1/0/0  ms-0/2/0  both down
rms1      On secondary 1d 23:56  ms-1/2/0  ms-0/3/0  primary down
rms2      On primary   10:10:27  ms-1/3/0  ms-0/2/0  secondary down
```


show interfaces statistics (SDK)

Syntax	show interface statistics <i>interface-name</i> <brief detail terse>
Release Information	For routers running JUNOS SDK applications, support for rms interfaces added in JUNOS Release 9.5.
Description	Display interface statistics, such as packets received, packets transmitted, and errors. For this command's usage in other contexts, see the <i>JUNOS Interfaces Command Reference</i> .
Options	<i>interface-name</i> —Display statistics for the interface named. brief detail terse—(Optional) Display the specified detail level of output.
Required Privilege Level	view
Output Fields	For descriptions of the output fields see the output fields table for the show interfaces (Adaptive Services) command in the <i>JUNOS Interfaces Command Reference</i> .
show interfaces statistics	<pre> user@host> show interface statistics rms0 Physical interface: rms0, Enabled, Physical link is Up Interface index: 158, SNMP ifIndex: 149 Type: Adaptive-Services, Link-level type: Adaptive-Services, MTU: 1504, Speed: 1000mbps Device flags : Present Running Interface flags: Point-To-Point SNMP-Traps Internal: 0x4000 Link type : Full-Duplex Link flags : None Last flapped : 2009-01-27 18:17:21 PST (16:29:12 ago) Input rate : 0 bps (0 pps) Output rate : 0 bps (0 pps) Logical interface rms0.1 (Index 74) (SNMP ifIndex 150) Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: Adaptive-Services Bandwidth: 800mbps Input packets : 0 Output packets: 0 Protocol inet, MTU: 1504 Flags: None Addresses, Flags: Is-Primary Local: 1.1.1.1 </pre>
show interfaces statistics brief	<pre> user@host> show interfaces statistics rms0 brief Physical interface: rms0, Enabled, Physical link is Up Type: Adaptive-Services, Link-level type: Adaptive-Services, MTU: 1504, Clocking: Unspecified, Speed: 1000mbps Device flags : Present Running Interface flags: Point-To-Point SNMP-Traps Internal: 0x4000 Logical interface rms0.1 Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: Adaptive-Services </pre>

```
inet 1.1.1.1 --> 0/0
```

show interfaces user@host> **show interfaces statistics detail**

statistics detail

```
Physical interface: rms0, Enabled, Physical link is Up
Interface index: 158, SNMP ifIndex: 149, Generation: 168
Type: Adaptive-Services, Link-level type: Adaptive-Services, MTU: 1504,
Clocking: Unspecified, Speed: 1000mbps
Device flags : Present Running
Interface flags: Point-To-Point SNMP-Traps Internal: 0x4000
Link type : Full-Duplex
Link flags : None
Physical info : Unspecified
Hold-times : Up 0 ms, Down 0 ms
Current address: Unspecified, Hardware address: Unspecified
Alternate link address: Unspecified
Last flapped : 2009-01-27 18:17:21 PST (16:29:22 ago)
Statistics last cleared: 20-01-27 18:20:31 PST (16:26:12 ago)
Traffic statistics:
Input bytes : 84000 0 bps
Output bytes : 84000 0 bps
Input packets: 1000 0 pps
Output packets: 1000 0 pps
IPv6 transit statistics:
Input bytes : 0
Output bytes : 0
Input packets: 0
Output packets: 0
Input errors:
Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards:
0,
Resource errors: 0
Output errors:
Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors:
0

Logical interface rms0.1 (Index 74) (SNMP ifIndex 150) (Generation 143)
Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: Adaptive-Services
Bandwidth: 800mbps
Traffic statistics:
Input bytes : 84000
Output bytes : 84000
Input packets: 1000
Output packets: 1000
Local statistics:
Input bytes : 0
Output bytes : 0
Input packets: 0
Output packets: 0
Transit statistics:
Input bytes : 84000 0 bps
Output bytes : 84000 0 bps
Input packets: 1000 0 pps
Output packets: 1000 0 pps
Protocol inet, MTU: 1504, Generation: 154, Route table: 0
Flags: None
Addresses, Flags: Is-Primary
Destination: Unspecified, Local: 1.1.1.1, Broadcast: Unspecified,
Generation: 170
```

