



JUNOS® Software

CLI User Guide

Release 9.6

Juniper Networks, Inc.

1194 North Mathilda Avenue
Sunnyvale, California 94089
USA

408-745-2000

www.juniper.net

Published: 2009-07-21

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 CLI User Guide

Release 9.6

Copyright © 2009, Juniper Networks, Inc.

All rights reserved. Printed in USA.

Writing: Abhilash Prabhakaran

Editing: Laura Singer

Illustration: Nathaniel Woodward

Cover Design: Edmonds Design

Revision History

July 2009— R1 JUNOS 9.6

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

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

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

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

- a. Customer shall use Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.
- b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius or IMS AAA software on multiple computers or virtual machines (e.g., Solaris zones) requires multiple licenses, regardless of whether such computers or virtualizations are physically contained on a single chassis.
- c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface, processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.
- d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.
- e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Abbreviated Table of Contents

	About This Guide	xxi
Part 1	Introduction	
Chapter 1	Introducing the CLI	3
Chapter 2	Getting Started: A Quick Tour of the CLI	9
Chapter 3	Getting Online Help	25
Part 2	Operational Mode and Configuration Mode	
Chapter 4	Using CLI Operational Commands to Monitor the Router	37
Chapter 5	Using Commands and Statements to Configure a Device Running JUNOS Software	61
Chapter 6	Managing Configurations	99
Chapter 7	Filtering Command Output	117
Chapter 8	Controlling the CLI Environment	127
Part 3	Advanced Features	
Chapter 9	Using Shortcuts, Wildcards, and Regular Expressions	135
Chapter 10	Configuration Groups	145
Chapter 11	Summary of Configuration Group Statements	187
Part 4	CLI Command Summaries	
Chapter 12	Summary of CLI Environment Commands	193
Chapter 13	Summary of CLI Configuration Mode Commands	209
Chapter 14	Summary of CLI Operational Mode Commands	243
Part 5	Index	
	Index	261
	Index of Statements and Commands	269

Table of Contents

	About This Guide	xxi
	JUNOS Documentation and Release Notes	xxi
	Objectives	xxii
	Audience	xxii
	Supported Platforms	xxii
	Using the Indexes	xxiii
	Using the Examples in This Manual	xxiii
	Merging a Full Example	xxiii
	Merging a Snippet	xxiv
	Documentation Conventions	xxiv
	Documentation Feedback	xxvi
	Requesting Technical Support	xxvii
Part 1	Introduction	
Chapter 1	Introducing the CLI	3
	Introducing the JUNOS Command-Line Interface	3
	Key Features of the CLI	4
	Understanding the JUNOS CLI Modes and Command and Statement Hierarchies	5
	JUNOS CLI Command Modes	5
	CLI Command Hierarchy	6
	Configuration Statement Hierarchy	6
	Moving Among Hierarchy Levels	7
	Other Tools to Configure and Monitor Devices Running JUNOS Software	8
	Commands and Configuration Statements for JUNOS FIPS	8
Chapter 2	Getting Started: A Quick Tour of the CLI	9
	Getting Started with the JUNOS Command-Line Interface	9
	Switching Between JUNOS CLI Operational and Configuration Modes	11

Configuring a User Account on a Device Running JUNOS Software	12
Checking the Status of a Device Running JUNOS Software	14
Configuring a Routing Protocol	16
Shortcut	17
Longer Configuration	17
Making Changes to a Routing Protocol Configuration	20
Rolling Back JUNOS Configuration Changes	22

Chapter 3

Getting Online Help 25

Getting Online Help from the JUNOS Command-Line Interface	25
Getting Help About Commands	26
Getting Help About a String in a Statement or Command	27
Getting Help About Configuration Statements	27
Getting Help About System Log Messages	28
JUNOS Command-Line Interface Online Help Features	28
Help for Omitted Statements	28
Using CLI Command Completion	29
Using Command Completion in Configuration Mode	29
Displaying Tips About CLI Commands	29
Examples: Using CLI Command Completion	30
Examples: Using Command Completion in Configuration Mode	31
Displaying the JUNOS CLI Command and Word History	32

Part 2

Operational Mode and Configuration Mode

Chapter 4

Using CLI Operational Commands to Monitor the Router 37

Overview of JUNOS CLI Operational Mode Commands	37
CLI Command Categories	37
Commonly Used Operational Mode Commands	39
JUNOS Operational Mode Commands That Combine Other Commands	40
Understanding brief, detail, extensive, and terse Options of JUNOS Operational Commands	41

Controlling the Scope of a Command	42
Operational Mode Commands on a TX Matrix Router	43
Examples of Routing Matrix Command Options	43
Monitoring Who Uses the CLI	45
Interface Naming Conventions Used in JUNOS Operational Commands	46
Physical Part of an Interface Name	46
Logical Part of an Interface Name	46
Channel Identifier Part of an Interface Name	47
Viewing Files and Directories on a Device Running JUNOS Software	47
Directories on the Router	48
Listing Files and Directories	48
Specifying Filenames and URLs	51
Displaying JUNOS Software Information	52
Managing Programs and Processes	54
Showing Software Processes	54
Restarting a JUNOS Software Process	56
Stopping the JUNOS Software	57
Rebooting the JUNOS Software	58
Using the JUNOS Comment Character #	59
Example: Using Comments in JUNOS Commands	59

Chapter 5

Using Commands and Statements to Configure a Device Running JUNOS Software 61

Understanding JUNOS CLI Configuration Mode	62
Configuration Mode Commands	62
Configuration Statements and Identifiers	64
Configuration Statement Hierarchy	66
Entering and Exiting Configuration Mode	68
Modifying the JUNOS Configuration	70
Displaying the Current Configuration	70
Example: Displaying the Current Configuration	71
Adding JUNOS Configuration Statements and Identifiers	72
Deleting a Statement from a JUNOS Configuration	73
Example: Deleting a Statement from the JUNOS Configuration	74
Copying a JUNOS Statement in the Configuration	75
Example: Copying a Statement in the JUNOS Configuration	75
Issuing Relative Configuration Commands	76
Renaming an Identifier	76
Example: Renaming an Identifier	76
Inserting a New Identifier	77
Example: Inserting a New Identifier	77
Deactivating and Reactivating Statements and Identifiers	79
Examples: Deactivating and Reactivating Statements and Identifiers	79
Adding Comments in a JUNOS Configuration	80
Example: Including Comments in a JUNOS Configurations	81
Verifying a JUNOS Configuration	82
Committing a JUNOS Configuration	83
Committing a JUNOS Configuration and Exiting Configuration Mode	84
Activating a JUNOS Configuration but Requiring Confirmation	84

Scheduling a JUNOS Commit Operation	85
Monitoring the JUNOS Commit Process	86
Adding a Comment to Describe the Committed Configuration	87
Updating the Alternate Boot Drive	88
When Multiple Users Configure the Software	88
Forms of the configure Command	89
Example: Using the configure Command	90
Displaying Users Currently Editing the Configuration	90
Using the configure exclusive Command	91
Updating the configure private Configuration	92
Displaying set Commands from the Configuration	93
Example: Displaying set Commands from the Configuration	93
Example: Displaying Required set Commands at the	
Current Hierarchy Level	94
Example: Displaying set Commands with the match Option	94
Displaying Additional Information About the Configuration	95

Chapter 6 Managing Configurations 99

Understanding How the JUNOS Configuration Is Stored	99
Returning to the Most Recently Committed Configuration	100
Returning to a Previously Committed JUNOS Configuration	100
Returning to a Configuration Prior to the One Most Recently	
Committed	101
Displaying Previous Configurations	101
Comparing Configuration Changes with a Prior Version	102
Creating and Returning to a Rescue Configuration	104
Saving a Configuration to a File	105
Loading a Configuration from a File	106
Examples: Loading a Configuration from a File	108
Additional Details About Specifying JUNOS Statements and Identifiers	110
Specifying Statements	110
Performing CLI Type-Checking	112
Synchronizing Routing Engines	113

Chapter 7 Filtering Command Output 117

Using the Pipe () Symbol to Filter JUNOS Command Output	117
Using Regular Expressions with the Pipe () Symbol to Filter JUNOS	
Command Output	118
Pipe () Filter Functions in the JUNOS Command-Line Interface	119
Comparing Configurations	119
Counting the Number of Lines of Output	121
Displaying Output in XML Tag Format	121
Ignoring Output That Does Not Match a Regular Expression	121
Displaying Output from the First Match of a Regular Expression	122
Retaining Output After the Last Screen	122
Displaying Output Beginning with the Last Entries	122
Displaying Output That Matches a Regular Expression	123
Preventing Output from Being Paginated	123

Sending Command Output to Other Users	124
Resolving IP Addresses	124
Saving Output to a File	124
Trimming Output by Specifying the Starting Column	125

Chapter 8 Controlling the CLI Environment 127

Controlling the JUNOS CLI Environment	127
Setting the Terminal Type	128
Setting the CLI Prompt	128
Setting the CLI Directory	128
Setting the CLI Timestamp	128
Setting the Idle Timeout	128
Setting the CLI to Prompt After a Software Upgrade	129
Setting Command Completion	129
Displaying CLI Settings	129
Example: Controlling the CLI Environment	129
Setting the JUNOS CLI Screen Length and Width	130
Setting the Screen Length	130
Setting the Screen Width	130
Understanding the Screen Length and Width Settings	130

Part 3 Advanced Features

Chapter 9 Using Shortcuts, Wildcards, and Regular Expressions 135

Using Keyboard Sequences to Move Around and Edit the JUNOS CLI	135
Using Wildcard Characters in Interface Names	137
Using Global Replace in a JUNOS Configuraton	137
Common Regular Expressions to Use with the replace Command	138
Using Global Replace in a JUNOS Configuration—Using the \n Back Reference	139
Using Global Replace in a JUNOS Configuration—Replacing an Interface Name	140
Using Global Replace in a JUNOS Configuration—Using the upto Option	141
Using Regular Expressions to Delete Related Items from a JUNOS Configuration	142

Chapter 10**Configuration Groups****145**

Understanding the JUNOS Configuration Groups	146
Configuration Groups Overview	146
Inheritance Model	146
Configuring Configuration Groups	147
Creating a JUNOS Configuration Group	147
Applying a JUNOS Configuration Group	148
Example: Configuring and Applying JUNOS Configuration Groups	149
Example: Creating and Applying Configuration Groups on a TX Matrix Router	150
Disabling Inheritance of a JUNOS Configuration Group	151
Using Wildcards with Configuration Groups	152
Configuring Sets of Statements with Configuration Groups	154
Configuring Interfaces Using JUNOS Configuration Groups	155
Configuring a Consistent Management IP Address	157
Configuring Peer Entities	158
Establishing Regional Configurations	160
Selecting Wildcard Names	161
Using JUNOS Defaults Groups	162
Example: Referencing the Preset Statement	163
Example: Viewing Default Statements That Have Been Applied to the Configuration	164
Inheritance Model	164
Creating and Applying Configuration Groups	165
Configuration Groups Configuration Guidelines	165
Creating a Configuration Group	165
Applying a Configuration Group	167
Example: Configuring and Applying Configuration Groups	168
Example: Creating and Applying Configuration Groups on a TX Matrix Router	168
Disabling Inheritance of a Configuration Group	169
Example: Disabling Inheritance on Interface so-1/1/0	170
Displaying Inherited Values	170
Using Wildcards with Configuration Groups	171
Example: Using Wildcards with Configuration Groups	173
Examples: Configuration Groups	174
Configuring Sets of Statements with Configuration Groups	175
Configuring Interfaces	176
Configuring a Consistent Management IP Address	178
Configuring Peer Entities	179
Establishing Regional Configurations	181
Selecting Wildcard Names	182
Using JUNOS Defaults Groups	183
Example: Referencing a Preset Statement	184
Example: Viewing Default Statements That Have Been Applied to the Configuration	185

Chapter 11	Summary of Configuration Group Statements	187
	apply-groups	187
	apply-groups-except	188
	groups	189
Part 4	CLI Command Summaries	
Chapter 12	Summary of CLI Environment Commands	193
	set cli complete-on-space	194
	set cli directory	195
	set cli idle-timeout	196
	set cli prompt	197
	set cli restart-on-upgrade	198
	set cli screen-length	199
	set cli screen-width	200
	set cli terminal	201
	set cli timestamp	202
	set date	203
	show cli	204
	show cli authorization	205
	show cli directory	206
	show cli history	207
Chapter 13	Summary of CLI Configuration Mode Commands	209
	activate	210
	annotate	211
	commit	212
	copy	215
	deactivate	216
	delete	217
	edit	218
	exit	219
	help	220
	insert	221
	load	222
	quit	224
	rename	225
	replace	226
	rollback	227
	run	228
	save	229
	set	231
	show	232

show display inheritance	233
show display omit	234
show display set	235
show display set relative	236
show groups junos-defaults	237
status	238
top	239
up	240
update	241
wildcard delete	242

Chapter 14**Summary of CLI Operational Mode Commands 243**

configure	244
file	245
help	246
(pipe)	247
request	249
restart	251
set	256
show	257

Part 5**Index**

Index	261
Index of Statements and Commands	269

List of Figures

Part 1

Introduction

Chapter 1	Introducing the CLI	3
	Figure 1: Monitoring and Configuring Routers	3
	Figure 2: Committing a Configuration	5
	Figure 3: CLI Command Hierarchy Example	6
	Figure 4: Configuration Statement Hierarchy Example	7

Part 2

Operational Mode and Configuration Mode

Chapter 4	Using CLI Operational Commands to Monitor the Router	37
	Figure 5: Commands That Combine Other Commands	40
	Figure 6: Command Output Options	42
	Figure 7: Restarting a Process	57
Chapter 5	Using Commands and Statements to Configure a Device Running JUNOS Software	61
	Figure 8: Configuration Mode Hierarchy of Statements	66
	Figure 9: Confirm a Configuration	85
Chapter 6	Managing Configurations	99
	Figure 10: Commands for Storing and Modifying the Router Configuration	100
	Figure 11: Overriding the Current Configuration	108
	Figure 12: Using the replace Option	108
	Figure 13: Using the merge Option	109
	Figure 14: Using a Patch File	109
	Figure 15: Using the set Option	110

Part 3

Advanced Features

Chapter 9	Using Shortcuts, Wildcards, and Regular Expressions	135
	Figure 16: Replacement by Object	141

List of Tables

	About This Guide	xxi
	Table 1: Notice Icons	xxv
	Table 2: Text and Syntax Conventions	xxv
Part 1	Introduction	
Chapter 1	Introducing the CLI	3
	Table 3: CLI Configuration Mode Navigation Commands	7
Part 2	Operational Mode and Configuration Mode	
Chapter 4	Using CLI Operational Commands to Monitor the Router	37
	Table 4: Commonly Used Operational Mode Commands	39
	Table 5: Directories on the Router	48
	Table 6: show system process extensive Command Output Fields	56
Chapter 5	Using Commands and Statements to Configure a Device Running JUNOS Software	61
	Table 7: Summary of Configuration Mode Commands	62
	Table 8: Configuration Mode Top-Level Statements	64
	Table 9: Forms of the configure Command	89
Chapter 6	Managing Configurations	99
	Table 10: CLI Configuration Input Types	112
Chapter 7	Filtering Command Output	117
	Table 11: Common Regular Expression Operators in Operational Mode Commands	118
Part 3	Advanced Features	
Chapter 9	Using Shortcuts, Wildcards, and Regular Expressions	135
	Table 12: CLI Keyboard Sequences	136
	Table 13: Wildcard Characters for Specifying Interface Names	137
	Table 14: Common Regular Expressions to Use with the replace Command	138
	Table 15: Replacement Examples	139

About This Guide

This preface provides the following guidelines for using the *JUNOS® Software CLI User Guide*:

- JUNOS Documentation and Release Notes on page xxi
- Objectives on page xxii
- Audience on page xxii
- Supported Platforms on page xxii
- Using the Indexes on page xxiii
- Using the Examples in This Manual on page xxiii
- Documentation Conventions on page xxiv
- Documentation Feedback on page xxvi
- Requesting Technical Support on page xxvii

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 describes how to use the JUNOS command-line interface (CLI) to configure, monitor, and manage Juniper Networks routers.



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 Platforms

For the features described in this manual, JUNOS Software currently supports the following platforms:

- J Series
- M Series

- MX Series
- T Series
- EX Series

Using the Indexes

This reference contains two indexes: a complete index that includes topic entries, and an index of statements and commands only.

In the index of statements and commands, an entry refers to a statement summary section only. In the complete index, the entry for a configuration statement or command contains at least two parts:

- The primary entry refers to the statement summary section.
- The secondary entry, *usage guidelines*, refers to the section in a configuration guidelines chapter that describes how to use the statement or command.