```
show interfaces user@host> show interfaces statistics terse rms0 terse  
statistics terse Interface      Admin Link Proto  Local      Remote  
rms0              up    up  
rms0.1            up    up   inet   1.1.1.1    --> 0/0
```

show services stateful-firewall flows (SDK)

Syntax	show services stateful-firewall flows <interface <i>interface-name</i> >
Release Information	For routers running JUNOS SDK applications, support for ms- interfaces added in JUNOS Release 9.5.
Description	Display stateful firewall flow table entries.
Options	none—Display standard information about all stateful firewall flows. interface <i>interface-name</i> —(Optional) Display information about the named interface.
Required Privilege Level	view
Output Fields	Table 5 on page 86 lists the output fields for the show services stateful-firewall flows command. Output fields are listed in the approximate order in which they appear.

Table 5: show services stateful-firewall flows Output Fields

Field	Field Description
Interface	Interface ID.
Service set	Name of service set.
Flow	Protocol used for the flow.
State	Status of the flow.
Dir	Direction of the flow: input (I) and output (O).
Frm count	Number of frames in the flow.

```

show services      user@host> show services stateful-firewall flows
stateful-firewall flows Interface: ms-2/2/0, Service ser: sset1
Flow Stats Dir Frm count
TCP 192.1.1.2.37822 -> 192.2.1.2.21 Watch I 16
TCP 192.2.1.2.21 -> 192.1.1.2.37822 Watch O 13
TCP 192.1.1.2.37822 -> 192.2.1.2.40598 Unknown I 2
TCP 192.2.1.2.40598 -> 192.1.1.2.37822 Unknown O 1

```

show services stateful-firewall statistics (SDK)

Syntax	show services stateful-firewall statistics <extensive>
Release Information	For routers running JUNOS SDK applications, support for ms- interfaces added in JUNOS Release 9.5.
Description	<p>Display only rule statistics or rule and application-level gateway (ALG) statistics.</p> <p>For this command's usage in other contexts, see the <i>JUNOS System Basics and Services Command Reference</i>.</p>
Options	<p>none—Display standard information about all stateful firewall statistics. For Multiservices interfaces, standard information is only rule statistics.</p> <p>extensive—(Optional) Display the extensive level of output. For Multiservices interfaces, the extensive level gives just rule and ALG statistics. Other statistics shown do not populate correctly and show all zeroes.</p>
Required Privilege Level	view
Output Fields	Table 6 on page 87 lists the output fields for the show services stateful-firewall statistics command. Output fields are listed in the approximate order in which they appear.

Table 6: show services stateful-firewall statistics Output Fields

Field	Field Description
Interface	Interface ID.
Service set	Name of service set.
Accept	Number of flows accepted.
Discard	Number of flows discarded.
Reject	Number of flows rejected.
Errors	Number of errors.

```

show services      user@host> show services stateful-firewall statistics
stateful-firewall Interface Service set Accept Discard Reject Errors
statistics       ms-5/1/0      sset1      1      1      0      0

```

show system processes (SDK)

Syntax	show system processes <wide>
Release Information	Support for JUNOS SDK applications added in JUNOS Release 9.0.
Description	Display information about software processes that are running on the router.
Options	wide—(Optional) Display process information that might be wider than 80 columns. This option shows the PROVIDER column.
Required Privilege Level	view
List of Sample Output	show system processes on page 88 show system processes wide on page 88
Output Fields	For a description of the output fields, see the output fields table for the show system processes command in the <i>JUNOS System Basics and Services Command Reference</i> .

show system processes

```

user@host> show system processes
      PID  TT  STAT      TIME COMMAND
...
      9   ??  DL      0:00.01 [pagedaemon]
...
    6463   ??  DL      0:00.18 [md9]
    6738   ??  S        0:00.44 /usr/sbin/mgd -N
    7001   ??  S        0:00.12 /opt/sbin/jnx-exampled -N
    7063   ??  Ss       0:00.03 mgd: (mgd) (regress)/dev/ttyp0 (mgd)

```

show system processes wide