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.xml; }

```

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 xxv 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 xxv 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>
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> >;

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

Convention	Description	Examples
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	broadcast multicast (string1 string2 string3)
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	rsvp { # Required for dynamic MPLS only
[] (square brackets)	Enclose a variable for which you can substitute one or more values.	community name members [community-ids]
Indentation and braces ({ })	Identify a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop address; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
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 name
- Document part number
- Page number
- Software release version (not required for Network Operations Guides [NOGs])

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 postsales 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 located at <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, visit us at <http://www.juniper.net/support/requesting-support.html>.

Part 1

Introduction

- Introducing the CLI on page 3
- Getting Started: A Quick Tour of the CLI on page 9
- Getting Online Help on page 25

Chapter 1

Introducing the CLI

This chapter contains the following topics:

- Introducing the JUNOS Command-Line Interface on page 3
- Understanding the JUNOS CLI Modes and Command and Statement Hierarchies on page 5
- Other Tools to Configure and Monitor Devices Running JUNOS Software on page 8
- Commands and Configuration Statements for JUNOS FIPS on page 8

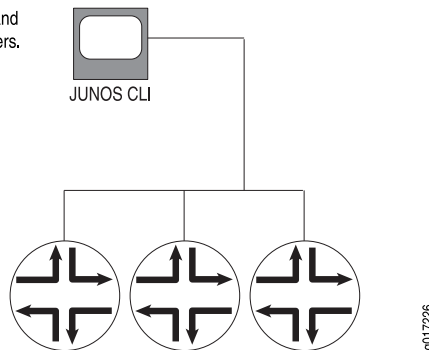
Introducing the JUNOS Command-Line Interface

The JUNOS command-line interface (CLI) is the software interface you use to access a device running JUNOS Software—whether from the console or through a network connection.

The JUNOS CLI is a Juniper Networks-specific command shell that runs on top of a FreeBSD UNIX-based operating system kernel. By leveraging industry-standard tools and utilities, the CLI provides a powerful set of commands that you can use to monitor and configure devices running JUNOS Software. (See Figure 1 on page 3.) The CLI is a straightforward command interface. You type commands on a single line, and the commands are executed when you press the Enter key.

Figure 1: Monitoring and Configuring Routers

Use the JUNOS CLI to monitor and configure Juniper Networks routers.



Key Features of the CLI

The JUNOS CLI commands and statements follow a hierarchical organization and have a regular syntax. The JUNOS CLI provides the following features to simplify CLI use:

- Consistent command names—Commands that provide the same type of function have the same name, regardless of the portion of the software on which they are operating. For example, all **show** commands display software information and statistics, and all **clear** commands erase various types of system information.
- Lists and short descriptions of available commands—Information about available commands is provided at each level of the CLI command hierarchy. If you type a question mark (?) at any level, you see a list of the available commands along with a short description of each command. This means that if you already are familiar with the JUNOS Software or with other routing software, you can use many of the CLI commands without referring to the documentation.
- Command completion—Command completion for command names (keywords) and for command options is available at each level of the hierarchy. To complete a command or option that you have partially typed, press the Tab key or the Spacebar. If the partially typed letters begin a string that uniquely identifies a command, the complete command name appears. Otherwise, a beep indicates that you have entered an ambiguous command, and the possible completions are displayed. Completion also applies to other strings, such as filenames, interface names, usernames, and configuration statements.
- Industry-standard technology—With FreeBSD UNIX as the kernel, a variety of UNIX utilities are available on the JUNOS CLI. For example, you can:
 - Use regular expression matching to locate and replace values and identifiers in a configuration, filter command output, or examine log file entries.
 - Use Emacs-based key sequences to move around on a command line and scroll through the recently executed commands and command output.
 - Store and archive JUNOS device files on a UNIX-based file system.
 - Use standard UNIX conventions to specify filenames and paths.
 - Exit from the CLI environment and create a UNIX C shell or Bourne shell to navigate the file system, manage router processes, and so on.

- Related Topics**
- Understanding the JUNOS CLI Modes and Command and Statement Hierarchies on page 5
 - Getting Started with the JUNOS Command-Line Interface on page 9
 - Other Tools to Configure and Monitor Devices Running JUNOS Software on page 8
 - Commands and Configuration Statements for JUNOS-FIPS on page 8

Understanding the JUNOS CLI Modes and Command and Statement Hierarchies

The JUNOS command-line interface (CLI) commands and statements are organized under two command modes and various hierarchies. The following sections provide you an overview of the JUNOS CLI command modes and commands and statements hierarchies:

- JUNOS CLI Command Modes on page 5
- CLI Command Hierarchy on page 6
- Configuration Statement Hierarchy on page 6
- Moving Among Hierarchy Levels on page 7

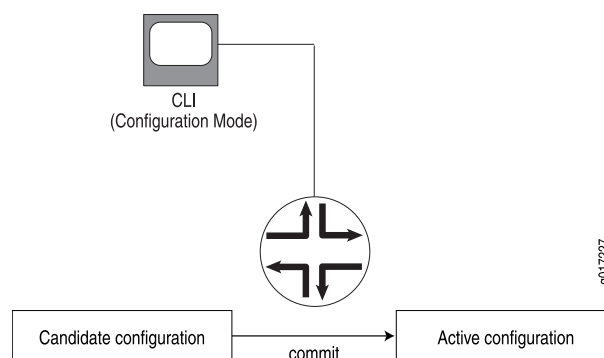
JUNOS CLI Command Modes

The JUNOS CLI has two modes:

- Operational mode—This mode displays the current status of the device. In operational mode, you enter commands to monitor and troubleshoot JUNOS Software and devices and network connectivity.
- Configuration mode—A JUNOS device configuration is stored as a hierarchy of statements. In configuration mode, you enter these statements to define all properties of JUNOS Software, including interfaces, general routing information, routing protocols, user access, and several system and hardware properties.

When you enter configuration mode, you are actually viewing and changing a file called the *candidate configuration*. The candidate configuration allows you to make configuration changes without causing operational changes to the current operating configuration, called the *active configuration*. The router does not implement the changes you added to the candidate configuration until you commit them, which activates the configuration on the router. (See Figure 2 on page 5.) Candidate configurations enable you to alter your configuration without causing potential damage to your current network operations.

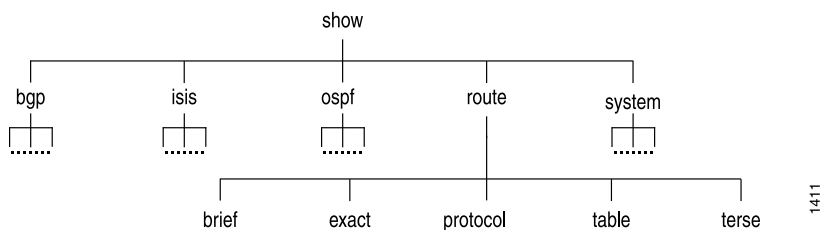
Figure 2: Committing a Configuration



CLI Command Hierarchy

CLI commands are organized in a hierarchy. Commands that perform a similar function are grouped together under the same level of the hierarchy. For example, all commands that display information about the system and the system software are grouped under the **show system** command, and all commands that display information about the routing table are grouped under the **show route** command. Figure 3 on page 6 illustrates a portion of the **show** command hierarchy.

Figure 3: CLI Command Hierarchy Example

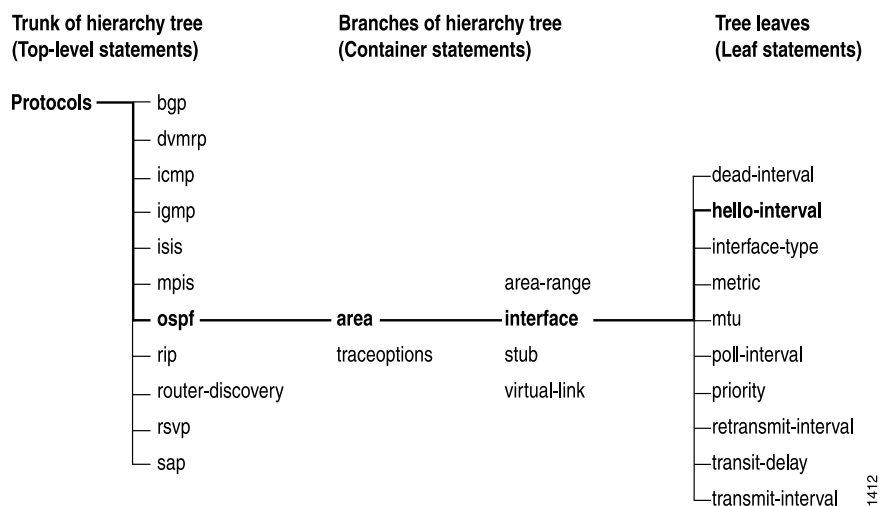


To execute a command, you enter the full command name, starting at the top level of the hierarchy. For example, to display a brief view of the routes in the router table, use the command **show route brief**.

Configuration Statement Hierarchy

The configuration statement hierarchy has two types of statements: *container statements*, which are statements that contain other statements, and *leaf statements*, which do not contain other statements. All of the container and leaf statements together form the *configuration hierarchy*.

Figure 4 on page 7 illustrates a part of the hierarchy tree. The **protocols** statement is a top-level statement at the trunk of the configuration tree. The **ospf**, **area**, and **interface** statements are all subordinate container statements of a higher statement (they are branches of the hierarchy tree), and the **hello-interval** statement is a leaf on the tree.

Figure 4: Configuration Statement Hierarchy Example

Moving Among Hierarchy Levels

You can use the CLI commands in Table 3 on page 7 to navigate the levels of the configuration statement hierarchy.

Table 3: CLI Configuration Mode Navigation Commands

Command	Description
<code>edit hierarchy-level</code>	Moves to an existing configuration statement hierarchy or creates a hierarchy and moves to that level.
<code>exit</code>	Moves up the hierarchy to the previous level where you were working. This command is, in effect, the opposite of the <code>edit</code> command. Alternatively, you can use the <code>quit</code> command. <code>exit</code> and <code>quit</code> are interchangeable.
<code>up</code>	Moves up the hierarchy one level at a time.
<code>top</code>	Moves directly to the top level of the hierarchy.

- Related Topics**
- Introducing the JUNOS Command-Line Interface on page 3
 - Getting Started with the JUNOS Command-Line Interface on page 9

Other Tools to Configure and Monitor Devices Running JUNOS Software

Apart from the command-line interface, JUNOS Software also supports the following applications, scripts, and utilities that enable you to configure and monitor devices running JUNOS Software:

- J-Web graphical user interface (GUI)—Allows you to monitor, configure, troubleshoot, and manage the router on a client by means of a Web browser with Hypertext Transfer Protocol (HTTP) or HTTP over Secure Sockets Layer (HTTPS) enabled. For more information, see the *J-Web Interface User Guide*.
- JUNOScript Application Programming Interface (API)—Application programmers can use the JUNOScript API to monitor and configure Juniper Networks routers. Juniper Networks provides a Perl module with the API to help you more quickly and easily develop custom Perl scripts for configuring and monitoring routers. For more information, see the *JUNOScript API Guide*.
- NETCONF Application Programming Interface (API)—Application programmers can also use the NETCONF API to monitor and configure Juniper Networks routers. For more information, see the *NETCONF API Guide*.
- JUNOS commit scripts and self-diagnosis features—You can define scripts to enforce custom configuration rules, use commit script macros to provide simplified aliases for frequently used configuration statements, and configure diagnostic event policies and actions associated with each policy. For more information, see the *JUNOS Configuration and Diagnostic Automation Guide*.
- Management Information Bases (MIBs)—You can use enterprise-specific and standard MIBs to retrieve information about the hardware and software components on a Juniper Networks router. For more information about MIBs, see the *JUNOS Network Management Configuration Guide*.

Related Topics

- Introducing the JUNOS Command-Line Interface on page 3
- Getting Started with the JUNOS Command-Line Interface on page 9
- Commands and Configuration Statements for JUNOS-FIPS on page 8

Commands and Configuration Statements for JUNOS FIPS

JUNOS FIPS enables you to configure a network of Juniper Networks routers in a Federal Information Processing Standards (FIPS) 140-2 environment.

The JUNOS FIPS software environment requires the installation of FIPS software by a crypto officer. In JUNOS FIPS, some JUNOS commands and statements have restrictions and some additional configuration statements are available. For more information, see the *Secure Configuration Guide for Common Criteria and JUNOS FIPS*.

Chapter 2

Getting Started: A Quick Tour of the CLI

This chapter contains the following topics:

- Getting Started with the JUNOS Command-Line Interface on page 9
- Switching Between JUNOS CLI Operational and Configuration Modes on page 11
- Configuring a User Account on a Device Running JUNOS Software on page 12
- Checking the Status of a Device Running JUNOS Software on page 14
- Configuring a Routing Protocol on page 16
- Rolling Back JUNOS Configuration Changes on page 22

Getting Started with the JUNOS Command-Line Interface

As an introduction to the JUNOS command-line interface (CLI), this topic provides instructions for simple steps you take after installing JUNOS Software on the device. It shows you how to start the CLI, view the command hierarchy, and make small configuration changes. The related topics listed at the end of this topic provide you more detailed information about using the CLI.



NOTE: The instructions and examples in this topic are based on sample M Series and T Series routers. You can use them as a guideline for entering commands on your devices running JUNOS Software.

Before You Begin

Make sure your device hardware is set up and JUNOS Software is installed. You must have a direct console connection to the device or network access using SSH or Telnet. If your device is not set up, follow the installation instructions provided with the device before proceeding.

Logging In to a Device Running JUNOS Software

Log in to the router and start the CLI:

1. Log in as **root**.

The root login account has superuser privileges, with access to all commands and statements.

2. Start the CLI:

```
root# cli
root@>
```

The `>` command prompt shows you are in operational mode. Later, when you enter configuration mode, the prompt will change to `#`.



NOTE: If you are using the root account for the first time on the device, remember that the device ships with no password required for root, but the first time you commit a configuration with JUNOS Release 7.6 software (or a later release), you must set a root password. Root access is not allowed over a telnet session. To enable root access over an SSH connection, you must configure the `system services ssh root-login allow` statement.

Displaying JUNOS Commands

The CLI includes several ways to get help about commands. This section shows some examples of how to get help.

1. Type `?` to show the top-level commands available in operational mode.

```
root@> ?
```

Possible completions:

clear	Clear information in the system
configure	Manipulate software configuration information
diagnose	Invoke diagnose script
file	Perform file operations
help	Provide help information
monitor	Show real-time debugging information
mtrace	Trace multicast path from source to receiver
ping	Ping remote target
quit	Exit the management session
request	Make system-level requests
restart	Restart software process
set	Set CLI properties, date/time, craft interface message
show	Show system information
ssh	Start secure shell on another host
start	Start shell
telnet	Telnet to another host
test	Perform diagnostic debugging
traceroute	Trace route to remote host

2. Type `file ?` to show all possible completions for the `file` command.

```
root@> file ?
```

Possible completions:

<[Enter]>	Execute this command
archive	Archives files from the system
checksum	Calculate file checksum
compare	Compare files
copy	Copy files (local or remote)
delete	Delete files from the system
list	List file information
rename	Rename files
show	Show file contents
source-address	Local address to use in originating the connection
	Pipe through a command

3. Type `file archive ?` to show all possible completions for the `file archive` command.

```
root@> file archive ?
```

Possible completions:

compress	Compresses the archived file using GNU gzip (.tgz)
destination	Name of created archive (URL, local, remote, or floppy)
source	Path of directory to archive

For more information about getting help about commands and statements, see “Getting Online Help” on page 25.

Switching Between JUNOS CLI Operational and Configuration Modes

When you monitor and configure a device running JUNOS Software, you may need to switch between operational mode and configuration mode. When you change to configuration mode, the command prompt also changes. The operational mode prompt is a right angle bracket (>) and the configuration mode prompt is a pound sign (#).

To switch between operational mode and configuration mode:

1. When you log in to the router and type the `cli` command, you are automatically in operational mode:

```
--- JUNOS 9.2B1.8 built 2008-05-09 23:41:29 UTC
% cli
user@host>
```

2. To enter configuration mode, type the `configure` command or the `edit` command from the CLI operation mode. For example:

```
user@host> configure
Entering configuration mode

[edit]
user@host#
```

The CLI prompt changes from `user@host>` to `user@host#` and a banner appears to indicate the hierarchy level.

3. You can return to operational mode in one of the following ways:
 - To commit the configuration and exit:

```
[edit]
user@host# commit and-quit
commit complete
Exiting configuration mode
user@host>
```

- To exit without committing:

```
[edit]
user@host# exit
Exiting configuration mode

user@host>
```

When you exit configuration mode, the CLI prompt changes from **user@host#** to **user@host>** and the banner no longer appears. You can enter or exit configuration mode as many times as you wish without committing your changes.

4. To display the output of an operational mode command, such as **show**, while in configuration mode, issue the **run** configuration mode command and then specify the operational mode command:

```
[edit]
user@host# run operational-mode-command
```

For example, to display the currently set priority value of the Virtual Router Redundancy Protocol (VRRP) primary router while you are modifying the VRRP configuration for a backup router:

```
[edit interfaces xe-4/2/0 unit 0 family inet vrrp-group 27]
user@host# show
virtual-address [ 192.168.1.15 ];
[edit interfaces xe-4/2/0 unit 0 family inet vrrp-group 27]
user@host# run show vrrp detail
Physical interface: xe-5/2/0, Unit: 0, Address: 192.168.29.10/24
Interface state: up, Group: 10, State: backup
Priority: 190, Advertisement interval: 3, Authentication type: simple
Preempt: yes, VIP count: 1, VIP: 192.168.29.55
Dead timer: 8.326, Master priority: 201, Master router: 192.168.29.254
[edit interfaces xe-4/2/0 unit 0 family inet vrrp-group 27]
user@host# set priority ...
```

Related Topics ■ Understanding the JUNOS CLI Modes and Command and Statement Hierarchies on page 5

Configuring a User Account on a Device Running JUNOS Software

This topic describes how to log on to a device running JUNOS Software using a root account and configure a new user account. You can configure an account for your own use or create a test account.

To configure a user account on the device:

1. Log in as root and enter configuration mode:

```
root@host> configure
[edit]
root@host#
```


The prompt in brackets ([edit]), also known as a *banner*, shows that you are in configuration edit mode, at the top of the hierarchy.

2. Change to the [edit system login] section of the configuration:

```
[edit]
root@host# edit system login
[edit system login]
root@host#
```

The prompt in brackets changes to [edit system login] to show you are at a new level in the hierarchy.

3. Now add a new user account:

```
[edit system login]
root@host# edit user nchen
```

This example adds an account `nchen` (for Nathan Chen), but you can use any account name.

4. Configure a full name for the account. If the name includes spaces, enclose the entire name in quotation marks (" "):

```
[edit system login user nchen]
root@host# set full-name "Nathan Chen"
```

5. Configure an account class. The account class sets the user access privileges for the account.

```
[edit system login user nchen]
root@host# set class super-user
```

6. Configure an authentication method and password for the account:

```
[edit system login user nchen]
root@host# set authentication plain-text-password
New password:
Retype new password:
```

When the new password prompt appears, enter a clear-text password that the system will encrypt, and then confirm the new password.

7. Commit the configuration:

```
[edit system login user nchen]
root@host# commit
commit complete
```

Configuration changes are not activated until you commit the configuration. If the commit is successful, a `commit complete` message appears.

8. Return to the top level of the configuration, and then exit:

```
[edit system login user nchen]
root@host# top
[edit]
root@host# exit
```

Exiting configuration mode

9. Log out of the device:

```
root@host> exit
% logout Connection closed.
```

10. To test your changes, log back in with the user account and password you just configured:

```
login: nchen
Password: password
— JUNOS 8.3-R1.1 built 2005-12-15 22:42:19 UTC
nchen@host>
```

When you log in, you should see the new username at the command prompt.

Congratulations! You have successfully used the CLI to view the device status and perform a simple configuration change. Now you are ready to learn more about the CLI. See the related topics listed in this section for more information about the JUNOS CLI features.

Alternatively, you can follow the instructions in “Configuring a Routing Protocol” on page 16 to create a more extensive configuration.



NOTE: For complete information about the commands to issue to configure your device, including examples, see JUNOS Software configuration guides.

- Related Topics**
- Getting Started with the JUNOS Command-Line Interface on page 9
 - Getting Online Help from the JUNOS Command-Line Interface on page 25

Checking the Status of a Device Running JUNOS Software

You can use **show** commands to check the status of the device and monitor the activities on the device.

To help you become familiar with **show** commands:

- Type **show ?** to display the list of **show** commands you can use to monitor the router:

```
root@> show ?
```

Possible completions:

accounting	Show accounting profiles and records
aps	Show Automatic Protection Switching information
arp	Show system Address Resolution Protocol table
entries	
as-path	Show table of known autonomous system paths
bfd	Show Bidirectional Forwarding Detection information
bgp	Show Border Gateway Protocol information

chassis	Show chassis information
class-of-service	Show class-of-service (CoS) information
cli	Show command-line interface settings
configuration	Show current configuration
connections	Show circuit cross-connect connections
dvmrp	Show Distance Vector Multicast Routing Protocol
info	
dynamic-tunnels	Show dynamic tunnel information information
esis	Show end system-to-intermediate system information
firewall	Show firewall information
helper	Show port-forwarding helper information
host	Show hostname information from domain name server
igmp	Show Internet Group Management Protocol information
ike	Show Internet Key Exchange information
ilmi	Show interim local management interface information
interfaces	Show interface information
ipsec	Show IP Security information
ipv6	Show IP version 6 information
isis	Show Intermediate System-to-Intermediate System
info	
l2circuit	Show Layer 2 circuit information
l2vpn	Show Layer 2 VPN information
lacp	Show Link Aggregation Control Protocol information
ldp	Show Label Distribution Protocol information
link-management	Show link management information
llc2	Show LLC2 protocol related information
log	Show contents of log file
mld	Show multicast listener discovery information
mpls	Show Multiprotocol Label Switching information
msdp	Show Multicast Source Discovery Protocol information
multicast	Show multicast information
ntp	Show Network Time Protocol information
ospf	Show Open Shortest Path First information
ospf3	Show Open Shortest Path First version 3 information
passive-monitoring	Show information about passive monitoring
pfe	Show Packet Forwarding Engine information
pgm	Show Pragmatic Generalized Multicast information
pim	Show Protocol Independent Multicast information
policer	Show interface policer counters and information
policy	Show policy information
ppp	Show PPP process information
rip	Show Routing Information Protocol information
ripng	Show Routing Information Protocol for IPv6 info
route	Show routing table information
rsvp	Show Resource Reservation Protocol information
sap	Show Session Announcement Protocol information
security	Show security information
services	Show services information
snmp	Show Simple Network Management Protocol information
system	Show system information
task	Show routing protocol per-task information
ted	Show Traffic Engineering Database information
version	Show software process revision levels
vp1s	Show VPLS information
vrrp	Show Virtual Router Redundancy Protocol information

- Use the `show chassis routing-engine` command to view the Routing Engine status:

```
root@> show chassis routing-engine
```

```

Routing Engine status:
Slot 0:
  Current state           Master
  Election priority       Master (default)
  Temperature             31 degrees C / 87 degrees F
  CPU temperature         32 degrees C / 89 degrees F
  DRAM                    768 MB
  Memory utilization      84 percent
  CPU utilization:
    User                  0 percent
    Background            0 percent
    Kernel                1 percent
    Interrupt             0 percent
    Idle                  99 percent
  Model                   RE-2.0
  Serial ID               b10000078c10d701
  Start time              2005-12-28 13:52:00 PST
  Uptime                  12 days, 3 hours, 44 minutes, 19 seconds
  Load averages:         1 minute   5 minute   15 minute
                        0.02        0.01        0.00

```

- Use the `show system storage` command to view available storage on the device:

```
root@> show system storage
```

Filesystem	Size	Used	Avail	Capacity	Mounted on
/dev/ad0s1a	865M	127M	669M	16%	/
devfs	1.0K	1.0K	0B	100%	/dev
devfs	1.0K	1.0K	0B	100%	/dev/
/dev/md0	30M	30M	0B	100%	/packages/mnt/jbase
/dev/md1	158M	158M	0B	100%	
/packages/mnt/jkernel-9.3B1.5					
/dev/md2	16M	16M	0B	100%	
/packages/mnt/jpfe-M7i-9.3B1.5					
/dev/md3	3.8M	3.8M	0B	100%	
/packages/mnt/jdocs-9.3B1.5					
/dev/md4	44M	44M	0B	100%	
/packages/mnt/jroute-9.3B1.5					
/dev/md5	12M	12M	0B	100%	
/packages/mnt/jcrypto-9.3B1.5					
/dev/md6	25M	25M	0B	100%	
/packages/mnt/jpfe-common-9.3B1.5					
/dev/md7	1.5G	196K	1.4G	0%	/tmp
/dev/md8	1.5G	910K	1.4G	0%	/mfs
/dev/ad0s1e	96M	38K	88M	0%	/config
procfs	4.0K	4.0K	0B	100%	/proc
/dev/ad1s1f	17G	2.6G	13G	17%	/var

Configuring a Routing Protocol

This section describes how to configure an OSPF backbone area that has two SONET interfaces. You can use it as a starting point for configuring additional protocols at a later time.

The final configuration looks like this:

```
[edit]
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
        dead-interval 20;
      }
      interface so-0/0/1 {
        hello-interval 5;
        dead-interval 20;
      }
    }
  }
}
```

This topic contains the following examples of configuring a routing protocol:

- Shortcut on page 17
- Longer Configuration on page 17
- Making Changes to a Routing Protocol Configuration on page 20

Shortcut

You can create this entire configuration with two commands:

```
[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/0 hello-interval 5
dead-interval 20
[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/1 hello-interval 5
dead-interval 20
```

Longer Configuration

This section provides a longer example of creating the above OSPF configuration. In the process, it illustrates how to use the different features of the CLI.

1. Enter configuration mode by issuing the **configure** top-level command:

```
user@host> configure
entering configuration mode
[edit]
user@host#
```

Notice that the prompt has changed to a pound sign (#) to indicate configuration mode.

2. To create the above configuration, you start by editing the **protocols ospf** statements:

```
[edit]
user@host# edit protocols ospf
[edit protocols ospf]
user@host#
```

3. Now add the OSPF area:

```
[edit protocols ospf]
user@host# edit area 0.0.0.0
[edit protocols ospf area 0.0.0.0]
user@host#
```

4. Add the first interface:

```
[edit protocols ospf area 0.0.0.0]
user@host# edit interface so0
[edit protocols ospf area 0.0.0.0 interface so-0/0/0]
user@host#
```

You now have four nested statements.

5. Set the hello and dead intervals.