```

user@host> show system processes wide
      PID  TT  STAT      TIME PROVIDER COMMAND
...
      9   ??  DL      0:00.01          [pagedaemon]
...
    6463   ??  DL      0:00.18          [md9]
    6738   ??  S        0:00.44          /usr/sbin/mgd -N
    7001   ??  S        0:00.12 jnx      /opt/sbin/jnx-exampled -N
    7063   ??  Ss       0:00.03 mgd: (mgd) (regress)/dev/ttyp0 (mgd)

```

show system processes health

Syntax	show system process health <process-name <i>name</i> pid <i>pid</i> >
Release Information	Command introduced in JUNOS Release 8.5.
Description	Display the resource utilization (health), of all the SDK applications currently running. You can display health information about one specific process by specifying either the process name or the process ID (PID).
Options	<p>process-name <i>name</i>—(Optional) Display health information about the process identified by the process name.</p> <p>pid <i>pid</i>—(Optional) Display health information about the process identified by the PID.</p>
Required Privilege Level	view
List of Sample Output	<p>show system processes health on page 90</p> <p>show system processes health pid on page 90</p> <p>show system processes health process-name on page 91</p>
Output Fields	For a description of the output fields, see Table 7 on page 89. Output fields are listed in the approximate order in which they appear.

Table 7: show system processes health Output Fields

Field Name	Field Description
PID	Process ID, a number that identifies the process.
Provider	Provider prefix. A string that identifies the provider of the SDK application.
Parent process	Process ID of the process that spawned the process in question.
Child processes	Process ID of the process that is launched from the process in question: <ul style="list-style-type: none"> ■ Process ID for child process—A number that identifies the child process. ■ Name of child process—A string that identifies the child process.
CPU accumulated	Maximum amount of CPU time that can be accumulated.
Heartbeat	A regular signal sent from a router to indicate that the router is up and running: <ul style="list-style-type: none"> ■ Interval—Number of seconds between heartbeats. ■ Allowed misses—Number of missed heartbeats allowed before the application restarts. ■ Last seen—Time in seconds when the last heartbeat occurred. ■ Total misses—Number of heartbeats missed so far.

Table 7: show system processes health Output Fields (continued)

Field Name	Field Description
Resource utilization	<p>How memory is divided:</p> <ul style="list-style-type: none"> ■ Area—Segment of memory. ■ Current—Current size of memory segment. ■ Max. allowed—Maximum size allowed for memory segment. ■ data size—Current and maximum sizes of data segment. ■ open files—Number of currently open files and the maximum allowed. ■ resident set size—Current and maximum size of resident set segment. ■ shared memory size—The amount of shared memory the process is using. ■ stack size—Current and maximum sizes of stack segment.