```
[edit protocols ospf area 0.0.0.0 interface so-0/0/0]

user@host# set ?

user@host# set hello-interval 5

user@host# set dead-interval 20

user@host#
```

6. You can see what is configured at the current level with the **show** command:

```
[edit protocols ospf area 0.0.0.0 interface so-0/0/0]
user@host# show
hello-interval 5;
dead-interval 20;
[edit protocols ospf area 0.0.0.0 interface so-0/0/0]
user@host#
```

7. You are finished at this level, so back up a level and take a look at what you have so far:

```
[edit protocols ospf area 0.0.0.0 interface so-0/0/0]
user@host# up
[edit protocols ospf area 0.0.0.0]
user@host# show
interface so-0/0/0 {
    hello-interval 5;
    dead-interval 20;
}
[edit protocols ospf area 0.0.0.0]
user@host#
```

The **interface** statement appears because you have moved to the **area** statement.

8. Add the second interface:

```
[edit protocols ospf area 0.0.0.0]
user@host# edit interface so-0/0/1
```

```

[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# set hello-interval 5
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# set dead-interval 20
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# up
[edit protocols ospf area 0.0.0.0]
user@host# show
interface so-0/0/0 {
    hello-interval 5;
    dead-interval 20;
}
interface so-0/0/1 {
    hello-interval 5;
    dead-interval 20;
}
[edit protocols ospf area 0.0.0.0]
user@host#

```

9. Back up to the top level and see what you have:

```

[edit protocols ospf area 0.0.0.0]
user@host# top
[edit]
user@host# show
protocols {
    ospf {
        area 0.0.0.0 {
            interface so-0/0/0 {
                hello-interval 5;
                dead-interval 20;
            }
            interface so-0/0/1 {
                hello-interval 5;
                dead-interval 20;
            }
        }
    }
}
[edit]
user@host#

```

This configuration now contains the statements you want.

10. Before committing the configuration (and thereby activating it), verify that the configuration is correct:

```

[edit]
user@host# commit check
configuration check succeeds
[edit]
user@host#

```

11. Commit the configuration to activate it on the router:

```

[edit]

```

```

user@host# commit
commit complete
[edit]
user@host#

```

Making Changes to a Routing Protocol Configuration

Suppose you decide to use different dead and hello intervals on interface so-0/0/1. You can make changes to the configuration.

1. Go directly to the appropriate hierarchy level by typing the full hierarchy path to the statement you want to edit.

```

[edit]
user@host# edit protocols ospf area 0.0.0.0 interface so-0/0/1
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# show
hello-interval 5;
dead-interval 20;
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# set hello-interval 7
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# set dead-interval 28
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# top
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
        dead-interval 20;
      }
      interface so-0/0/1 {
        hello-interval 7;
        dead-interval 28;
      }
    }
  }
}
[edit]
user@host#

```

2. If you decide not to run OSPF on the first interface, delete the statement:

```

[edit]
user@host# edit protocols ospf area 0.0.0.0
[edit protocols ospf area 0.0.0.0]
user@host# delete interface so-0/0/0
[edit protocols ospf area 0.0.0.0]
user@host# top
[edit]
user@host# show
protocols {

```



```

ospf {
  area 0.0.0.0 {
    interface so-0/0/1 {
      hello-interval 7;
      dead-interval 28;
    }
  }
}
[edit]
user@host#

```

Everything inside the statement you deleted was deleted with it. You can also eliminate the entire OSPF configuration by simply entering **delete protocols ospf** while at the top level.

3. If you decide to use the default values for the hello and dead intervals on your remaining interface but you want OSPF to run on that interface, delete the hello and dead interval timers:

```

[edit]
user@host# edit protocols ospf area 0.0.0.0 interface so-0/0/1
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# delete hello-interval
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# delete dead-interval
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# top
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/1;
    }
  }
}
[edit]
user@host#

```

You can set multiple statements at the same time as long as they are all part of the same hierarchy (the path of statements from the top inward, as well as one or more statements at the bottom of the hierarchy). This feature can reduce considerably the number of commands you must enter.

4. To go back to the original hello and dead interval timers on interface **so-0/0/1**, enter:

```

[edit]
user@host# edit protocols ospf area 0.0.0.0 interface so-0/0/1
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# set hello-interval 5 dead-interval 20
[edit protocols ospf area 0.0.0.0 interface so-0/0/1]
user@host# exit
[edit]
user@host# show

```

```

protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/1 {
        hello-interval 5;
        dead-interval 20;
      }
    }
  }
}
[edit]
user@host#

```

5. You also can re-create the other interface, as you had it before, with only a single entry:

```

[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/1 hello-interval 5
dead-interval 20
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
        dead-interval 20;
      }
      interface so-0/0/1 {
        hello-interval 5;
        dead-interval 20;
      }
    }
  }
}
[edit]
user@host#

```

Related Topics ■ Getting Started with the JUNOS Command-Line Interface on page 9

Rolling Back JUNOS Configuration Changes

This topic shows how to use the **rollback** command to return to the most recently committed JUNOS configuration. The **rollback** command is useful if you make configuration changes and then decide not to keep the changes.

The following procedure shows how to configure an SNMP health monitor on a device running JUNOS Software and then return to the most recently committed configuration that does not include the health monitor. When configured, the SNMP health monitor provides network management system (NMS) with predefined monitoring for file system usage, CPU usage, and memory usage on the device.

1. Enter configuration mode:

```

user@host> configure
entering configuration mode
[edit]
user@host#

```

2. Show the current configuration (if any) for SNMP:

```

[edit]
user@host# show snmp

```

No `snmp` statements appear. SNMP has not been configured on the router.

3. Configure the health monitor:

```

[edit]
user@host# set snmp health-monitor

```

4. Show the new configuration:

```

[edit]
user@host# show snmp
health-monitor;

```

The `health-monitor` statement indicates that SNMP health monitoring is configured on the router.

5. Use the `rollback` configuration mode command to return to the most recently committed configuration:

```

[edit]
user@host# rollback
load complete

```

6. Show the configuration again to make sure your change is no longer present:

```

[edit]
user@host# show snmp

```

No `snmp` configuration statements appear. The health monitor is no longer configured.

7. Use the `commit` command to activate the configuration to which you rolled back:

```

[edit]
user@host# commit

```

8. Exit configuration mode:

```

[edit]
user@host# exit
Exiting configuration mode

```

You can also use the `rollback` command to return to earlier configurations. For more information, see “Managing Configurations” on page 99.

Chapter 3

Getting Online Help

This chapter describes the JUNOS command-line interface (CLI) online help.

Topics include:

- Getting Online Help from the JUNOS Command-Line Interface on page 25
- JUNOS Command-Line Interface Online Help Features on page 28
- Examples: Using CLI Command Completion on page 30
- Examples: Using Command Completion in Configuration Mode on page 31
- Displaying the JUNOS CLI Command and Word History on page 32

Getting Online Help from the JUNOS Command-Line Interface

The JUNOS command-line interface (CLI) has a context-sensitive online help feature that enables you to access information about commands and statements from the JUNOS CLI. This topic contains the following sections:

- Getting Help About Commands on page 26
- Getting Help About a String in a Statement or Command on page 27
- Getting Help About Configuration Statements on page 27
- Getting Help About System Log Messages on page 28

Getting Help About Commands

Information about commands is provided at each level of the CLI command hierarchy. You can type a question mark to get help about commands:

- If you type the question mark at the command-line prompt, the CLI lists the available commands and options. For example, to view a list of top-level operational mode commands, type a question mark (?) at the command-line prompt.

```
user@host> ?

Possible completions:
clear          Clear information in the system
configure      Manipulate software configuration information
file           Perform file operations
help           Provide help information
mtrace         Trace mtrace packets from source to receiver.
monitor        Real-time debugging
ping           Ping a remote target
quit           Exit the management session
request        Make system-level requests
restart        Restart a software process
set            Set CLI properties, date, time, craft display text
show           Show information about the system
ssh            Open a secure shell to another host
start          Start a software process
telnet         Telnet to another host
test           Diagnostic debugging commands
traceroute     Trace the route to a remote host
user@host>
```

- If you type the question mark after entering the complete name of a command or command option, the CLI lists the available commands and options and then redisplay the command names and options that you typed.

```
user@host> clear ?

Possible completions:
arp            Clear address-resolution information
bgp            Clear BGP information
chassis        Clear chassis information
firewall       Clear firewall counters
igmp           Clear IGMP information
interfaces     Clear interface information
ilmi           Clear ILMI statistics information
isis           Clear IS-IS information
ldp            Clear LDP information
log            Clear contents of a log file
mpls           Clear MPLS information
msdp           Clear MSDP information
multicast      Clear Multicast information
ospf           Clear OSPF information
pim            Clear PIM information
rip            Clear RIP information
route          Clear routing table information
rsvp           Clear RSVP information
```

```
snmp      Clear SNMP information
system    Clear system status
vrrp      Clear VRRP statistics information
user@host> clear
```

- If you type the question mark in the middle of a command name, the CLI lists possible command completions that match the letters you have entered so far. It then redisplay the letters that you typed. For example, to list all operational mode commands that start with the letter *c*, type the following:

```
user@host> c?

Possible completions:
clear      Clear information in the system
configure  Manipulate software configuration information
user@host> c
```

- For introductory information on using the question mark or the help command, you can also type **help** and press Enter:

```
user@host> help
```

Getting Help About a String in a Statement or Command

You can use the **help** command to display help about a text string contained in a statement or command name:

```
help apropos string
```

string is a text string about which you want to get help. This string is used to match statement or command names as well as to match the help strings that are displayed for the statements or commands.

If the string contains spaces, enclose it in quotation marks (" "). You can also specify a regular expression for the string, using standard UNIX-style regular expression syntax.

In configuration mode, this command displays statement names and help text that match the string specified. In operational mode, this command displays command names and help text that match the string specified.

Getting Help About Configuration Statements

You can display help based on text contained in a statement name using the **help topic** and **help reference** commands:

```
help topic word
help reference statement-name
```

The **help topic** command displays usage guidelines for the statement based on information that appears in the JUNOS configuration guides. The **help reference**

command displays summary information about the statement based on the summary descriptions that appear in the JUNOS configuration guides.

Getting Help About System Log Messages

You can display help based on a system log tag using the **help syslog** command:

```
help syslog syslog-tag
```

The **help syslog** command displays the contents of a system log message.

- Related Topics**
- JUNOS Command-Line Interface Online Help Features on page 28
 - Getting Started with the JUNOS Command-Line Interface on page 9

JUNOS Command-Line Interface Online Help Features

The JUNOS command-line interface online help provides you the following features for ease of use and error prevention:

- Help for Omitted Statements on page 28
- Using CLI Command Completion on page 29
- Using Command Completion in Configuration Mode on page 29
- Displaying Tips About CLI Commands on page 29

Help for Omitted Statements

If you have omitted a required statement at a particular hierarchy level, when you attempt to move from that hierarchy level or when you issue the **show** command in configuration mode, a message indicates which statement is missing. For example:

```
[edit protocols pim interface so-0/0/0]
user@host# top
Warning: missing mandatory statement: 'mode'
[edit]
user@host# show
protocols {
  pim {
    interface so-0/0/0 {
      priority 4;
      version 2;
      # Warning: missing mandatory statement(s): 'mode'
    }
  }
}
```


Using CLI Command Completion

The JUNOS CLI provides you a command completion option that enables JUNOS Software to recognize commands and options based on the initial few letters you typed. That is, you do not always have to remember or type the full command or option name for the CLI to recognize it.

- To display all possible command or option completions, type the partial command followed immediately by a question mark.
- To complete a command or option that you have partially typed, press Tab or the Spacebar. If the partially typed letters begin a string that uniquely identifies a command, the complete command name appears. Otherwise, a prompt indicates that you have entered an ambiguous command, and the possible completions are displayed.

Command completion also applies to other strings, such as filenames, interface names, and usernames. To display all possible values, type a partial string followed immediately by a question mark. To complete a string, press Tab..

Using Command Completion in Configuration Mode

The CLI command completion functions also apply to the commands in configuration mode and to configuration statements. Specifically, to display all possible commands or statements, type the partial string followed immediately by a question mark; to complete a command or statement that you have partially typed, press Tab or the Spacebar.

Command completion also applies to identifiers, with one slight difference. To display all possible identifiers, type a partial string followed immediately by a question mark. To complete an identifier, you must press Tab. This scheme allows you to enter identifiers with similar names; then press the Spacebar when you are done typing the identifier name.

Displaying Tips About CLI Commands

To get tips about CLI commands, issue the `help tip cli` command. Each time you enter the command, a new tip appears. For example:

```
user@host> help tip cli
JUNOS tip:
Use 'request system software validate' to validate the incoming software
against the current configuration without impacting the running system.
user@host> help tip cli
JUNOS tip:
Use 'commit and-quit' to exit configuration mode after the commit has
succeeded. If the commit fails you are left in configuration mode.
```

You can also enter `help tip cli number` to associate a tip with a number. This enables you to recall the tip at a later time. For example:

```

user@host> help tip cli 10
JUNOS tip:
Use '#' in the beginning of a line in command scripts to cause the
rest of the line to be ignored.

user@host> help tip cli
JUNOS tip:
Use the 'apply-groups' statement at any level of the configuration
hierarchy to inherit configuration statements from a configuration group.

user@host>

```

Examples: Using CLI Command Completion

The following examples show how you can use the command completion feature in JUNOS Software. Issue the `show interfaces` command:

```

user@host> sh<Space>ow i<Space>
'i' is ambiguous.
Possible completions:
igmp          Show information about IGMP
interface     Show interface information
isis          Show information about IS-IS
user@host> show in<Space>terfaces
Physical interface: at-0/1/0, Enabled, Physical link is Up
Interface index: 11, SNMP ifIndex: 65
Link-level type: ATM-PVC, MTU: 4482, Clocking: Internal, SONET mode
Speed: OC12, Loopback: None, Payload scrambler: Enabled
Device flags: Present Running
Link flags: 0x01
...
user@host>

```

Display a list of all log files whose names start with the string “messages,” and then display the contents of one of the files:

```

user@myhost> show log mes?
Possible completions:
<filename>Log file to display
messagesSize: 1417052, Last changed: Mar 3 00:33
messages.0.gzSize: 145575, Last changed: Mar 3 00:00
messages.1.gzSize: 134253, Last changed: Mar 2 23:00
messages.10.gzSize: 137022, Last changed: Mar 2 14:00
messages.2.grSize: 137112, Last changed: Mar 2 22:00
messages.3.gzSize: 121633, Last changed: Mar 2 21:00
messages.4.gzSize: 135715, Last changed: Mar 2 20:00
messages.5.gzSize: 137504, Last changed: Mar 2 19:00
messages.6.gzSize: 134591, Last changed: Mar 2 18:00
messages.7.gzSize: 132670, Last changed: Mar 2 17:00
messages.8.gzSize: 136596, Last changed: Mar 2 16:00
messages.9.gzSize: 136210, Last changed: Mar 2 15:00
user@myhost> show log mes<Tab>sages.4<Tab>.gz<Enter>
Jan 15 21:00:00 myhost newsyslog[1381]: logfile turned over
...

```

Examples: Using Command Completion in Configuration Mode

List the configuration mode commands:

```
[edit]
user@host# ?
<[Enter]>      Execute this command
activate       Remove the inactive tag from a statement
annotate       Annotate the statement with a comment
commit         Commit current set of changes
copy           Copy a statement
deactivate     Add the inactive tag to a statement
delete         Delete a data element
edit           Edit a sub-element
exit           Exit from this level
extension      Extension operations
help           Provide help information
insert         Insert a new ordered data element
load           Load configuration from ASCII file
quit           Quit from this level
rename         Rename a statement
replace        Replace character string in configuration
rollback       Roll back to previous committed configuration
run            Run an operational-mode command
save           Save configuration to ASCII file
set            Set a parameter
show           Show a parameter
status         Show users currently editing configuration
top            Exit to top level of configuration
up             Exit one level of configuration
wildcard       Wildcard operations
[edit]user@host#
```

List all the statements available at a particular hierarchy level:

```
[edit]
user@host# edit ?
Possible completions:
> accounting-options  Accounting data configuration
> chassis             Chassis configuration
> class-of-service    Class-of-service configuration
> firewall            Define a firewall configuration
> forwarding-options  Configure options to control packet sampling
> groups              Configuration groups
> interfaces           Interface configuration
> policy-options      Routing policy option configuration
> protocols           Routing protocol configuration
> routing-instances   Routing instance configuration
> routing-options     Protocol-independent routing option configuration
> snmp                Simple Network Management Protocol
> system              System parameters
user@host# edit protocols ?
Possible completions:
<[Enter]>      Execute this command
> bgp          BGP options
> connections  Circuit cross-connect configuration
> dvmrp        DVMRP options
```

```

> igmp          IGMP options
> isis          IS-IS options
> ldp           LDP options
> mpls          Multiprotocol Label Switching options
> msdp          MSDP options
> ospf          OSPF configuration
> pim           PIM options
> rip           RIP options
> router-discovery ICMP router discovery options
> rsvp          RSVP options
> sapSession    Advertisement Protocol options
> vrrp          VRRP options
|              Pipe through a command
[edit]
user@host# edit protocols

```

List all commands that start with a particular letter or string:

```

user@host# edit routing-options a?
Possible completions:
> aggregate      Coalesced routes
> autonomous-system Autonomous system number
[edit]
user@host# edit routing-options a

```

List all configured Asynchronous Transfer Mode (ATM) interfaces:

```

[edit]
user@host# edit interfaces at?
<interface_name>      Interface name
  at-0/2/0              Interface name
  at-0/2/1              Interface name
[edit]
user@host# edit interfaces at

```

Display a list of all configured policy statements:

```

[edit]
user@host# show policy-options policy-statement ?