```

show system processes health
user@host> show system processes health
PID: 10075 (jnx-flow-mgmt)
Provider: jnx
Parent process: 1 (init)
CPU accumulated: 0 seconds
Resource utilization:
  Area          Current  Max. allowed
  data size      7KB      32MB
  open files     20       64
  resident set size 12KB     24MB
  shared memory size 4KB
  stack size     2KB      8MB
PID: 420 (jnx-exampled)
Provider: jnx
Parent process: 1 (init)
Child processes: 1
  PID  Process name
  421  jnx-exampled
CPU accumulated: 21 seconds
Heartbeat:
  Interval:      1s
  Allowed misses: 5
  Last seen:     1s ago (0 missed)
  Total misses:  2
Resource utilization:
  Area          Current  Max. allowed
  data size      24KB     16MB
  open files     23       128
  resident set size 1532KB   24MB
  shared memory size 43KB
  stack size     8KB      8MB

```

show system processes health pid The output for the `show system processes health pid pid` command is identical to that for the `show system processes health` command except that health information is displayed for only the process specified by the PID. For sample output, see `show system processes health` on page 90.

show system processes health *process-name* The output for the `show system processes health process-name` name command is identical to that for the `show system processes health` command except that health information is displayed for only the process named. For sample output, see `show system processes health` on page 90.

show system processes providers

Syntax	show system processes providers
Release Information	Command introduced in JUNOS Release 8.5.
Description	Display information about software processes that are running on the router. The output is similar to that of the show system processes extensive command, but this command displays only provider processes (that is, only external processes). Also, this command's output has an extra column labeled PROVIDER to display the provider prefix.
Required Privilege Level	view
List of Sample Output	show system processes providers on page 92
Output Fields	For a description of the output fields, see the output fields table for the show system processes command in the <i>JUNOS System Basics and Services Command Reference</i> . This table explains all the fields except for PROVIDER which is for the string that is the provider prefix for the SDK application running the process.
show system processes providers	<pre> user@host> show system processes providers last pid: 7014; load averages: 0.19, 0.09, 0.05 up 0+00:57:29 12:54:45 54 processes: 1 running, 53 sleeping Mem: 101M Active, 105M Inact, 31M Wired, 132M Cache, 69M Buf, 369M Free Swap: 1536M Total, 1536M Free PID USERNAME PROVIDER PRI NICE SIZE RES STATE TIME WCPU COMMAND 7001 root jnx 96 0 3240K 2700K select 0:00 0.00% jnx-exampled </pre>

show system processes resource-limits process-name

Syntax	show system processes resource-limits process-name <i>process-ui-name</i>
Release Information	Command introduced in JUNOS Release 9.6.
Description	Display the resource limits of a process in an SDK package.
Required Privilege Level	view
Output Fields	For a description of the output fields, see Table 8 on page 93. Output fields are listed in the approximate order in which they appear.

Table 8: show system processes resource-limits process-name Output Fields

Field Name	Field Description
Provider-ID	Process ID, a number that identifies the process.
Provider-prefix	Provider prefix. A string that identifies the provider of the SDK application.
Area	Segment of memory.
Max. allowed	Current value of the limits applied in the kernel (effective limit).
Max. configurable	Maximum value the administrator can set for the limit in the configuration.
cpu/priority	Highest priority number (nice level) process can run at.
cpu/time	Maximum amount of CPU time that can be accumulated.
file/core-size	Maximum size of a core file that can be created.
file/open	Maximum number of simultaneous open files.
file/size	Maximum size of a file that can be created.
memory/data size	Maximum size of data segment.
memory/locked-in	Maximum number of bytes that can be locked into memory.
memory/resident set size	Maximum size of resident set.
memory/stack size	Maximum size of stack segment.
memory/socket-buffers	Maximum amount of physical memory that may be dedicated to the socket buffers.

show system processes resource-limits process-name user@host> **show system processes resource-limits process-name jnx-example-service**

Resource Limits			
Area	Max. allowed	Max. configurable	
cpu/priority	10	10	
file/open	16	64	
memory/data-size	32MB	32MB	
memory/locked-in	16MB	16MB	
memory/resident-set-size	8MB	24MB	
memory/stack-size	4MB	8MB	

show system processes resource-limits process-name brief The output for the show system processes resource-limits process-name jnx-example-service brief command is the same as for the show system processes resource-limits process-name jnx-example-service command.

show system processes resource-limits process-name detail user@host> **show system processes resource-limits process-name jnx-example-service detail**

Provider-ID: 0x8000
Provider-prefix: systest

Resource Limits			
Area	Max. allowed	Max. configurable	
cpu/priority	10	10	
cpu/time	unlimited	unlimited	
file/size	unlimited	unlimited	
file/open	16	64	
file/core-size	unlimited	unlimited	
memory/data-size	32MB	32MB	
memory/locked-in	16MB	16MB	
memory/resident-set-size	8MB	24MB	
memory/stack-size	4MB	8MB	
memory/socket-buffers	unlimited	unlimited	

show system resource-cleanup processes

Syntax	show system resource-cleanup processes <detail>
Release Information	Command introduced in JUNOS Release 9.3.
Description	Display the list of processes that have been registered for resource cleanup services.
Options	detail—(Optional) Display the list of processes that have been registered for resource clean up services, along with the resources that have been requested for clean up.
Required Privilege Level	view
List of Sample Output	show system resource-cleanup processes on page 95 show system resource-cleanup processes detail on page 95
Output Fields	For a description of the output fields, see Table 9 on page 95. Output fields are listed in the approximate order in which they appear.

Table 9: show system resource-cleanup processes Output Fields

Field Name	Field Description
PID	Process ID, a number that identifies a process.
Process name	String that identifies the process.
Resources to clean	Resources that have been registered to be cleaned up.

```

show system      user@host> show system resource-cleanup processes
resource-cleanup PID      Process name      Resources to clean
processes       420      jnx-exampld      GENCFG, SYSV shared memory

```

```

show system      user@host> show system resource-cleanup processes detail
resource-cleanup PID      Process name      Resources to clean
processes detail 420      jnx-exampld      GENCFG blob major ID 0x8000, minor ID 0x0000
                                SYSV shared memory ID 65536, key 1108955839
                                SYSV shared memory ID 65537, key 1108955837

```

show version (SDK)

Syntax	show version
Release Information	Support for JUNOS SDK applications added in JUNOS Release 8.5.
Description	Display the hostname and version information of the software running on the router. For routers that have SDK application packages installed, the show version command lists those packages.
Required Privilege Level	view
List of Sample Output	show version on page 96
Output Fields	For a description of the output fields, see Table 10 on page 96. Output fields are listed in the approximate order in which they appear.

Table 10: show version Output Fields

Field Name	Field Description
Hostname	Router name.
Model	Router model number.
contents listing (unlabeled)	Package contents installed.

```

show version      user@host> show version
                    Hostname: router1
                    Model: m10i
                    JUNOS Base OS boot [I20070611_2103]
                    JUNOS Base OS Software Suite [8.5I20070611_2103]
                    JUNOS Kernel Software Suite [8.5I20070611_2103]
                    JUNOS Crypto Software Suite [8.5I20070611_2103]
                    JUNOS Packet Forwarding Engine Support (M/T Common) [8.5I20070611_2103]
                    JUNOS Packet Forwarding Engine Support (M7i/M10i) [8.5I20070611_2103]
                    JUNOS Online Documentation [8.5I20070611_2103]
                    JUNOS Routing Software Suite [8.5I20070611_2103]
                    JUNOS SDK Gateway Example Control Component [8.5I20070612_1932]
                    JUNOS SDK Gateway Example Dataplane Component [8.5I20070612_1932]
                    JUNOS SDK Gateway Example Management Component [8.5I20070612_1932]

```

Part 3

Index

- Index on page 99
- Index of Statements and Commands on page 103

Index

Symbols

#, comments in configuration statements.....	xx
(), in syntax descriptions.....	xx
< >, in syntax descriptions.....	xx
[], in configuration statements.....	xx
{ }, in configuration statements.....	xx
(pipe), in syntax descriptions.....	xx

B

braces, in configuration statements.....	xx
brackets	
angle, in syntax descriptions.....	xx
square, in configuration statements.....	xx

C

clear interfaces statistics command	
SDK applications.....	64
clear services stateful-firewall flows command	
SDK applications.....	65
comments, in configuration statements.....	xx
configuration mode commands.....	37, 38
control cores.....	5
conventions	
text and syntax.....	xix
curly braces, in configuration statements.....	xx
customer support.....	xxi
contacting JTAC.....	xxi

D

data cores.....	5
data flow affinity.....	5, 6
documentation	
comments on.....	xx

E

extension	
defined.....	3
extension package-name (show delete)	
command.....	44
usage guidelines.....	38

extension package-name delete command	
usage guidelines.....	40
extension package-name show command	
how it filters.....	38
usage guidelines.....	39
extension-provider statement.....	48
usage guidelines.....	5, 6, 8, 9, 10
extension-service statement.....	49
usage guidelines.....	19, 27, 28
extensions statement.....	50
usage guidelines.....	4

F

font conventions.....	xix
forwarding database.....	5, 6
setting for stateful firewall.....	24

H

high availability	
operational commands for.....	32
SDK, configuring.....	31

I

icons defined, notice.....	xix
----------------------------	-----

J

jservices-sfw package.....	23
----------------------------	----

L

Level I policy.....	13
Level II policy.....	13
Level III policy.....	13
Level IV policy.....	14

M

manifest file.....	13
manuals	
comments on.....	xx

ms- interface	
commands that support.....	26
forwarding-options sampling family	
statement.....	27
redundancy.....	32
Multiservices PIC	
configuring.....	5
configuring control and data cores.....	5
configuring data flow affinity.....	6
configuring object cache.....	6
configuring packages on the PIC.....	8
configuring Policy and Forwarding databases.....	6
configuring sampling service set.....	20
configuring syslog.....	9
configuring wired process memory.....	10

O

object cache.....	5, 6
setting for stateful firewall.....	24
Open IP Solution Development Program <i>See</i> OSDP	
OSDP.....	3