Possible completions:
  <policy_name>      Name to identify a policy filter
[edit]
user@host# show policy-options policy-statement

```

Displaying the JUNOS CLI Command and Word History

The following sections show you how to view the JUNOS command-line interface command and word history:

Displaying Command History

To display a list of recent commands that you issued, use the `show cli history` command:

```

user@host> show cli history
03-03 01:00:50 – show cli history

```

```

03-03 01:01:12 – show interfaces terse
03-03 01:01:22 – show interfaces lo0
03-03 01:01:44 – show bgp next-hop-database
03-03 01:01:51 – show cli history

```

By default, this command displays the last 100 commands issued in the CLI. If you specify a number with the command, it displays that number of recent commands. For example:

```

user@host> show cli history 3
01:01:44 – show bgp next-hop-database
01:01:51 – show cli history
01:02:51 – show cli history 3

```

Displaying Word History You can type Esc + . (period) or Alt + . (period) to insert the last word of the previous command. Repeat Esc + . or Alt + . to scroll backwards through the list of recently entered words. For example:

```

user@host> show interfaces terse fe-0/0/0
Interface      Admin  Link  Proto  Local  Remote
fe-0/0/0       up     up
fe-0/0/0.0     up     up     inet   192.168.220.1/30
user@host> <Esc>
user@host> fe-0/0/0

```

If you scroll completely to the beginning of the list, typing Esc + . or Alt + . again restarts scrolling from the last word entered.

Part 2

Operational Mode and Configuration Mode

- Using CLI Operational Commands to Monitor the Router on page 37
- Using Commands and Statements to Configure a Device Running JUNOS Software on page 61
- Managing Configurations on page 99
- Filtering Command Output on page 117
- Controlling the CLI Environment on page 127

Chapter 4

Using CLI Operational Commands to Monitor the Router

This chapter provides information about CLI operational commands.

Topics include:

- Overview of JUNOS CLI Operational Mode Commands on page 37
- JUNOS Operational Mode Commands That Combine Other Commands on page 40
- Understanding brief, detail, extensive, and terse Options of JUNOS Operational Commands on page 41
- Controlling the Scope of a Command on page 42
- Monitoring Who Uses the CLI on page 45
- Interface Naming Conventions Used in JUNOS Operational Commands on page 46
- Viewing Files and Directories on a Device Running JUNOS Software on page 47
- Displaying JUNOS Software Information on page 52
- Managing Programs and Processes on page 54
- Using the JUNOS Comment Character # on page 59
- Example: Using Comments in JUNOS Commands on page 59

Overview of JUNOS CLI Operational Mode Commands

This topic provides an overview of JUNOS operational mode commands, and contains the following sections:

- CLI Command Categories on page 37
- Commonly Used Operational Mode Commands on page 39

CLI Command Categories

When you log in to a device running JUNOS Software and the CLI starts, there are a number of broad groups of CLI commands:

- Commands for controlling the CLI environment—The commands in the **set** hierarchy configure the CLI display screen. For information about these commands, see “Controlling the CLI Environment” on page 127.
- Commands for monitoring and troubleshooting—The following commands display information and statistics about the software and test network connectivity. Detailed command descriptions are provided in the *JUNOS Interfaces Command Reference*.
 - **clear**—Clear statistics and protocol database information.
 - **mtrace**—Trace mtrace packets from source to receiver.
 - **monitor**—Perform real-time debugging of various software components, including the routing protocols and interfaces.
 - **ping**—Determine the reachability of a remote network host.
 - **show**—Display the current configuration and information about interfaces, routing protocols, routing tables, routing policy filters, system alarms, and the chassis.
 - **test**—Test the configuration and application of policy filters and autonomous system (AS) path regular expressions.
 - **traceroute**—Trace the route to a remote network host.
- Commands for connecting to other network systems—The **ssh** command opens secure shell connections, and the **telnet** command opens telnet sessions to other hosts on the network. For information about these commands, see the *JUNOS System Basics and Services Command Reference*.
- Commands for copying files—The **copy** command copies files from one location on the router to another, from the router to a remote system, or from a remote system to the router. For information about these commands, see the *JUNOS System Basics and Services Command Reference*.
- Commands for restarting software processes—The commands in the **restart** hierarchy restart the various JUNOS Software processes, including the routing protocol, interface, and SNMP. For information about these commands, see the *JUNOS System Basics and Services Command Reference*.
- A command—**request**—for performing system-level operations, including stopping and rebooting the router and loading JUNOS Software images. For information about this command, see the *JUNOS System Basics and Services Command Reference*.
- A command—**start**—to exit the CLI and start a UNIX shell. For information about this command, see the *JUNOS System Basics and Services Command Reference*.
- A command—**configure**—for entering configuration mode, which provides a series of commands that configure JUNOS Software, including the routing protocols, interfaces, network management, and user access. For information about the CLI configuration commands, see “Using Commands and Statements to Configure a Device Running JUNOS Software” on page 61.

- A command—**quit**—to exit the CLI. For information about this command, see the *JUNOS System Basics and Services Command Reference*.
- For more information about the CLI operational mode commands, see the *JUNOS Interfaces Command Reference* and the *JUNOS System Basics and Services Command Reference*.

Commonly Used Operational Mode Commands

Table 4 on page 39 lists some operational commands you may find useful for monitoring router operation. For a complete description of operational commands, see the JUNOS command references.

Table 4: Commonly Used Operational Mode Commands

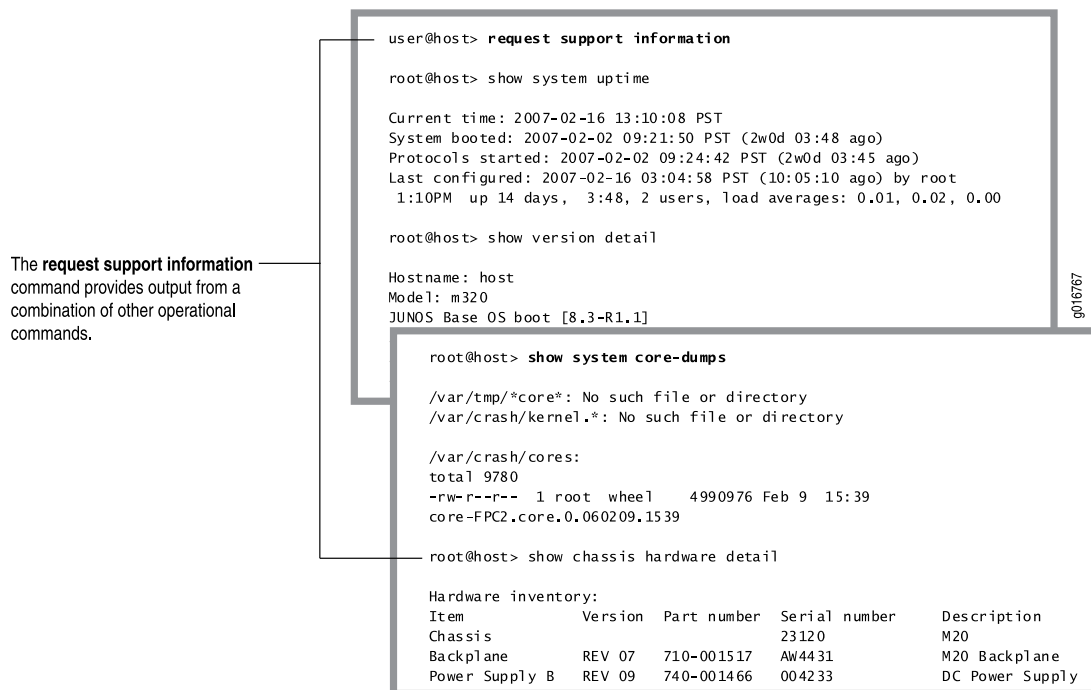
Items to Check	Description	Command
Software version	Versions of software running on the router	<code>show version</code>
Log files	Contents of the log files	<code>monitor</code>
	Log files and their contents and recent user logins	<code>show log</code>
Remote systems	Host reachability and network connectivity	<code>ping</code>
	Route to a network system	<code>traceroute</code>
Configuration	Current system configuration	<code>show configuration</code>
Manipulate files	List of files and directories on the router	<code>file list</code>
	Contents of a file	<code>file show</code>
Interface information	Detailed information about interfaces	<code>show interfaces</code>
Chassis	Chassis alarm status	<code>show chassis alarms</code>
	Information currently on craft display	<code>show chassis craft-interface</code>
	Router environment information	<code>show chassis environment</code>
	Hardware inventory	<code>show chassis hardware</code>
Routing table information	Information about entries in the routing tables	<code>show route</code>
Forwarding table information	Information about data in the kernel's forwarding table	<code>show route forwarding-table</code>
IS-IS	Adjacent routers	<code>show isis adjacency</code>
OSPF	Display standard information about OSPF neighbors	<code>show ospf neighbor</code>
BGP	Display information about BGP neighbors	<code>show bgp neighbor</code>

Table 4: Commonly Used Operational Mode Commands *(continued)*

Items to Check	Description	Command
MPLS	Status of interfaces on which MPLS is running	show mpls interface
	Configured LSPs on the router, as well as all ingress, transit, and egress LSPs	show mpls lsp
	Routes that form a label-switched path	show route label-switched-path
RSVP	Status of interfaces on which RSVP is running	show rsvp interface
	Currently active RSVP sessions	show rsvp session
	RSVP packet and error counters	show rsvp statistics

JUNOS Operational Mode Commands That Combine Other Commands

In some cases, some JUNOS operational commands are created from a combination of other operational commands. These commands can be useful shortcuts for collecting information about the device. (See Figure 5 on page 40.)

Figure 5: Commands That Combine Other Commands

Understanding brief, detail, extensive, and terse Options of JUNOS Operational Commands

The JUNOS operational mode commands can include brief, detail, extensive, or terse options. You can use these options to control the amount of information you want to view.

1. Use the ? prompt to list options available for the command. For example:

```
user@host> show interfaces fe-1/1/1 ?
```

Possible completions:

<[Enter]>	Execute this command
brief	Display brief output
descriptions	Display interface description strings
detail	Display detailed output
extensive	Display extensive output
media	Display media information
snmp-index	SNMP index of interface
statistics	Display statistics and detailed output
terse	Display terse output
	Pipe through a command

2. Choose the option you wish to use with the command. (See Figure 6 on page 42.)

Figure 6: Command Output OptionsCommand output with the **brief** option.

```

user@host> show interfaces fe-1/1/1 brief
Physical interface: fe-1/1/1, Enabled, Physical link is Down
Link-level type: Ethernet, MTU: 1514, Speed: 100mbps, Loopback:
Disabled, Source filtering: Disabled,
Flow control: Enabled
Device flags : Present Running Down
Interface flags: Hardware-Down SNMP-Traps Internal: 0x4000
Link flags : None

```

Command output with the **terse** option.

```

user@host> show interfaces fe-1/1/1 terse
Interface      Admin Link Proto  Local      Remote
fe-1/1/1       up    down

```

Command output with the **extensive** option.

```

user@host> show interfaces fe-1/1/1 extensive
Physical interface: fe-1/1/1, Enabled, Physical link is Down
Interface index: 141, SNMP ifIndex: 33, Generation: 24
Link-level type: Ethernet, MTU: 1514, Speed: 100mbps, Loopback:
Disabled, Source filtering: Disabled,
Flow control: Enabled
Device flags : Present Running Down
Interface flags: Hardware-Down SNMP-Traps Internal: 0x4000
Link flags : None
CoS queues : 4 supported, 4 maximum usable queues
Hold-times : Up 0 ms, Down 0 ms
Current address: 00:90:69:d0:f8:9e, Hardware address: 00:90:69:d0:f8:9e
Last flapped : 2007-02-02 09:26:25 PST (2w0d 03:40 ago)
Statistics last cleared: Never
Traffic statistics:
Input bytes :                0                0 bps
Output bytes :                0                0 bps
Input packets:                0                0 pps
Output packets:                0                0 pps
---(more)---

```

Controlling the Scope of a Command

The JUNOS CLI operational commands include options that you can use to identify specific components on a device running JUNOS Software. For example:

1. Type the **show interfaces** command to display information about all interfaces on the router.

```

user@host> show interfaces
Physical interface: so-0/0/0, Enabled, Physical link is Up
Interface index: 128, SNMP ifIndex: 23
Link-level type: PPP, MTU: 4474, Clocking: Internal, SONET mode, Speed:
OC3,
Loopback: None, FCS: 16, Payload scrambler: Enabled
Device flags : Present Running
Interface flags: Point-To-Point SNMP-Traps Internal: 0x4000
Link flags : Keepalives
Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
Keepalive: Input: 13861 (00:00:05 ago), Output: 13891 (00:00:01 ago)
LCP state: Opened
NCP state: inet: Opened, inet6: Not-configured, iso: Opened, mp1s:
Not-configured
CHAP state: Closed
PAP state: Closed
CoS queues : 4 supported, 4 maximum usable queues
Last flapped : 2008-06-02 17:16:14 PDT (1d 14:21 ago)
Input rate : 40 bps (0 pps)
Output rate : 48 bps (0 pps)

```

---(more)---

2. To display information about a specific interface, type that interface as a command option:

```
user@host> show interfaces fe-0/1/3
Physical interface: fe-0/1/3, Enabled, Physical link is Up
  Interface index: 135, SNMP ifIndex: 30
  Link-level type: Ethernet, MTU: 1514, Speed: 100mbps, MAC-REWRITE Error:
None,
  Loopback: Disabled, Source filtering: Disabled, Flow control: Enabled
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Link flags     : None
  CoS queues    : 4 supported, 4 maximum usable queues
  Current address: 00:05:85:8f:c8:22, Hardware address: 00:05:85:8f:c8:22

  Last flapped   : 2008-06-02 17:16:15 PDT (1d 14:28 ago)
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Active alarms  : None
  Active defects : None

user@host>
```

For more information on specifying interface names as command options, see “Using the JUNOS Comment Character #” on page 59.

Operational Mode Commands on a TX Matrix Router

When you issue operational mode commands on the TX Matrix router, CLI command options allow you to restrict the command output to show only a component of the routing matrix rather than the routing matrix as a whole.

These are the options shown in the CLI:

- `scc`—The TX Matrix router.
- `lcc number`—A specific T640 router.
- `all-lcc`—All T640 routers.

If you specify none of these options, then the command applies by default to the whole routing matrix: the TX Matrix router and all connected T640 routers.

Examples of Routing Matrix Command Options

The following output samples, using the `show version` command, demonstrate some different options for viewing information about the routing matrix.

```
user@host> show version ?
Possible completions:
  <[Enter]>          Execute this command
```

all-lcc	Show software version on all LCC chassis
brief	Display brief output
detail	Display detailed output
lcc	Show software version on specific LCC (0..3)
scc	Show software version on the SCC
	Pipe through a command

**Sample Output: No
Routing Matrix Options
Specified**

```

user@host> show version
scc-re0:
-----
Hostname: scc
Model: TX Matrix
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
lcc0-re0:
-----
Hostname: lcc0
Model: t640
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
JUNOS Support Tools Package [7.0-20040630.0]
lcc1-re0:
-----
Hostname: lcc1
Model: t640
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
JUNOS Support Tools Package [7.0-20040630.0]

```

**Sample Output: TX
Matrix Router Only (scc
Option)**

```

user@host> show version scc
Hostname: scc
Model: TX Matrix
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]

```

**Sample Output: Specific
T640 Router (lcc
number Option)**

```

user@host> show version lcc 0
lcc0-re0:
-----
Hostname: lcc0
Model: t640

```



```

JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
JUNOS Support Tools Package [7.0-20040630.0]

```

**Sample Output: All T640
Routers
(all-icc Option)**

```

user@host> show version all-icc
lcc0-re0:
-----
Hostname: lcc0
Model: t640
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
JUNOS Support Tools Package [7.0-20040630.0]
lcc1-re0:
-----
Hostname: lcc1
Model: t640
JUNOS Base OS boot [7.0-20040630.0]
JUNOS Base OS Software Suite [7.0-20040629.0]
JUNOS Kernel Software Suite [7.0-20040630.0]
JUNOS Packet Forwarding Engine Support (T-Series) [7.0-20040630.0]
JUNOS Routing Software Suite [7.0-20040630.0]
JUNOS Online Documentation [7.0-20040630.0]
JUNOS Crypto Software Suite [7.0-20040630.0]
JUNOS Support Tools Package [7.0-20040630.0]

```

Monitoring Who Uses the CLI

Depending upon how you configure JUNOS Software, multiple users can log in to the router, use the CLI, and configure or modify the software configuration.

If, when you enter configuration mode, another user is also in configuration mode, a notification message is displayed that indicates who the user is and what portion of the configuration the person is viewing or editing:

```

user@host> configure
Entering configuration mode
Users currently editing the configuration:
  root terminal d0 (pid 4137) on since 2008-04-09 23:03:07 PDT, idle 7w6d 08:22
  [edit]
The configuration has been changed but not committed

[edit]
user@host#

```

Interface Naming Conventions Used in JUNOS Operational Commands

This topic explains the interface naming conventions used in JUNOS operational commands, and contains the following sections:

- Physical Part of an Interface Name on page 46
- Logical Part of an Interface Name on page 46
- Channel Identifier Part of an Interface Name on page 47

Physical Part of an Interface Name

The M Series and T Series routers use one convention for interface naming, whereas the J Series router uses another.

- M Series and T Series interface names—On the M Series and T Series routers, when you display information about an interface, you specify the interface type, the slot in which the Flexible PIC Concentrator (FPC) is installed, the slot on the FPC in which the PIC is located, and the configured port number.

In the physical part of the interface name, a hyphen (-) separates the media type from the FPC number, and a slash (/) separates the FPC, PIC, and port numbers:

type-fpc/pic/port



NOTE: Exceptions to the *type-fpc/pic/port* physical description include the aggregated Ethernet and aggregated SONET/SDH interfaces, which use the syntax *aenumber* and *asnumber*, respectively.

- J Series interface names—On the J Series router, when you display information about an interface, you specify the interface type, the slot in which the Physical Interface Module (PIM) is installed, 0, and the configured port number.

In the physical part of the interface name, a hyphen (-) separates the media type from the PIM number, and a slash (/) separates the PIM, 0, and port numbers:

type-pim/0/port



NOTE: An exception to the *type-pim/0/port* physical description is the ISDN dialer interface, which uses the syntax *dlnumber*.

Logical Part of an Interface Name

The logical unit part of the interface name corresponds to the logical unit number, which can be a number from 0 through 16,384. In the virtual part of the name, a period (.) separates the port and logical unit numbers:

- M Series and T Series routers:

type-fpc/pic/port.logical

- J Series router:

type-pim/0/port.logical

Channel Identifier Part of an Interface Name

The channel identifier part of the interface name is required only on channelized interfaces. For channelized interfaces, channel 0 identifies the first channelized interface. For channelized intelligent queuing (IQ) interfaces, channel 1 identifies the first channelized interface.



NOTE: Depending on the type of channelized interface, up to three levels of channelization can be specified. For more information, see the *JUNOS Network Interfaces Configuration Guide*.

A colon (:) separates the physical and virtual parts of the interface name:

- M Series and T Series routers:

type-fpc/pic/port:channel

type-fpc//pic/port:channel:channel

type-fpc/pic/port:channel:channel:channel

- J Series routers:

type-pim/0/port:channel

type-pim/0/port:channel:channel

type-pim/0/port:channel:channel:channel

Viewing Files and Directories on a Device Running JUNOS Software

JUNOS Software stores information in files on the device, including configuration files, log files, and router software files. This topic shows some examples of operational commands that you can use to view files and directories on a device running JUNOS Software.

Sections include:

- Directories on the Router on page 48
- Listing Files and Directories on page 48
- Specifying Filenames and URLs on page 51

Directories on the Router

Table 5 on page 48 lists some standard directories on a device running JUNOS Software.

Table 5: Directories on the Router

Directory	Description
/config	This directory is located on the router's internal flash drive. It contains the active configuration (<code>juniper.conf</code>) and rollback files 1, 2, and 3.
/var/db/config	This directory is located on the router's hard drive and contains rollback files 4 through 49.
/var/tmp	This directory is located on the router's hard drive. It holds core files from the various processes on the Routing Engines. Core files are generated when a particular process crashes and are used by Juniper Networks engineers to diagnose the reason for failure.
/var/log	This directory is located on the router's hard drive. It contains files generated by both the router's logging function as well as the <code>traceoptions</code> command.
/var/home	This directory is located on the router's hard drive. It contains a subdirectory for each configured user on the router. These individual user directories are the default file location for many JUNOS Software commands.
/altroot	This directory is located on the router's hard drive and contains a copy of the root file structure from the internal flash drive. This directory is used in certain disaster recovery modes where the internal flash drive is not operational.
/altconfig	This directory is located on the router's hard drive and contains a copy of the <code>/config</code> file structure from the internal flash drive. This directory is also used in certain disaster recovery modes when the internal flash drive is not operational.

Listing Files and Directories

You can view the device's directory structure as well as individual files by issuing the `file` command in operational mode.

1. To get help about the **file** command, type the following:

```

user@host> file ?
Possible completions:
<[Enter]>      Execute this command
archive        Archives files from the system
checksum       Calculate file checksum
compare        Compare files
copy           Copy files (local or remote)
delete         Delete files from the system
list           List file information
rename         Rename files
show           Show file contents
source-address Local address to use in originating the connection

|
user@host> file

```

Help shows that the **file** command includes several options for manipulating files.

2. Use the **list** option to see the directory structure of the router. For example, to show the files located in your home directory on the router:

```

user@host> file list
.ssh/
common

```

The default directory for the **file list** command is the home directory of the user logged in to the router. In fact, the user's home directory is the default directory for most of JUNOS Software commands requiring a filename.

3. To view the contents of other file directories, specify the directory location. For example:

```

user@host> file list /config
juniper.conf
juniper.conf.1.gz
juniper.conf.2.gz
juniper.conf.3.gz

```

4. You can also use the router's context-sensitive help system to locate a directory. For example:

```

user@host> file list /?
Possible completions:
<[Enter]>      Execute this command
<path>        Path to list
/COPYRIGHT     Size: 6355, Last changed: Feb 13 2005
/altconfig/    Last changed: Aug 07 2007
/altroot/      Last changed: Aug 07 2007
/bin/          Last changed: Apr 09 22:31:35
/boot/         Last changed: Apr 09 23:28:39
/config/       Last changed: Apr 16 22:35:35
/data/         Last changed: Aug 07 2007
/dev/          Last changed: Apr 09 22:36:21
/etc/          Last changed: Apr 11 03:14:22

```

```

/kernel                               Size: 27823246, Last changed: Aug 07 2007
/mfs/                                  Last changed: Apr 09 22:36:49
/mnt/                                  Last changed: Jan 11 2007
/modules/                              Last changed: Apr 09 22:33:54
/opt/                                  Last changed: Apr 09 22:31:00
/packages/                             Last changed: Apr 09 22:34:38
/proc/                                 Last changed: May 07 20:25:46
/rdm.taf                               Size: 498, Last changed: Apr 09 22:37:31
/root/                                 Last changed: Apr 10 02:19:45
/sbin/                                 Last changed: Apr 09 22:33:55
/staging/                              Last changed: Apr 09 23:28:41
/tmp/                                  Last changed: Apr 11 03:14:49
/usr/                                  Last changed: Apr 09 22:31:34
/var/                                  Last changed: Apr 09 22:37:30
user@host> file list /var/?
<[Enter]>                               Execute this command
<path>                                Path to list
/var/account/                          Last changed: Jul 09 2007
/var/at/                               Last changed: Jul 09 2007
/var/backups/                          Last changed: Jul 09 2007
/var/bin/                              Last changed: Jul 09 2007
/var/crash/                            Last changed: Apr 09 22:31:08
/var/cron/                             Last changed: Jul 09 2007
/var/db/                               Last changed: May 07 20:28:40
/var/empty/                            Last changed: Jul 09 2007
/var/etc/                              Last changed: Apr 16 22:35:36
/var/heimdal/                          Last changed: Jul 10 2007
/var/home/                             Last changed: Apr 09 22:59:18
/var/jail/                             Last changed: Oct 31 2007
/var/log/                              Last changed: Apr 17 02:00:10
/var/mail/                             Last changed: Jul 09 2007
/var/messages/                         Last changed: Jul 09 2007
/var/named/                            Last changed: Jul 10 2007
/var/packages/                         Last changed: Jan 18 02:38:59
/var/pdb/                              Last changed: Oct 31 2007
/var/preserve/                         Last changed: Jul 09 2007
/var/run/                              Last changed: Apr 17 02:00:01
/var/rundb/                            Last changed: Apr 17 00:46:00
/var/rwho/                             Last changed: Jul 09 2007
/var/sdb/                              Last changed: Apr 09 22:37:31
/var/spool/                            Last changed: Jul 09 2007
/var/sw/                               Last changed: Jul 09 2007
/var/tmp/                              Last changed: Apr 09 23:28:41
/var/transfer/                         Last changed: Jul 09 2007
/var/yp/                               Last changed: Jul 09 2007
user@host> file list /var/

```

5. You can also display the contents of a file. For example:

```

user@host> file show /var/log/inventory
Jul  9 23:17:46 CHASSISD release 8.4I0 built by builder on 2007-06-12
07:58:27 UTC
Jul  9 23:18:05 CHASSISD release 8.4I0 built by builder on 2007-06-12
07:58:27 UTC
Jul  9 23:18:06 Routing Engine 0 - part number 740-003239, serial number
9000016755
Jul  9 23:18:15 Routing Engine 1 - part number 740-003239, serial number
9001018324
Jul  9 23:19:03 SSB 0 - part number 710-001951, serial number AZ8025

```

```

Ju1  9 23:19:03 SSRAM bank 0 - part number 710-001385, serial number 243071
Ju1  9 23:19:03 SSRAM bank 1 - part number 710-001385, serial number 410608
...

```

Specifying Filenames and URLs

In some CLI commands and configuration statements—including file copy, file archive, load, save, set system login user *username* authentication *load-key-file*, and request system software add—you can include a filename. On a routing matrix, you can include chassis information as part of the filename (for example, *lcc0*, *lcc0-re0*, or *lcc0-re1*).

You can specify a filename or URL in one of the following ways:

- *filename*—File in the user's current directory on the local flash drive. You can use wildcards to specify multiple source files or a single destination file. Wildcards are not supported in Hypertext Transfer Protocol (HTTP) or FTP.



NOTE: Wildcards are supported only by the file (compare | copy | delete | list | rename | show) commands. When you issue the file show command with a wildcard, it must resolve to one filename.

- *path/filename*—File on the local flash disk.
- */var/filename* or */var/path/filename*—File on the local hard disk. You can also specify a file on a local Routing Engine for a specific T640 router on a routing matrix:

```
user@host> file delete lcc0-re0:/var/tmp/junk
```

- *a:filename* or *a:path/filename*—File on the local drive. The default path is / (the root-level directory). The removable media can be in MS-DOS or UNIX (UFS) format.
- *hostname:/path/filename*, *hostname:filename*, *hostname:path/filename*, or *scp://hostname/path/filename*—File on an scp/ssh client. This form is not available in the worldwide version of JUNOS Software. The default path is the user's home directory on the remote system. You can also specify *hostname* as *username@hostname*.
- *ftp://hostname/path/filename*—File on an FTP server. You can also specify *hostname* as *username@hostname* or *username:password@hostname*. The default path is the user's home directory. To specify an absolute path, the path must start with %2F; for example, *ftp://hostname/%2Fpath/filename*. To have the system prompt you for the password, specify *prompt* in place of the password. If a password is required, and you do not specify the password or *prompt*, an error message is displayed:

```
user@host> file copy ftp://username@ftp.hostname.net//filename
```

```
file copy ftp.hostname.net: Not logged in.
```

```
user@host> file copy ftp://username:prompt@ftp.hostname.net//filename
```

```
Password for username@ftp.hostname.net:
```

- `http://hostname/path/filename`—File on an HTTP server. You can also specify *hostname* as `username@hostname` or `username:password@hostname`. If a password is required and you omit it, you are prompted for it.
- `re0:/path/filename` or `re1:/path/filename`—File on a local Routing Engine. You can also specify a file on a local Routing Engine for a specific T640 router on a routing matrix:

```
user@host> show log lcc0-re1:chassisd
```

Displaying JUNOS Software Information

You can display JUNOS Software version information and other status to determine if the version of JUNOS Software that you are running supports particular features or hardware.

To display JUNOS Software information:

1. Make sure you are in operational mode.
2. To display brief information and status for the kernel and Packet Forwarding Engine, enter the `show version brief` command. This command shows version information for JUNOS Software packages installed on the router. For example:

```
user@host> show version brief
Hostname: host
Model: m7i
JUNOS Base OS boot [9.1R1.8]
JUNOS Base OS Software Suite [9.1R1.8]
JUNOS Kernel Software Suite [9.1R1.8]
JUNOS Crypto Software Suite [9.1R1.8]
JUNOS Packet Forwarding Engine Support (M/T Common) [9.1R1.8]
JUNOS Packet Forwarding Engine Support (M7i/M10i) [9.1R1.8]
JUNOS Online Documentation [9.1R1.8]
JUNOS Routing Software Suite [9.1R1.8]

user@host>
```

If the **JUNOS Crypto Software Suite** is listed, the router has Canada and USA encrypted JUNOS Software. If the **JUNOS Crypto Software Suite** is not listed, the router is running worldwide nonencrypted JUNOS Software.

3. To display detailed version information, enter the `show version detail` command. This command display shows the hostname and version information for JUNOS Software packages installed on your router. It also includes the version information for each software process. For example:


```
user@host> show version detail
```

```

Hostname: host
Model: m20
JUNOS Base OS boot [8.4R1.13]
JUNOS Base OS Software Suite [8.4R1.13]
JUNOS Kernel Software Suite [8.4R1.13]
JUNOS Crypto Software Suite [8.4R1.13]
JUNOS Packet Forwarding Engine Support (M/T Common) [8.4R1.13]
JUNOS Packet Forwarding Engine Support (M20/M40) [8.4R1.13]
JUNOS Online Documentation [8.4R1.13]
JUNOS Routing Software Suite [8.4R1.13]
KERNEL 8.4R1.13 #0 built by builder on 2007-08-08 00:33:41 UTC
MGD release 8.4R1.13 built by builder on 2007-08-08 00:34:00 UTC
CLI release 8.4R1.13 built by builder on 2007-08-08 00:34:47 UTC
RPD release 8.4R1.13 built by builder on 2007-08-08 00:45:21 UTC
CHASSISD release 8.4R1.13 built by builder on 2007-08-08 00:36:59 UTC
DFWD release 8.4R1.13 built by builder on 2007-08-08 00:34:30 UTC
DCD release 8.4R1.13 built by builder on 2007-08-08 00:34:24 UTC
SNMPD release 8.4R1.13 built by builder on 2007-08-08 00:42:24 UTC
MIB2D release 8.4R1.13 built by builder on 2007-08-08 00:46:47 UTC
APSD release 8.4R1.13 built by builder on 2007-08-08 00:36:39 UTC
VRRPD release 8.4R1.13 built by builder on 2007-08-08 00:45:44 UTC
ALARMD release 8.4R1.13 built by builder on 2007-08-08 00:34:30 UTC
PFED release 8.4R1.13 built by builder on 2007-08-08 00:41:54 UTC
CRAFTD release 8.4R1.13 built by builder on 2007-08-08 00:39:03 UTC
SAMPLED release 8.4R1.13 built by builder on 2007-08-08 00:36:05 UTC
ILMID release 8.4R1.13 built by builder on 2007-08-08 00:36:51 UTC
RMOPD release 8.4R1.13 built by builder on 2007-08-08 00:42:04 UTC
COSD release 8.4R1.13 built by builder on 2007-08-08 00:38:39 UTC
FSAD release 8.4R1.13 built by builder on 2007-08-08 00:43:01 UTC
IRSD release 8.4R1.13 built by builder on 2007-08-08 00:35:37 UTC
FUD release 8.4R1.13 built by builder on 2007-08-08 00:44:36 UTC
RTSPD release 8.4R1.13 built by builder on 2007-08-08 00:29:14 UTC
SMARTD release 8.4R1.13 built by builder on 2007-08-08 00:13:32 UTC
KSYNCD release 8.4R1.13 built by builder on 2007-08-08 00:33:17 UTC
SPD release 8.4R1.13 built by builder on 2007-08-08 00:43:50 UTC
L2TPD release 8.4R1.13 built by builder on 2007-08-08 00:43:12 UTC
HTTPD release 8.4R1.13 built by builder on 2007-08-08 00:36:27 UTC
PPPOED release 8.4R1.13 built by builder on 2007-08-08 00:36:04 UTC
RDD release 8.4R1.13 built by builder on 2007-08-08 00:33:49 UTC
PPPD release 8.4R1.13 built by builder on 2007-08-08 00:45:13 UTC
DFCD release 8.4R1.13 built by builder on 2007-08-08 00:39:11 UTC
DLSWD release 8.4R1.13 built by builder on 2007-08-08 00:42:37 UTC
LACPD release 8.4R1.13 built by builder on 2007-08-08 00:35:41 UTC
USBD release 8.4R1.13 built by builder on 2007-08-08 00:30:01 UTC
LFMD release 8.4R1.13 built by builder on 2007-08-08 00:35:52 UTC
CFMD release 8.4R1.13 built by builder on 2007-08-08 00:34:45 UTC
JDHCPD release 8.4R1.13 built by builder on 2007-08-08 00:35:40 UTC
PGCPD release 8.4R1.13 built by builder on 2007-08-08 00:46:31 UTC
SSD release 8.4R1.13 built by builder on 2007-08-08 00:36:17 UTC
MSPD release 8.4R1.13 built by builder on 2007-08-08 00:33:42 UTC
KMD release 8.4R1.13 built by builder on 2007-08-08 00:44:02 UTC
PPMD release 8.4R1.13 built by builder on 2007-08-08 00:36:03 UTC
LMPD release 8.4R1.13 built by builder on 2007-08-08 00:33:49 UTC
LRMUXD release 8.4R1.13 built by builder on 2007-08-08 00:33:55 UTC
PGMD release 8.4R1.13 built by builder on 2007-08-08 00:36:01 UTC
BFDD release 8.4R1.13 built by builder on 2007-08-08 00:44:22 UTC
SDXD release 8.4R1.13 built by builder on 2007-08-08 00:36:18 UTC

```

```

AUDITD release 8.4R1.13 built by builder on 2007-08-08 00:34:40 UTC
L2ALD release 8.4R1.13 built by builder on 2007-08-08 00:40:05 UTC
EVENTD release 8.4R1.13 built by builder on 2007-08-08 00:39:55 UTC
L2CPD release 8.4R1.13 built by builder on 2007-08-08 00:41:04 UTC
MPLSOAMD release 8.4R1.13 built by builder on 2007-08-08 00:45:11 UTC
jroute-dd release 8.4R1.13 built by builder on 2007-08-08 00:31:01 UTC
jkernel-dd release 8.4R1.13 built by builder on 2007-08-08 00:30:30 UTC
jcrypto-dd release 8.4R1.13 built by builder on 2007-08-08 00:30:12 UTC
jdocs-dd release 8.4R1.13 built by builder on 2007-08-08 00:02:52 UTC

```

```
user@host>
```

Managing Programs and Processes

This topic shows some examples of JUNOS operational commands that you can use to manage programs and processes on a device running JUNOS Software.

Sections include:

- Showing Software Processes on page 54
- Restarting a JUNOS Software Process on page 56
- Stopping the JUNOS Software on page 57
- Rebooting the JUNOS Software on page 58

Showing Software Processes

To verify system operation or to begin diagnosing an error condition, you may need to display information about software processes running on the router.

To show software processes:

1. Make sure you are in operational mode.
2. Type the **show system processes extensive** command. This command shows the CPU utilization on the router and lists the processes in order of CPU utilization. For example:

```
user@host> show system processes extensive
```

```

1ast pid: 28689; load averages: 0.01, 0.00, 0.00 up 56+06:16:13
04:52:04
73 processes: 1 running, 72 sleeping

```

```

Mem: 101M Active, 101M Inact, 98M Wired, 159M Cache, 69M Buf, 286M Free
Swap: 1536M Total, 1536M Free

```

PID	USERNAME	PRI	NICE	SIZE	RES	STATE	TIME	WCPU	CPU	COMMAND
3365	root	2	0	21408K	4464K	select	511:23	0.00%	0.00%	
	chassisd									
3508	root	2	0	3352K	1168K	select	32:45	0.00%	0.00%	12ald

```

3525 root          2   0  3904K  1620K select  13:40  0.00%  0.00% dcd
5532 root          2   0 11660K  2856K kqread  10:36  0.00%  0.00% rpd
3366 root          2   0  2080K   828K select   8:33  0.00%  0.00% alarmd

3529 root          2   0  2040K   428K select   7:32  0.00%  0.00% irsd
3375 root          2   0  2900K  1600K select   6:01  0.00%  0.00% ppm
3506 root          2   0  5176K  2568K select   5:38  0.00%  0.00% mib2d

4957 root          2   0  1284K   624K select   5:16  0.00%  0.00% ntpd
 6 root         18   0    0K    0K syncer   4:49  0.00%  0.00% syncer

3521 root          2   0  2312K   928K select   2:14  0.00%  0.00% lfm
3526 root          2   0  5192K  1988K select   2:04  0.00%  0.00% snmpd

3543 root          2   0    0K    0K peer_s   1:46  0.00%  0.00% peer
proxy
3512 root          2   0  3472K  1044K select   1:44  0.00%  0.00% rmopd

3537 root          2   0    0K    0K peer_s   1:30  0.00%  0.00% peer
proxy
3527 root          2   0  3100K  1176K select   1:14  0.00%  0.00% pfd
3380 root          2   0  3208K  1052K select   1:11  0.00%  0.00% bfdd
4136 root          2   0 11252K  3668K select   0:54  0.00%  0.00% cli
3280 root          2   0  2248K  1420K select   0:28  0.00%  0.00% eventd

3528 root          2   0  2708K   672K select   0:28  0.00%  0.00% dfwd
 7 root         -2   0    0K    0K vlruwt   0:26  0.00%  0.00% vntru

3371 root          2   0  1024K   216K sbwait   0:25  0.00%  0.00%
tnp.snmpd
 13 root         -18   0    0K    0K psleep   0:24  0.00%  0.00%
vmuncacheda
3376 root          2   0  1228K   672K select   0:22  0.00%  0.00% smartd

 5 root         -18   0    0K    0K psleep   0:17  0.00%  0.00%
bufdaemon
3368 root          2   0 15648K  9428K select   0:17  0.00%  0.00% mgd
3362 root          2   0  1020K   204K select   0:15  0.00%  0.00%
watchdog
3381 root          2   0  2124K   808K select   0:15  0.00%  0.00% lacpd

3524 root          2   0  6276K  1492K select   0:14  0.00%  0.00% kmd
3343 root         10   0  1156K   404K nanslp   0:14  0.00%  0.00% cron
---(more)---

```

Table 6 on page 56 lists and describes the output fields included in this example. The fields are listed in alphabetical order.

Table 6: show system process extensive Command Output Fields

Field	Description
COMMAND	Command that is running.
CPU	Raw (unweighted) CPU usage. The value of this field is used to sort the processes in the output.
last pid	Last process identifier assigned to the process.
load averages	Three load averages, followed by the current time.
Mem	Information about physical and virtual memory allocation.
NICE	UNIX “nice” value. The nice value allows a process to change its final scheduling priority.
PID	Process identifier.
PRI	Current kernel scheduling priority of the process. A lower number indicates a higher priority.
processes	Number of existing processes and the number of processes in each state (sleeping, running, starting, zombies, and stopped).
RES	Current amount of resident memory, in KB.
SIZE	Total size of the process (text, data, and stack), in KB.
STATE	Current state of the process (sleep, wait, run, idle, zombi, or stop).
Swap	Information about physical and virtual memory allocation.
USERNAME	Owner of the process.
WCPU	Weighted CPU usage.

Restarting a JUNOS Software Process

To correct an error condition, you might need to restart a software process running on the router. You can use the **restart** command to force a restart of a software process.



CAUTION: Do not restart a software process unless specifically asked to do so by your Juniper Networks customer support representative. Restarting a software process during normal operation of a router could cause interruption of packet forwarding and loss of data.

To restart a software process:

1. Make sure you are in operational mode.

2. Type the following command:

```
user@host> restart process-name < (immediately | gracefully | soft) >
```

- *process-name* is the name of the process that you want to restart. For example, *routing* or *class-of-service*. You can use the command completion feature of JUNOS Software to see a list of software processes that you can restart using this command. For more information on using the command completion feature, see “Using CLI Command Completion” on page 29.
- *gracefully* restarts the software process after performing clean-up tasks.
- *immediately* restarts the software process without performing any clean-up tasks.
- *soft* rereads and reactivates the configuration without completely restarting the software processes. For example, BGP peers stay up and the routing table stays constant.

The following example shows how to restart the routing process:

```
user@host> restart routing
Routing protocol daemon started, pid 751
```

When a process restarts, the process identifier (PID) is updated. (See Figure 7 on page 57.)

Figure 7: Restarting a Process

	PID	USERNAME	PRI	NICE	SIZE	RES	STATE	TIME	WCPU	CPU	COMMAND
PID before restart	546	root	10	0	9096K	1720K	nanslp	0:21	0.00%	0.00%	chassisd
	685	root	2	0	12716K	3840K	kqread	0:01	0.00%	0.00%	rp
	553	root	2	0	8792K	1544K	select	0:01	0.00%	0.00%	mib2d
PID after restart	547	root	2	0	7732K	888K	select	0:00	0.00%	0.00%	alarmd
	545	root	2	0	10292K	2268K	select	0:00	0.00%	0.00%	dcd
	1	root	10	0	816K	520K	wait	0:00	0.00%	0.00%	init
	550	root	2	-12	1308K	692K	select	0:00	0.00%	0.00%	ntpd
	758	root	32	0	21716K	832K	RUN	0:00	0.00%	0.00%	top
	560	root	2	0	8208K	1088K	select	0:00	0.00%	0.00%	rmopd
	561	root	2	0	8188K	1156K	select	0:00	0.00%	0.00%	cosd
	559	root	2	0	1632K	840K	select	0:00	0.00%	0.00%	ilmid
	573	lab	2	0	7480K	2580K	select	0:00	0.00%	0.00%	cli
	751	root	2	0	12716K	3944K	kqread	0:00	0.00%	0.00%	rp
	558	root	2	20	8708K	1880K	select	0:00	0.00%	0.00%	sampld
	555	root	2	0	1856K	932K	select	0:00	0.00%	0.00%	vrrpd
	686	root	2	0	7808K	940K	select	0:00	0.00%	0.00%	apsd

Stopping the JUNOS Software

To avoid damage to the file system, you must gracefully shut down JUNOS Software before powering off the router.

To stop JUNOS Software:

1. Make sure you are in operational mode.

2. Enter the **request system halt** command. This command stops all system processes and halts the operating system. For example:

```
user@host> request system halt

Halt the system? [yes,no] (no) yes
shutdown: [pid 3110]
Shutdown NOW!
*** FINAL System shutdown message from root@host ***
System going down IMMEDIATELY
user@host> Dec 17 17:28:40 init: syslogd (PID 2514) exited with status=0
Normal Exit
Waiting (max 60 seconds) for system process `bufdaemon' to stop...stopped
Waiting (max 60 seconds) for system process `syncer' to stop...stopped
syncing disks... 4
done
Uptime: 3h31m41s
ata0: resetting devices.. done
The operating system has halted.
Please press any key to reboot.
```

Rebooting the JUNOS Software

After a software upgrade or to recover (occasionally) from an error condition, you will need to reboot JUNOS Software.

To reboot JUNOS Software:

1. Make sure you are in operational mode.
2. Enter the **request system reboot** command. This command displays the final stages of the system shutdown and executes the reboot. Reboot requests are recorded to the system log files, which you can view with the **show log messages** command. For example:

```
user@host>request system rebootReboot the system? [yes,no] (no)yes

shutdown: [pid 845]
Shutdown NOW!
*** FINAL System shutdown message from root@host ***
System going down IMMEDIATELY
user@host> Dec 17 17:34:20 init: syslogd (PID 409) exited with status=0
Normal Exit
Waiting (max 60 seconds) for system process `bufdaemon' to stop...stopped
Waiting (max 60 seconds) for system process `syncer' to stop...stopped
syncing disks... 10 6
done
Uptime: 2m45s
ata0: resetting devices.. done
Rebooting...
```

Using the JUNOS Comment Character

The comment character in JUNOS Software enables you to copy operational mode commands that include comments from a file and paste them into the CLI. A pound sign (#) at the beginning of the command-line indicates a comment line. This is useful for describing frequently used operational mode commands; for example, a user's work instructions on how to monitor the network. To add a comment to a command file, the first character of the line must be #. When you start a command with #, the rest of the line is disregarded by JUNOS Software.

To add comments in operational mode, start with a # and end with a new line (carriage return):

```
user@host> # comment-string
```

comment-string is the text of the comment. The comment text can be any length, but each comment line must begin with a #.

Related Topics ■ Example: Using Comments in JUNOS Commands on page 59

Example: Using Comments in JUNOS Commands

File with Comments	<pre>#Command 1: Show the router version show version #Command 2: Show all router interfaces show interfaces terse</pre>
Copy and Paste Contents of the File into the CLI	<pre>user@host> #Command 1: Show the router version user@host> show version Hostname: myhost Model: m5 JUNOS Base OS boot [6.4-20040511.0] JUNOS Base OS Software Suite [6.4-20040511.0] JUNOS Kernel Software Suite [6.4-20040511.0] JUNOS Packet Forwarding Engine Support (M5/M10) [6.4-20040511.0] JUNOS Routing Software Suite [6.4-20040511.0] JUNOS Online Documentation [6.4-20040511.0] JUNOS Crypto Software Suite [6.4-20040511.0] user@host> # Command 2: Show all router interfaces user@host> show interfaces terse Interface Admin Link Proto Local Remote fe-0/0/0 up up fe-0/0/1 up down fe-0/0/2 up down mo-0/1/0 up mo-0/1/0.16383 up up inet 10.0.0.1 -> 10.0.0.17 so-0/2/0 up up so-0/2/1 up up dsc up up fxp0 up up fxp0.0 up up inet 192.168.70.62/21 fxp1 up up fxp1.0 up up tnp 4</pre>

```
gre up up
ipip up up
lo0 up up
lo0.0 up up inet 127.0.0.1 -> 0/0
lo0.16385 up up inet
```


Chapter 5

Using Commands and Statements to Configure a Device Running JUNOS Software

This chapter describes how to use the CLI to configure the router.

Topics include:

- Understanding JUNOS CLI Configuration Mode on page 62
- Entering and Exiting Configuration Mode on page 68
- Modifying the JUNOS Configuration on page 70
- Displaying the Current Configuration on page 70
- Example: Displaying the Current Configuration on page 71
- Adding JUNOS Configuration Statements and Identifiers on page 72
- Deleting a Statement from a JUNOS Configuration on page 73
- Example: Deleting a Statement from the JUNOS Configuration on page 74
- Copying a JUNOS Statement in the Configuration on page 75
- Example: Copying a Statement in the JUNOS Configuration on page 75
- Issuing Relative Configuration Commands on page 76
- Renaming an Identifier on page 76
- Example: Renaming an Identifier on page 76
- Inserting a New Identifier on page 77
- Example: Inserting a New Identifier on page 77
- Deactivating and Reactivating Statements and Identifiers on page 79
- Examples: Deactivating and Reactivating Statements and Identifiers on page 79
- Adding Comments in a JUNOS Configuration on page 80
- Example: Including Comments in a JUNOS Configurations on page 81
- Verifying a JUNOS Configuration on page 82
- Committing a JUNOS Configuration on page 83
- Committing a JUNOS Configuration and Exiting Configuration Mode on page 84
- Activating a JUNOS Configuration but Requiring Confirmation on page 84

- Scheduling a JUNOS Commit Operation on page 85
- Monitoring the JUNOS Commit Process on page 86
- Adding a Comment to Describe the Committed Configuration on page 87
- Updating the Alternate Boot Drive on page 88
- When Multiple Users Configure the Software on page 88
- Forms of the configure Command on page 89
- Example: Using the configure Command on page 90
- Displaying Users Currently Editing the Configuration on page 90
- Using the configure exclusive Command on page 91
- Updating the configure private Configuration on page 92
- Displaying set Commands from the Configuration on page 93
- Displaying Additional Information About the Configuration on page 95

Understanding JUNOS CLI Configuration Mode

You can configure all properties of JUNOS Software, including interfaces, general routing information, routing protocols, and user access, as well as several system hardware properties.

As described in “Understanding the JUNOS CLI Modes and Command and Statement Hierarchies” on page 5, a router configuration is stored as a hierarchy of statements. In configuration mode, you create the specific hierarchy of configuration statements that you want to use. When you have finished entering the configuration statements, you commit them, which activates the configuration on the router.

You can create the hierarchy interactively or you can create an ASCII text file that is loaded onto the router and then committed.

This topic in this section include:

- Configuration Mode Commands on page 62
- Configuration Statements and Identifiers on page 64
- Configuration Statement Hierarchy on page 66

Configuration Mode Commands

Table 7 on page 62 summarizes each CLI configuration mode command. The commands are organized alphabetically.

Table 7: Summary of Configuration Mode Commands

Command	Description
activate	Remove the <code>inactive:</code> tag from a statement, effectively reading the statement or identifier to the configuration. Statements or identifiers that have been activated take effect when you next issue the <code>commit</code> command.

Table 7: Summary of Configuration Mode Commands *(continued)*

Command	Description
<code>annotate</code>	Add comments to a configuration. You can add comments only at the current hierarchy level.
<code>commit</code>	Commit the set of changes to the database and cause the changes to take operational effect.
<code>copy</code>	Make a copy of an existing statement in the configuration.
<code>deactivate</code>	Add the inactive: tag to a statement, effectively commenting out the statement or identifier from the configuration. Statements or identifiers marked as inactive do not take effect when you issue the <code>commit</code> command.
<code>delete</code>	Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it.
<code>edit</code>	Move inside the specified statement hierarchy. If the statement does not exist, it is created.
<code>exit</code>	Exit the current level of the statement hierarchy, returning to the level prior to the last <code>edit</code> command, or exit from configuration mode. The <code>quit</code> and <code>exit</code> commands are synonyms.
<code>extension</code>	Manage configurations that are contributed by SDK application packages by either displaying or deleting user-defined configuration contributed by the named SDK application package. A configuration defined in any native JUNOS package is never deleted by the <code>extension</code> command.
<code>help</code>	Display help about available configuration statements.
<code>insert</code>	Insert an identifier into an existing hierarchy.
<code>load</code>	Load a configuration from an ASCII configuration file or from terminal input. Your current location in the configuration hierarchy is ignored when the load operation occurs.
<code>quit</code>	Exit the current level of the statement hierarchy, returning to the level prior to the last <code>edit</code> command, or exit from configuration mode. The <code>quit</code> and <code>exit</code> commands are synonyms.
<code>rename</code>	Rename an existing configuration statement or identifier.
<code>replace</code>	Replace identifiers or values in a configuration.
<code>rollback</code>	Return to a previously committed configuration. The software saves the last 10 committed configurations, including the rollback number, date, time, and name of the user who issued the <code>commit configuration</code> command.
<code>run</code>	Run a top-level CLI command without exiting from configuration mode.

Table 7: Summary of Configuration Mode Commands *(continued)*

Command	Description
save	Save the configuration to an ASCII file. The contents of the current level of the statement hierarchy (and below) are saved, along with the statement hierarchy containing it. This allows a section of the configuration to be saved, while fully specifying the statement hierarchy.
set	Create a statement hierarchy and set identifier values. This is similar to edit except that your current level in the hierarchy does not change.
show	Display the current configuration.
status	Display the users currently editing the configuration.
top	Return to the top level of configuration command mode, which is indicated by the [edit] banner.
up	Move up one level in the statement hierarchy.
update	Update a private database.
wildcard	Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it. You can use regular expressions to specify the pattern based on which you want to delete multiple items.

For more information about configuration mode commands, see “Summary of CLI Configuration Mode Commands” on page 209.

Configuration Statements and Identifiers

You can configure router properties by including the corresponding statements in the configuration. Typically, a statement consists of a keyword, which is fixed text, and, optionally, an identifier. An identifier is an identifying name that you can define, such as the name of an interface, or a username, which allows you and the CLI to differentiate among a collection of statements.

Table 8 on page 64 describes top-level CLI configuration mode statements.

Table 8: Configuration Mode Top-Level Statements

Statement	Description
access	Configure the Challenge Handshake Authentication Protocol (CHAP). For information about the statements in this hierarchy, see the <i>JUNOS System Basics Configuration Guide</i> .
accounting-options	Configure accounting statistics data collection for interfaces and firewall filters. For information about the statements in this hierarchy, see the <i>JUNOS Network Management Configuration Guide</i> .

Table 8: Configuration Mode Top-Level Statements *(continued)*

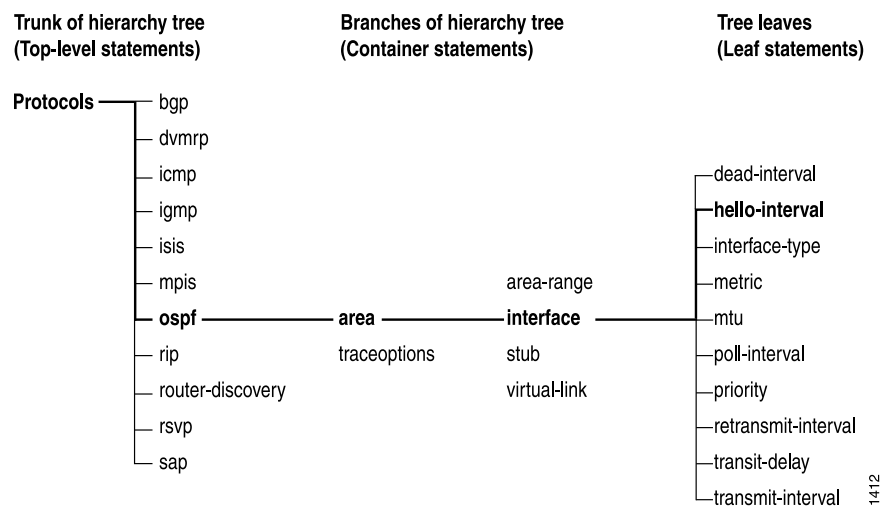
Statement	Description
chassis	Configure properties of the router chassis, including conditions that activate alarms and SONET/SDH framing and concatenation properties. For information about the statements in this hierarchy, see the <i>JUNOS System Basics Configuration Guide</i> .
class-of-service	Configure class-of-service parameters. For information about the statements in this hierarchy, see the <i>JUNOS Class of Service Configuration Guide</i> .
firewall	Define filters that select packets based on their contents. For information about the statements in this hierarchy, see the <i>JUNOS Policy Framework Configuration Guide</i> .
forwarding-options	Define forwarding options, including traffic sampling options. For information about the statements in this hierarchy, see the <i>JUNOS Network Interfaces Configuration Guide</i> .
groups	Configure configuration groups. For information about statements in this hierarchy, see the <i>JUNOS System Basics Configuration Guide</i> .
interfaces	Configure interface information, such as encapsulation, interfaces, virtual channel identifiers (VCIs), and data-link connection identifiers (DLCIs). For information about the statements in this hierarchy, see the <i>JUNOS Network Interfaces Configuration Guide</i> .
policy-options	Define routing policies, which allow you to filter and set properties in incoming and outgoing routes. For information about the statements in this hierarchy, see the <i>JUNOS Policy Framework Configuration Guide</i> .
protocols	Configure routing protocols, including BGP, IS-IS, LDP, MPLS, OSPF, RIP, and RSVP. For information about the statements in this hierarchy, see the chapters that discuss how to configure the individual routing protocols in the <i>JUNOS Routing Protocols Configuration Guide</i> and the <i>JUNOS MPLS Applications Configuration Guide</i> .
routing-instances	Configure multiple routing instances. For information about the statements in this hierarchy, see the <i>JUNOS Routing Protocols Configuration Guide</i> .
routing-options	Configure protocol-independent routing options, such as static routes, autonomous system numbers, confederation members, and global tracing (debugging) operations to log. For information about the statements in this hierarchy, see the <i>JUNOS Routing Protocols Configuration Guide</i> .
security	Configure IP Security (IPsec) services. For information about the statements in this hierarchy see the <i>JUNOS System Basics Configuration Guide</i> .
snmp	Configure SNMP community strings, interfaces, traps, and notifications. For information about the statements in this hierarchy, see the <i>JUNOS Network Management Configuration Guide</i> .
system	Configure systemwide properties, including the hostname, domain name, Domain Name System (DNS) server, user logins and permissions, mappings between hostnames and addresses, and software processes. For information about the statements in this hierarchy, see the <i>JUNOS System Basics Configuration Guide</i> .

For specific information on configuration statements, see the JUNOS configuration guides.

Configuration Statement Hierarchy

The JUNOS Software configuration consists of a hierarchy of *statements*. There are two types of statements: *container statements*, which are statements that contain other statements, and *leaf statements*, which do not contain other statements (see Figure 8 on page 66). All of the container and leaf statements together form the *configuration hierarchy*.

Figure 8: Configuration Mode Hierarchy of Statements



Each statement at the top level of the configuration hierarchy resides at the trunk (or root level) of a hierarchy tree. The top-level statements are container statements, containing other statements that form the tree branches. The leaf statements are the leaves of the hierarchy tree. An individual hierarchy of statements, which starts at the trunk of the hierarchy tree, is called a *statement path*. Figure 8 on page 66 illustrates the hierarchy tree, showing a statement path for the portion of the protocol configuration hierarchy that configures the hello interval on an interface in an OSPF area.

The **protocols** statement is a top-level statement at the trunk of the configuration tree. The **ospf**, **area**, and **interface** statements are all subordinate container statements of a higher statement (they are branches of the hierarchy tree); and the **hello-interval** statement is a leaf on the tree, which, in this case, contains a data value: the length of the hello interval, in seconds.

The CLI represents the statement path shown in Figure 8 on page 66 as `[protocols ospf area area-number interface interface-name]` and displays the configuration as follows:

```
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
```

```
        hello-interval 5;
    }
    interface so-0/0/1 {
        hello-interval 5;
    }
}
}
```

The CLI indents each level in the hierarchy to indicate each statement's relative position in the hierarchy and generally sets off each level with braces, using an open brace at the beginning of each hierarchy level and a closing brace at the end. If the statement at a hierarchy level is empty, the braces are not printed.

Each leaf statement ends with a semicolon. If the hierarchy does not extend as far as a leaf statement, the last statement in the hierarchy ends with a semicolon.

Entering and Exiting Configuration Mode

You configure JUNOS Software by entering configuration mode and creating a hierarchy of configuration mode statements.

- To enter configuration mode, use the **configure** command.

When you enter configuration mode, the following configuration mode commands are available:

```

user@host>configure

entering configuration mode

[edit]

user@host#?

possible completions:
  <[Enter]>      Execute this command
  activate       Remove the inactive tag from a statement
  annotate       Annotate the statement with a comment
  commit         Commit current set of changes
  copy           Copy a statement
  deactivate     Add the inactive tag to a statement
  delete         Delete a data element
  edit           Edit a sub-element
  exit           Exit from this level
  help           Provide help information
  insert         Insert a new ordered data element
  load           Load configuration from ASCII file
  quit           Quit from this level
  rename         Rename a statement
  replace        Replace character string in configuration
  rollback       Roll back to previous committed configuration
  run            Run an operational-mode command
  save           Save configuration to ASCII file
  set            Set a parameter
  show           Show a parameter
  status         Show users currently editing configuration
  top            Exit to top level of configuration
  up             Exit one level of configuration
  wildcard       Wildcard operations
[edit]
user@host>

```

Users must have configure permission to view and use the **configure** command. When in configuration mode, a user can view and modify only those statements for which they have access privileges set. For more information, see the *JUNOS System Basics Configuration Guide*.

- If you enter configuration mode and another user is also in configuration mode, a message shows the user's name and what part of the configuration the user is viewing or editing:


```

user@host> configure
Entering configuration mode
Users currently editing the configuration:
  root terminal d0 (pid 4137) on since 2008-04-09 23:03:07 PDT, idle 7w6d
08:22
[edit]
The configuration has been changed but not committed

[edit]
user@host#

```

Up to 32 users can be in configuration mode simultaneously, and they all can make changes to the configuration at the same time. For more information, see “When Multiple Users Configure the Software” on page 88.

- To exit configuration mode, use the **exit configuration-mode** configuration mode command from any level, or use the **exit** command from the top level. For example:

```

[edit protocols ospf area 0.0.0.0 interface so-0/0/0]
user@host# exit configuration-mode
exiting configuration mode
user@host>

[edit]
user@host# exit
exiting configuration mode
user@host>

```

If you try to exit from configuration mode using the **exit** command and the configuration contains changes that have not been committed, you see a message and prompt:

```

[edit]
user@host# exit
The configuration has been changed but not committed
Exit with uncommitted changes? [yes,no] (yes) <Enter>
Exiting configuration mode
user@host>

```

- To exit with uncommitted changes without having to respond to a prompt, use the **exit configuration-mode** command. This command is useful when you are using scripts to perform remote configuration.