P

packages	
jservices-sfw.....	23
SDK application, installed on PIC.....	5, 8
parentheses, in syntax descriptions.....	xx
pmond <i>See</i> process health monitor	
policy database.....	5, 6
setting for stateful firewall.....	24
policy levels.....	13
and inheriting limits.....	16
and manifest file.....	13
and roles.....	13
Level I.....	13
Level II.....	13
Level III.....	13
Level IV.....	14
process health monitor.....	33
process-monitor statement.....	51
usage guidelines.....	33
provider	
defined.....	3
provider ID	
defined.....	4
enabling.....	4

R

redundancy-options statement	
usage guidelines.....	31
request interface (revert switchover) command	
SDK applications.....	66

resource limits	
displaying.....	15
example.....	16
resource-cleanup statement.....	52
usage guidelines.....	34
resource-limits statement.....	53
usage guidelines.....	14
resources	
cleaning up.....	34
monitoring.....	33
resources statement.....	55
rms interface	
creating.....	31
defined.....	31

S

sampling <i>See</i> traffic sampling	
sampling service set.....	20
SDK applications	
defined.....	3
deployment.....	4
SDK applications configuration	
overview.....	3
SDK service process <i>See</i> ssd	
service order	
SDK, configuring.....	20
service set	
sampling.....	20
service sets	
SDK, configuring.....	19
service-order statement.....	56
usage guidelines.....	19, 20
Services SDK	
defined.....	3
show chassis pic command	
SDK.....	67
show extension-provider system connections	
command.....	68
show extension-provider system packages	
command.....	71
show extension-provider system processes	
command.....	73
show extension-provider system uptime	
command.....	78
show extension-provider system virtual-memory	
command.....	79
show interfaces redundancy command	
SDK.....	82
show interfaces statistics command	
SDK applications.....	83
show services stateful-firewall flows command	
SDK.....	86
show services stateful-firewall statistics command	
SDK.....	87

show system processes command	
SDK.....	88
show system processes health command.....	89
show system processes providers command.....	92
show system processes resource-limits process-name	
command.....	93
example.....	16
usage guidelines.....	15
show system resource-cleanup processes	
command.....	95
show version command	
SDK applications.....	96
show display detail command	
SDK applications.....	45
usage guidelines.....	37
Software Development Kit <i>See</i> SDK	
ssd (SDK service process)	
enabling.....	4
stateful firewall	
restrictions.....	6
stateful firewall plug-in	
configuring memory for.....	24
stateful firewalls	
jservices-sfw package.....	23
operational commands for.....	26
SDK Kerberos-enabled, configuring.....	25
SDK plug-in for, loading.....	23
support, technical <i>See</i> technical support	
syntax conventions.....	xix
syslog statement	
SDK applications.....	57
usage guidelines.....	5, 9

T

technical support	
contacting JTAC.....	xxi
traceoptions statement.....	58
process monitor.....	59
resource cleanup.....	61
usage guidelines	
process montitor.....	33
resource cleanup.....	34
traffic sampling	
and forwarding database.....	5, 6
in SDK applications.....	27, 28

U

user cores.....	5
-----------------	---

W

wired process memory.....	10
---------------------------	----

Index of Statements and Commands

C

clear interfaces statistics command	
SDK applications.....	64
clear services stateful-firewall flows command	
SDK applications.....	65

E

extension-provider statement.....	48
extension-service statement.....	49
extensions statement.....	50

P

process-monitor statement.....	51
--------------------------------	----

R

request interface (revert switchover) command	
SDK applications.....	66
resource-cleanup statement.....	52
resource-limits statement.....	53
resources statement.....	55

S

service-order statement.....	56
show chassis pic command	
SDK.....	67
show extension-provider system connections	
command.....	68
show extension-provider system packages	
command.....	71
show extension-provider system processes	
command.....	73
show extension-provider system uptime	
command.....	78
show extension-provider system virtual-memory	
command.....	79
show interfaces redundancy command	
SDK.....	82
show interfaces statistics command	
SDK applications.....	83

show services stateful-firewall flows command	
SDK.....	86
show services stateful-firewall statistics command	
SDK.....	87
show system processes command	
SDK.....	88
show system processes health command.....	89
show system processes providers command.....	92
show system resource-cleanup processes	
command.....	95
show version command	
SDK applications.....	96
syslog statement	
SDK applications.....	57

T

traceoptions statement.....	58
process monitor.....	59
resource cleanup.....	61