```

[edit]
user@host# exit configuration-mode
The configuration has been changed but not committed
Exiting configuration mode
user@host>

```

Modifying the JUNOS Configuration

To configure a device running JUNOS Software or to modify an existing JUNOS configuration, you add statements to the configuration. For each statement hierarchy, you create the hierarchy starting with a statement at the top level and continuing with statements that move progressively lower in the hierarchy.

To modify the hierarchy, you use two configuration mode commands:

- **edit**—Moves to a particular hierarchy level. If that hierarchy level does not exist, the **edit** command creates it. The **edit** command has the following syntax:

```
edit <statement-path>
```

- **set**—Creates a configuration statement and sets identifier values. After you issue a **set** command, you remain at the same level in the hierarchy. The **set** command has the following syntax:

```
set <statement-path> statement <identifier>
```

statement-path is the hierarchy to the configuration statement and the statement itself. If you have already moved to the statement's hierarchy level, you can omit the statement path. *statement* is the configuration statement itself. *identifier* is a string that identifies an instance of a statement.

You cannot use the **edit** command to change the value of identifiers. You must use the **set** command.

Displaying the Current Configuration

To display the current configuration for a device running JUNOS Software, use the **show** configuration mode command. This command displays the configuration at the current hierarchy level or at the specified level.

```
user@host# show <statement-path>
```

The configuration statements appear in a fixed order, and interfaces appear alphabetically by type, and then in numerical order by slot number, PIC number, and port number. Note that when you configure the router, you can enter statements in any order.

You also can use the CLI operational mode **show configuration** command to display the last committed current configuration, which is the configuration currently running on the router:

```
user@host> show configuration
```

When you show a configuration, a timestamp at the top of the configuration indicates when the configuration was last changed:

```
## Last commit: 2006-07-18 11:21:58 PDT by echen
version 8.3
```

If you have omitted a required statement at a particular hierarchy level, when you issue the **show** command in configuration mode, a message indicates which statement is missing. As long as a mandatory statement is missing, the CLI continues to display this message each time you issue a **show** command. For example:

```
[edit]
user@host# show
protocols {
  pim {
    interface so-0/0/0 {
      priority 4;
      version 2;
      # Warning: missing mandatory statement(s): 'mode'
    }
  }
}
```

Example: Displaying the Current Configuration

The following example shows how you can display the current JUNOS configuration. To display the entire configuration:

```
[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/0 hello-interval 5
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
```

Display a particular hierarchy in the configuration:

```
[edit]
user@host# show protocols ospf area 0.0.0.0
interface so-0/0/0 {
  hello-interval 5;
}
```

Move down to a level and display the configuration at that level:

```
[edit]
user@host# edit protocols ospf area 0.0.0.0
[edit protocols ospf area 0.0.0.0]
user@host# show
interface so-0/0/0 {
  hello-interval 5;
}
```

Display all of the last committed configuration:

```

[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/0 hello-interval 5
[edit]
user@host# commit
commit complete
[edit]
user@host# quit
exiting configuration mode
user@host> show configuration
## Last commit: 2006-08-10 11:21:58 PDT by user
version 8.3
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}

```

Adding JUNOS Configuration Statements and Identifiers

All properties of a device running JUNOS Software are configured by including *statements* in the configuration. A statement consists of a keyword, which is fixed text, and, optionally, an *identifier*. An identifier is an identifying name which you define, such as the name of an interface or a username, and which allows you and the CLI to discriminate among a collection of statements.

For example, the following list shows the statements available at the top level of configuration mode.

```

user@host# set?
Possible completions:
> accounting-options  Accounting data configuration
+ apply-groups        Groups from which to inherit configuration data
> chassis             Chassis configuration
> class-of-service    Class-of-service configuration
> firewall            Define a firewall configuration
> forwarding-options  Configure options to control packet sampling
> groups              Configuration groups
> interfaces          Interface configuration
> policy-options      Routing policy option configuration
> protocols           Routing protocol configuration
> routing-instances   Routing instance configuration
> routing-options     Protocol-independent routing option configuration
> snmp               Simple Network Management Protocol
> system              System parameters

```

An angle bracket (>) before the statement name indicates that it is a container statement and that you can define other statements at levels below it. If there is no angle bracket (>) before the statement name, the statement is a leaf statement; you cannot define other statements at hierarchy levels below it.

A plus sign (+) before the statement name indicates that it can contain a set of values. To specify a set, include the values in brackets. For example:

```
[edit]
user@host# set policy-options community my-as1-transit members [65535:10
65535:11]
```

In some statements, you can include an identifier. For some identifiers, such as interface names, you must specify the identifier in a precise format. For example, the interface name **so-0/0/0** refers to a SONET/SDH interface that is on the Flexible PIC Concentrator (FPC) in slot 0, in the first PIC location, and in the first port on the Physical Interface Card (PIC). For other identifiers, such as interface descriptive text and policy and firewall term names, you can specify any name, including special characters, spaces, and tabs.

You must enclose in quotation marks (double quotes) identifiers and any strings that include a space or tab character or any of the following characters:

```
()[]{}!@#$%^&|'=?
```

If you do not type an option for a statement that requires one, a message indicates the type of information required. In this example, you need to type an area number to complete the command:

```
[edit]
user@host# set protocols ospf area<Enter>
          ^
syntax error, expecting <identifier>
```

Deleting a Statement from a JUNOS Configuration

To delete a statement or identifier from a JUNOS configuration, use the **delete** configuration mode command. Deleting a statement or an identifier effectively "unconfigures" the functionality associated with that statement or identifier, returning that functionality to its default condition.

```
user@host# delete <statement-path> <identifier>
```

When you delete a statement, the statement and all its subordinate statements and identifiers are removed from the configuration.

For statements that can have more than one identifier, when you delete one identifier, only that identifier is deleted. The other identifiers in the statement remain.

To delete the entire hierarchy starting at the current hierarchy level, do not specify a statement or an identifier in the **delete** command. When you omit the statement or identifier, you are prompted to confirm the deletion:

```
[edit]
user@host# delete
Delete everything under this level? [yes, no] (no)
Possible completions:
no    Don't delete everything under this level
```

```
yes    Delete everything under this level
Delete everything under this level? [yes, no] (no)
```

Example: Deleting a Statement from the JUNOS Configuration

The following example shows how to delete the `ospf` statement, effectively unconfiguring OSPF on the router:

```
[edit]
user@host# set protocols ospf area 0.0.0.0 interface so-0/0/0 hello-interval 5
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
[edit]
user@host# delete protocols ospf
[edit]
user@host# show
[edit]
user@host#
```

Delete all statements from the current level down:

```
[edit]
user@host# edit protocols ospf area 0.0.0.0
[edit protocols ospf area 0.0.0.0]
user@host# set interface so-0/0/0 hello-interval 5
[edit protocols ospf area 0.0.0.0]
user@host# delete
Delete everything under this level? [yes, no] (no) yes
[edit protocols ospf area 0.0.0.0]
user@host# show
[edit]
user@host#
```

Unconfigure a particular property:

```
[edit]
user@host# set interfaces so-3/0/0 speed 100mb
[edit]
user@host# show
interfaces {
  so-3/0/0 {
    speed 100mb;
  }
}
[edit]
```

```

user@host# delete interfaces so-3/0/0 speed
[edit]
user@host# show
interfaces {
  so-3/0/0;
}

```

For information about how to use regular expressions to delete related configuration items, see “Using Global Replace in a JUNOS Configuration” Using the upto Option” on page 141.

Copying a JUNOS Statement in the Configuration

When you have many statements in a JUNOS configuration that are similar, you can add one statement and then make copies of that statement. Copying a statement duplicates that statement and the entire hierarchy of statements configured under that statement. Copying statements is useful when you are configuring many physical or logical interfaces of the same type.

To make a copy of an existing statement in the configuration, use the configuration mode **copy** command:

```

user@host# copy existing-statement to new-statement

```

Immediately after you have copied a portion of the configuration, the configuration might not be valid. You must check the validity of the new configuration, and if necessary, modify either the copied portion or the original portion for the configuration to be valid.

Example: Copying a Statement in the JUNOS Configuration

The following example shows how you can create one virtual connection (VC) on an interface, and then copy its configuration to create a second VC:

```

[edit interfaces]
user@host# show
at-1/0/0 {
  description "PAIX to MAE West"
  encapsulation atm-pvc;
  unit 61 {
    point-to-point;
    vci 0.61;
    family inet {
      address 10.0.1.1/24;
    }
  }
}
[edit interfaces]
user@host# edit at-1/0/0
[edit interfaces at-1/0/0]
user@host# copy unit 61 to unit 62
[edit interfaces at-1/0/0]
user@host# show

```

```

description "PAIX to MAE West"
encapsulation atm-pvc;
unit 61 {
    point-to-point;
    vci 0.61;
    family inet {
        address 10.0.1.1/24;
    }
}
unit 62 {
    point-to-point;
    vci 0.61;
    family inet {
        address 10.0.1.1/24;
    }
}

```

Issuing Relative Configuration Commands

The **top** or **up** command followed by another configuration command, including **edit**, **insert**, **delete**, **deactivate**, **annotate**, or **show** enables you to quickly move to the top of the hierarchy or to a level above the area you are configuring.

To issue configuration mode commands from the top of the hierarchy, use the **top** command; then specify a configuration command. For example:

```

[edit interfaces fxp0 unit 0 family inet]
user@host# top edit system login
[edit system login]
user@host#

```

To issue configuration mode commands from a location higher up in the hierarchy, use the **up** configuration mode command; specify the number of levels you want to move up the hierarchy and then specify a configuration command. For example:

```

[edit protocols bgp]
user@host# up 2 activate system

```

Renaming an Identifier

When modifying a JUNOS configuration, you can rename an identifier that is already in the configuration. You can do this either by deleting the identifier (using the **delete** command) and then adding the renamed identifier (using the **set** and **edit** commands), or you can rename the identifier using the **rename** configuration mode command:

```

user@host# rename <statement-path> identifier1 to identifier2

```

Example: Renaming an Identifier

Change the Network Time Protocol (NTP) server address to 10.0.0.6:

```

[edit]

```



```
user@host# rename system network-time server 10.0.0.7 to server 10.0.0.6
```

Inserting a New Identifier

When configuring a device running JUNOS Software, you can enter most statements and identifiers in any order. Regardless of the order in which you enter the configuration statements, the CLI always displays the configuration in a strict order. However, there are a few cases where the ordering of the statements matters because the configuration statements create a sequence that is analyzed in order.

For example, in a routing policy or firewall filter, you define terms that are analyzed sequentially. Also, when you create a named path in dynamic MPLS, you define an ordered list of the transit routers in the path, starting with the first transit router and ending with the last one.

To modify a portion of the configuration in which the statement order matters, use the `insert` configuration mode command:

```
user@host#insert <statement-path> identifier1 (before | after) identifier2
```

If you do not use the `insert` command, but instead simply configure the identifier, it is placed at the end of the list of similar identifiers.

Example: Inserting a New Identifier

Insert policy terms in a routing policy configuration. Note that if you do not use the `insert` command, but rather just configure another term, the added term is placed at the end of the existing list of terms. Also note that you must create the term, as shown in this example, before you can place it with the `insert` command.

```
[edit]
user@host# show
policy-options {
  policy-statement statics {
    term term1 {
      from {
        route-filter 192.168.0.0/16 orlonger;
        route-filter 224.0.0.0/3 orlonger;
      }
      then reject;
    }
    term term2 {
      from protocol direct;
      then reject;
    }
    term term3 {
      from protocol static;
      then reject;
    }
    term term4 {
      then accept;
    }
  }
}
```

```

}
[edit]
user@host# rename policy-options policy-statement statics term term4 to term
term6
[edit]
user@host# set policy-options policy-statement statics term term4 from protocol
local
[edit]
user@host# set policy-options policy-statement statics term term4 then reject
[edit]
user@host# set policy-options policy-statement statics term term5 from protocol
aggregate
[edit]
user@host# set policy-options policy-statement statics term term5 then reject
[edit]
user@host# insert policy-options policy-statement statics term term4 after term
term3
[edit]
user@host# insert policy-options policy-statement statics term term5 after term
term4
[edit]
user@host# show policy-options policy-statement statics
term term1 {
    from {
        route-filter 192.168.0.0/16 orlonger;
        route-filter 224.0.0.0/3 orlonger;
    }
    then reject;
}
term term2 {
    from protocol direct;
    then reject;
}
term term3 {
    from protocol static;
    then accept;
}
term term4 {
    from protocol local;
    then reject;
}
term term5 {
    from protocol aggregate;
    then reject;
}
term term6 {
    then accept;
}

```

Insert a transit router in a dynamic MPLS path:

```

[edit protocols mpls path ny-sf]
user@host# show
1.1.1.1;
2.2.2.2;
3.3.3.3 loose;

```

```

4.4.4.4 strict;
6.6.6.6;
[edit protocols mpls path ny-sf]
user@host# insert 5.5.5.5 before 6.6.6.6
[edit protocols mpls path ny-sf]
user@host# set 5.5.5.5 strict
[edit protocols mpls path ny-sf]
user@host# show
1.1.1.1;
2.2.2.2;
3.3.3.3 loose;
4.4.4.4 strict;
5.5.5.5 strict;
6.6.6.6;

```

Deactivating and Reactivating Statements and Identifiers

In a JUNOS configuration, you can deactivate statements and identifiers so that they do not take effect when you issue the **commit** command. Any deactivated statements and identifiers are marked with the **inactive:** tag. They remain in the configuration, but are not activated when you issue a **commit** command.

To deactivate a statement or identifier, use the **deactivate** configuration mode command:

```
user@host# deactivate (statement | identifier )
```

To reactivate a statement or identifier, use the **activate** configuration mode command:

```
user@host# activate (statement | identifier )
```

In both commands, the *statement* or *identifier* you specify must be at the current hierarchy level.

In some portions of the configuration hierarchy, you can include a **disable** statement to disable functionality. One example is disabling an interface by including the **disable** statement at the **[edit interface *interface-name*]** hierarchy level. When you deactivate a statement, that specific object or property is completely ignored and is not applied at all when you issue a **commit** command. When you disable a functionality, it is activated when you issue a **commit** command but is treated as though it is down or administratively disabled.

Examples: Deactivating and Reactivating Statements and Identifiers

Deactivate an interface in the configuration:

```

[edit interfaces]
user@host# show
at-5/2/0 {
  traceoptions {
    traceflag all;
  }
  atm-options {

```

```

        vpi 0 maximum-vcs 256;
    }
    unit 0 {
        ...
        [edit interfaces]
        user@host# deactivate at-5/2/0
        [edit interfaces]
        user@host# show
        inactive: at-5/2/0 {
            traceoptions {
                traceflag all;
            }
            ...
        }
    }
}

```

Reactivate the interface:

```

[edit interfaces]
user@host# activate at-5/2/0
[edit interfaces]
user@host# show
at-5/2/0 {
    traceoptions {
        traceflag all;
    }
    ...
}

```

Adding Comments in a JUNOS Configuration

You can include comments in a JUNOS configuration to describe any statement in the configuration. You can add comments interactively in the CLI and by editing the ASCII configuration file.

When you add comments in configuration mode, they are associated with a statement at the current level. Each statement can have one single-line comment associated with it. Before you can associate a comment with a statement, the statement must exist. The comment is placed on the line preceding the statement.

To add comments to a configuration, use the **annotate** configuration mode command:

```

user@host# annotate statement "comment-string"

```

statement is the configuration statement to which you are attaching the comment; it must be at the current hierarchy level. If a comment for the specified *statement* already exists, it is deleted and replaced with the new comment.

comment-string is the text of the comment. The comment text can be any length, and you must type it on a single line. If the comment contains spaces, you must enclose it in quotation marks. In the comment string, you can include the comment delimiters `/* */` or `#`. If you do not specify any, the comment string is enclosed with the `/* */` comment delimiters.

To delete an existing comment, specify an empty comment string:

```
user@host# annotate statement ""
```

When you edit the ASCII configuration file and add comments, they can be one or more lines and must precede the statement they are associated with. If you place the comments in other places in the file, such as on the same line following a statement or on a separate line following a statement, they are removed when you use the **load** command to open the configuration into the CLI.

When you include comments in the configuration file directly, you can format comments in the following ways:

- Start the comment with a **/*** and end it with a ***/**. The comment text can be on a single line or can span multiple lines.
- Start the comment with a **#** and end it with a new line (carriage return).

If you add comments with the **annotate** command, you can view the comments within the configuration by entering the **show** configuration mode command or the **show configuration** operational mode command.

When configuring interfaces, you can add comments about the interface by including the **description** statement at the [edit interfaces *interface-name*] hierarchy level. Any comments you include appear in the output of the **show interfaces** commands. For more information about the **description** statement, see the *JUNOS Network Interfaces Configuration Guide*.

Example: Including Comments in a JUNOS Configurations

To add comments to a JUNOS configuration:

```
[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
[edit]
user@host# edit protocols ospf
[edit protocols ospf]
user@host# set area 0.0.0.0
user@host# annotate area 0.0.0.0 "Backbone area configuration added June 15,
1998"
[edit protocols ospf]
user@host# edit area 0.0.0.0
[edit protocols ospf area 0.0.0.0]
user@host# annotate interface so0 "Interface from router sj1 to router sj2"
[edit protocols ospf area 0.0.0.0]
```

```

user@host# top
[edit]
user@host# show
protocols {
  ospf {
    /* Backbone area configuration added June 15, 1998 */
    area 0.0.0.0 {
      /* Interface from router sj1 to router sj2 */
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
[edit]
user@host#

```

The following excerpt from a configuration example illustrates how to enter comments in a configuration file:

```

/* This comment goes with routing-options */
routing-options {
  /* This comment goes with routing-options traceoptions */
  traceoptions {
    /* This comment goes with routing-options traceoptions tracefile */
    tracefile rpd size 1m files 10;
    /* This comment goes with routing-options traceoptions traceflag task */
    traceflag task;
    /* This comment goes with routing-options traceoptions traceflag general */
    traceflag general;
  }
  autonomous-system 10458; /* This comment is dropped */
}
routing-options {
  rib-groups {
    ifrg {
      import-rib [ inet.0 inet.2 ];
      /* A comment here is dropped */
    }
    dvmrp-rib {
      import-rib inet.2;
      export-rib inet.2;
      /* A comment here is dropped */
    }
    /* A comment here is dropped */
  }
  /* A comment here is dropped */
}

```

Verifying a JUNOS Configuration

To verify that the syntax of a JUNOS configuration is correct, use the configuration mode `commit check` command:

```
[edit]
```

```

user@host# commit check
configuration check succeeds
[edit]
user@host#

```

If the `commit check` command finds an error, a message indicates the location of the error.

Committing a JUNOS Configuration

To save JUNOS Software configuration changes to the configuration database and activate the configuration on the router, use the `commit` configuration mode command:

```

[edit]
user@host# commit
commit complete
[edit]
user@host#

```

When you enter the `commit` command, the configuration is first checked for syntax errors (`commit check`). Then, if the syntax is correct, the configuration is activated and becomes the current, operational router configuration.

You can issue the `commit` command from any hierarchy level.

If the configuration contains syntax errors, a message indicates the location of the error and the configuration is not activated. The error message has the following format:

```

[edit edit-path]
'offending-statement;'
error-message

```

For example:

```

[edit firewall filter login-allowed term allowed from]
'icmp-type [ echo-request echo-reply ];'
keyword 'echo-reply' unrecognized

```

You must correct the error before recommitting the configuration. To return quickly to the hierarchy level where the error is located, copy the path from the first line of the error and paste it at the configuration mode prompt at the `[edit]` hierarchy level.

When you commit a configuration, you commit the entire configuration in its current form. If more than one user is modifying the configuration, committing it saves and activates the changes of all the users.



NOTE: If you are using JUNOS Software in a Common Criteria environment, system log messages are created whenever a **secret** attribute is changed (for example, password changes or changes to the RADIUS shared secret). These changes are logged during the following configuration load operations:

```
load merge
load replace
load override
load update
```

For more information, see the *Secure Configuration Guide for Common Criteria and JUNOS FIPS*.

Committing a JUNOS Configuration and Exiting Configuration Mode

To save JUNOS Software configuration changes, activate the configuration on the device, and exit configuration mode, use the **commit and-quit** configuration mode command. This command succeeds only if the configuration contains no errors.

```
[edit]
user@host# commit and-quit
commit complete
exiting configuration mode
user@host>
```

Activating a JUNOS Configuration but Requiring Confirmation

When you commit the current candidate configuration, you can require an explicit confirmation for the commit to become permanent. This is useful if you want to verify that a configuration change works correctly and does not prevent access to the router. If the change prevents access or causes other errors, the router automatically returns to the previous configuration and restores access after the rollback confirmation timeout passes. This feature is called automatic rollback.

To commit the current candidate configuration but require an explicit confirmation for the commit to become permanent, use the **commit confirmed** configuration mode command:

```
[edit]
user@host# commit confirmed
commit confirmed will be automatically rolled back in 10 minutes unless confirmed
commit complete
#commit confirmed will be rolled back in 10 minutes
[edit]
user@host#
```

Once you have verified that the change works correctly, you can keep the new configuration active by entering a **commit** or **commit check** command within 10 minutes of the **commit confirmed** command. For example:


```
[edit]
user@host# commit check
commit confirmed will be automatically rolled back in 10 minutes unless confirmed
commit complete
#commit confirmed will be rolled back in 10 minutes
[edit]
user@host#
```

If the commit is not confirmed within a certain time (10 minutes by default), JUNOS Software automatically rolls back to the previous configuration and a broadcast message is sent to all logged-in users.

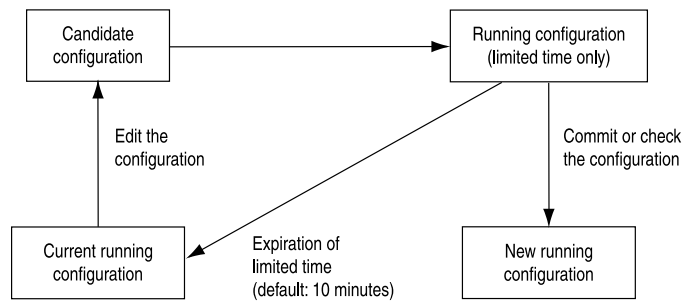
To show when a rollback is scheduled after a **commit confirmed** command, enter the **show system commit** command. For example:

```
user@host>show system commit
0 2005-01-05 15:00:37 PST by root via cli commit confirmed, rollback in 3mins
```

Like the **commit** command, the **commit confirmed** command verifies the configuration syntax and reports any errors. If there are no errors, the configuration is activated and begins running on the router.

Figure 9 on page 85 illustrates how the **commit confirmed** command works.

Figure 9: Confirm a Configuration



To change the amount of time before you have to confirm the new configuration, specify the number of minutes when you issue the command:

```
[edit]
user@host# commit confirmed minutes
commit complete
[edit]
user@host#
```

Scheduling a JUNOS Commit Operation

You can schedule when you want your candidate configuration to become active. To save JUNOS Software configuration changes and activate the configuration on the router at a future time or upon reboot, use the **commit at** configuration mode command, specifying **reboot** or a future time at the **[edit]** hierarchy level:

```
[edit]
user@host # commit at string
```

string is **reboot** or the future time to activate the configuration changes. You can specify time in two formats:

- A time value in the form *hh:mm[:ss]* (hours, minutes, and optionally seconds)—Commit the configuration at the specified time, which must be in the future but before 11:59:59 PM on the day the **commit at** configuration command is issued. Use 24-hour time for the *hh* value; for example, **04:30:00** is 4:30:00 AM, and **20:00** is 8:00 PM. The time is interpreted with respect to the clock and time zone settings on the router.
- A date and time value in the form *yyyy-mm-dd hh:mm[:ss]* (year, month, date, hours, minutes, and, optionally, seconds)—Commit the configuration at the specified day and time, which must be after the **commit at** command is issued. Use 24-hour time for the *hh* value. For example, **2003-08-21 12:30:00** is 12:30 PM on August 21, 2003. The time is interpreted with respect to the clock and time zone settings on the router.

Enclose the *string* value in quotation marks (" "). For example, **commit at "18:00:00"**. For date and time, include both values in the same set of quotation marks. For example, **commit at "2005-03-10 14:00:00"**.

A commit check is performed immediately when you issue the **commit at** configuration mode command. If the result of the check is successful, then the current user is logged out of configuration mode, and the configuration data is left in a read-only state. No other commit can be performed until the scheduled commit is completed.



NOTE: If the JUNOS Software fails before the configuration changes become active, all configuration changes are lost.

You cannot issue the **commit at** configuration command after you issue the **request system reboot** command.

You cannot issue the **request system reboot** command once you schedule a commit operation for a specific time in the future.

You cannot commit a configuration when a scheduled commit is pending. For information about how to cancel a scheduled configuration by means of the **clear** command, see the *JUNOS System Basics and Services Command Reference*.

Monitoring the JUNOS Commit Process

To monitor the JUNOS commit process, use the **display detail** command after the pipe with the **commit** command:

```
user@host# commit | display detail
```

For example:

```
[edit]
user@host# commit | display detail
2003-09-22 15:39:39 PDT: exporting juniper.conf
2003-09-22 15:39:39 PDT: setup foreign files
2003-09-22 15:39:39 PDT: propagating foreign files
2003-09-22 15:39:39 PDT: complete foreign files
2003-09-22 15:39:40 PDT: copying configuration to juniper.data+
2003-09-22 15:39:40 PDT: dropping unchanged foreign files
2003-09-22 15:39:40 PDT: daemons checking new configuration
2003-09-22 15:39:41 PDT: commit wrapup...
2003-09-22 15:39:42 PDT: activating '/var/etc/ntp.conf'
2003-09-22 15:39:42 PDT: activating '/var/etc/kmd.conf'
2003-09-22 15:39:42 PDT: activating '/var/db/juniper.data'
2003-09-22 15:39:42 PDT: notifying daemons of new configuration
2003-09-22 15:39:42 PDT: signaling 'Firewall daemon', pid 24567, signal 1,
status 0
2003-09-22 15:39:42 PDT: signaling 'Interface daemon', pid 24568, signal 1,
status 0
2003-09-22 15:39:43 PDT: signaling 'Routing protocol daemon', pid 25679,
signal 1, status 0
2003-09-22 15:39:43 PDT: signaling 'MIB2 daemon', pid 24549, signal 1,
status 0
2003-09-22 15:39:43 PDT: signaling 'NTP daemon', pid 37863, signal 1, status 0
2003-09-22 15:39:43 PDT: signaling 'Sonet APS daemon', pid 24551, signal 1,
status 0
2003-09-22 15:39:43 PDT: signaling 'VRRP daemon', pid 24552, signal 1,
status 0
2003-09-22 15:39:43 PDT: signaling 'PFE daemon', pid 2316, signal 1, status 0
2003-09-22 15:39:43 PDT: signaling 'Traffic sampling control daemon', pid 24553
signal 1, status 0
2003-09-22 15:39:43 PDT: signaling 'IPsec Key Management daemon', pid
24556, signal 1, status 0
2003-09-22 15:39:43 PDT: signaling 'Forwarding UDP daemon', pid 2320,
signal 1, status 0
commit complete
```

Adding a Comment to Describe the Committed Configuration

You can include a comment that describes changes to the committed configuration. To do so, include the `commit comment` statement. The comment can be as long as 512 bytes and you must type it on a single line.

```
[edit]
user@host# commit comment comment-string
```

comment-string is the text of the comment.



NOTE: You cannot include a comment with the `commit check` command.

To add a comment to the `commit` command, include the `comment` statement after the `commit` command:

```
[edit]
user@host# commit comment "add user joe"
commit complete
[edit]
user@host#
```

To add a comment to the `commit confirmed` command, include the `comment` statement after the `commit confirmed` command:

```
[edit]
user@host# commit confirmed comment "add customer to port 27"
commit confirmed will be automatically rolled back in 10 minutes unless confirmed
commit complete
[edit]
user@host#
```

To view these commit comments, issue the `show system commit operational mode` command.

Updating the Alternate Boot Drive

After you commit the configuration and are satisfied that it is running successfully, you should issue the `request system snapshot` command to back up the new software onto the `/altconfig` file system. If you do not issue the `request system snapshot` command, the configuration on the alternate boot drive will be out of sync with the configuration on the primary boot drive.

The `request system snapshot` command backs up the root file system to `/altroot`, and `/config` to `/altconfig`. The root and `/config` file systems are on the router's flash drive, and the `/altroot` and `/altconfig` file systems are on the router's hard disk (if available).



NOTE: To back up the file system on a J Series Services Router, you must specify a media type (primary compact flash drive, removable compact flash drive, or USB storage device) for backup. For more information, see the *J Series Services Router Administration Guide*.

After you issue the `request system snapshot` command, you cannot return to the previous version of the software because the running and backup copies of the software are identical.

When Multiple Users Configure the Software

Up to 32 users can be in configuration mode simultaneously, and they all can be making changes to the configuration. All changes made by all users are visible to everyone editing the configuration—the changes become visible as soon as the user presses the Enter key at the end of a command that changes the configuration, such as `set`, `edit`, or `delete`.

When any of the users editing the configuration issues a `commit` command, all changes made by all users are checked and activated.

Forms of the configure Command

The JUNOS Software supports three forms of the **configure** command: **configure**, **configure private**, and **configure exclusive**. These forms control how users edit and commit configurations and can be useful when multiple users configure the software. See Table 9 on page 89.

Table 9: Forms of the configure Command

Command	Edit Access	Commit Access
configure	<ul style="list-style-type: none"> ■ No one can lock the configuration. All users can make configuration changes. <p>When you enter configuration mode, the CLI displays the following information:</p> <ul style="list-style-type: none"> ■ A list of other users editing the configuration. ■ Hierarchy levels the users are viewing or editing. ■ Whether the configuration has been changed, but not committed. 	<ul style="list-style-type: none"> ■ No one can lock the configuration. All users can commit all changes to the configuration. ■ If you and another user make changes and the other user commits changes, your changes are committed as well.
configure exclusive	<ul style="list-style-type: none"> ■ One user locks the configuration and makes changes without interference from other users. ■ Other users can enter and exit configuration mode, but they cannot change the configuration. ■ If you enter configuration mode while another user has locked the configuration (with the configure exclusive command), the CLI displays the user and the hierarchy level the user is viewing or editing. ■ If you enter configuration mode while another user has locked the configuration, you can forcibly log out that user with the request system logout operational mode command. For details, see the <i>JUNOS System Basics and Services Command Reference</i>. 	

Table 9: Forms of the configure Command *(continued)*

Command	Edit Access	Commit Access
configure private	<ul style="list-style-type: none"> ■ Multiple users can edit the configuration at the same time. ■ Each user has a private candidate configuration to edit independently of other users. 	<ul style="list-style-type: none"> ■ When you commit the configuration, the router verifies that the operational (running) configuration has not been modified by another user before accepting your private candidate configuration as the new operational configuration. ■ If the configuration has been modified by another user, you can merge the modifications into your private candidate configuration and attempt to commit again.

Example: Using the configure Command

If, when you enter configuration mode, another user is also in configuration mode, a message shows who the user is and what part of the configuration that user is viewing or editing:

```
user@host> configure
Entering configuration mode
Current configuration users:
root terminal p3 (pid 1088) on since 1999-05-13 01:03:27 EDT
[edit interfaces so-3/0/0 unit 0 family inet]
The configuration has been changed but not committed
[edit]
user@host#
```

If, when you enter configuration mode, the configuration contains changes that have not been committed, a message appears:

```
user@host> configure
Entering configuration mode
The configuration has been changed but not committed
[edit]
user@host#
```

Displaying Users Currently Editing the Configuration

To display the users currently editing the configuration, use the **status** configuration mode command:

```
user@host# status
Users currently editing the configuration:
rchen terminal p0 (pid 55691) on since 2006-03-01 13:17:25 PST
[edit interfaces]
```

The system displays who is editing the configuration (*rchen*), where the user is logged in (*terminal p0*), the date and time the user logged in (*2006-03-01 13:17:25 PST*), and what level of the hierarchy the user is editing (*[edit interfaces]*).

If you issue the **status** configuration mode command and a user has scheduled a candidate configuration to become active for a future time, the system displays who scheduled the commit (*root*), where the user is logged in (*terminal d0*), the date and time the user logged in (*2002-10-31 14:55:15 PST*), and that a commit is pending (*commit at*).

```
[edit]
user@host# status
Users currently editing the configuration:
root terminal d0 (pid 767) on since 2002-10-31 14:55:15 PST, idle 00:03:09
commit at
```

For information about how to schedule a commit, see “Scheduling a JUNOS Commit Operation” on page 85.

If you issue the **status** configuration mode command and a user is editing the configuration in configure exclusive mode, the system displays who is editing the configuration (*root*), where the user is logged in (*terminal d0*), the date and time the user logged in (*2002-11-01 13:05:11 PST*), and that a user is editing the configuration in configure exclusive mode (*exclusive [edit]*).

```
[edit]
user@host# status
Users currently editing the configuration:
root terminal d0 (pid 2088) on since 2002-11-01 13:05:11 PST
exclusive [edit]
```

For more information about the configure exclusive mode, see “Using the configure exclusive Command” on page 91.

Using the configure exclusive Command

If you enter configuration mode with the **configure exclusive** command, you lock the candidate *global* configuration (also known as the *shared configuration* or *shared configuration database*) for as long as you remain in configuration mode, allowing you to make changes without interference from other users. Other users can enter and exit configuration mode, but they cannot change the configuration.

If another user has locked the configuration, and you need to forcibly log the person out, enter the operational mode command **request system logout pid *pid_number***.

If you enter configuration mode and another user is also in configuration mode and has locked the configuration, a message indicates who the user is and what portion of the configuration that user is viewing or editing:

```
user@host> configure
Entering configuration mode
Users currently editing the configuration:
root terminal p3 (pid 1088) on since 2000-10-30 19:47:58 EDT, idle 00:00:44
```

```
exclusive [edit interfaces so-3/0/0 unit 0 family inet]
```

In configure exclusive mode, any uncommitted changes are discarded when you exit:

```
user@host> configure exclusive
warning: uncommitted changes will be discarded on exit
Entering configuration mode
[edit]
user@host# set system host-name cool
[edit]
user@host# quit
The configuration has been changed but not committed
warning: Auto rollback on exiting 'configure exclusive'
Discard uncommitted changes? [yes,no] (yes)
warning: discarding uncommitted changes
load complete
Exiting configuration mode
```

When you use the **yes** option to exit configure exclusive mode, JUNOS Software discards your uncommitted changes and rolls back your configuration. The **no** option allows you to continue editing or to commit your changes in configure exclusive mode.

When a user exits from configure exclusive mode while another user is in configure private mode, JUNOS Software will roll back any uncommitted changes.

Updating the configure private Configuration

When you are in configure private mode, you must work with a copy of the most recently committed shared configuration. If the global configuration changes, you can issue the **update** command to update your private candidate configuration. When you do this, your private candidate configuration contains a copy of the most recently committed configuration with your private changes merged in. For example:

```
[edit]
user@host# update
[edit]
user@host#
```



NOTE: Merge conflicts can occur when you issue the **update** command.

You can also issue the **rollback** command to discard your private candidate configuration changes and obtain the most recently committed configuration:

```
[edit]
user@host# rollback
[edit]
user@host#
```


Displaying set Commands from the Configuration

In configuration mode, you can display the configuration as a series of configuration mode commands required to re-create the configuration. This is useful if you are not familiar with how to use configuration mode commands or if you want to cut, paste, and edit the displayed configuration. For information about the **set** command, see “Displaying the Current Configuration” on page 70.

To display the configuration as a series of configuration mode commands required to re-create the configuration from the top level of the hierarchy as **set** commands, issue the **show configuration** mode command with the **| display set** option:

```
user@host# show | display set
```

This topic contains the following examples:

- Example: Displaying set Commands from the Configuration on page 93
- Example: Displaying Required set Commands at the Current Hierarchy Level on page 94
- Example: Displaying set Commands with the match Option on page 94

Example: Displaying set Commands from the Configuration

Display the **set** commands from the configuration at the **[edit interfaces]** hierarchy level:

```
[edit interfaces fe-0/0/0]
user@host# show
unit 0 {
  family inet {
    address 192.107.1.230/24;
  }
  family iso;
  family mpls;
}
inactive: unit 1 {
  family inet {
    address 10.0.0.1/8;
  }
}
user@host# show | display set
set interfaces fe-0/0/0 unit 0 family inet address 192.107.1.230/24
set interfaces fe-0/0/0 unit 0 family iso
set interfaces fe-0/0/0 unit 0 family mpls
set interfaces fe-0/0/0 unit 1 family inet address 10.0.0.1/8
deactivate interfaces fe-0/0/0 unit 1
```

To display the configuration as a series of configuration mode commands required to re-create the configuration from the current hierarchy level, issue the **show configuration** mode command with the **| display set relative** option:

```
user@host# show | display set relative
```

Example: Displaying Required set Commands at the Current Hierarchy Level

Display the configuration as a series of configuration mode commands required to re-create the configuration from the current hierarchy level:

```
[edit interfaces fe-0/0/0]
user@host# show
unit 0 {
  family inet {
    address 192.107.1.230/24;
  }
  family iso;
  family mpls;
}
inactive: unit 1 {
  family inet {
    address 10.0.0.1/8;
  }
}
user@host# show | display set relative
set unit 0 family inet address 192.107.1.230/24
set unit 0 family iso
set unit 0 family mpls
set unit 1 family inet address 10.0.0.1/8
deactivate unit 1
```

To display the configuration as set commands and search for text matching a regular expression by filtering output, specify the match option after the pipe:

```
user@host# show | display set | match regular-expression
```

Example: Displaying set Commands with the match Option

Display IP addresses associated with an interface:

```
xe-2/3/0 {
  unit 0 {
    family inet {
      address 192.107.9.106/30;
    }
  }
}
so-5/1/0 {
  unit 0 {
    family inet {
      address 192.107.9.15/32 {
        destination 192.107.9.192;
      }
    }
  }
}
lo0 {
  unit 0 {
```

```

        family inet {
            address 127.0.0.1/32;
        }
    }
}
user@host# show interfaces | display set | match address
set interfaces xe-2/3/0 unit 0 family inet address 192.168.9.106/30
set interfaces so-5/1/0 unit 0 family inet address 192.168.9.15/32 destination
192.168.9.192
set interfaces lo0 unit 0 family inet address 127.0.0.1/32

```

Displaying Additional Information About the Configuration

In configuration mode only, to display additional information about the configuration, use the `display detail` command after the pipe in conjunction with a `show` command. The additional information includes the help string that explains each configuration statement and the permission bits required to add and modify the configuration statement.

```
user@host# show <hierarchy-level> | display detail
```

For example:

```

[edit]
user@host# show | display detail
##
## version: Software version information
## require: system
##
version "3.4R1 [tlim]";
system {
    ##
    ## host-name: Host name for this router
    ## match: ^[:alnum:]._]+$
    ## require: system
    ##
}
host-name router-name;
##
## domain-name: Domain name for this router
## match: ^[:alnum:]._]+$
## require: system
##
domain-name isp.net;
##
## backup-router: Address of router to use while booting
##
backup-router 192.168.100.1;
root-authentication {
    ##
    ## encrypted-password: Encrypted password string
    ##
    encrypted-password "$1$BYJQE$/ocQof8pmcm7MSGK0"; # SECRET-DATA
}
##

```

```

## name-server: DNS name servers
## require: system
##
name-server {
    ##
    ## name-server: DNS name server address
    ##
    208.197.1.0;
}
login {
    ##
    ## class: User name (login)
    ## match: ^[:alnum:][_]+$
    ##
    class super-user {
        ##
        ## permissions: Set of permitted operation categories
        ##
        permissions all;
    }
    ...
    ##
    ## services: System services
    ## require: system
    ##
    services {
        ## services: Service name
        ##
        ftp;
        ##
        ## services: Service name
        ##
        telnet;
        ##
    }
    syslog {
        ##
        ## file-name: File to record logging data
        ##
        file messages {
            ##
            ## Facility type
            ## Level name
            ##
            any notice;
            ##
            ## Facility type
            ## Level name
            ##
            authorization info;
        }
    }
}
chassis {
    alarm {
        sonet {

```

```

    ##
    ## lol: Loss of light
    ## alias: loss-of-light
    ##
    lol red;
  }
}
}
interfaces {
  ##
  ## Interface name
  ##
  at-2/1/1 {
    atm-options {
      ##
      ## vpi: Virtual path index
      ## range: 0 .. 255
      ## maximum-vcs: Maximum number of virtual circuits on this VP
      ##
      vpi 0 maximum-vcs 512;
    }
    ##
    ## unit: Logical unit number
    ## range: 0 .. 16384
    ##
    unit 0 {
      ##
      ## vci: ATM point-to-point virtual circuit identifier ([vpi.]vci)
      ##
      vci 0.128;
    }
  }
}
...

```


Chapter 6

Managing Configurations

This chapter provides basic information about managing configurations.

Topics include:

- Understanding How the JUNOS Configuration Is Stored on page 99
- Returning to the Most Recently Committed Configuration on page 100
- Returning to a Previously Committed JUNOS Configuration on page 100
- Loading a Configuration from a File on page 106
- Examples: Loading a Configuration from a File on page 108
- Additional Details About Specifying JUNOS Statements and Identifiers on page 110
- Synchronizing Routing Engines on page 113

Understanding How the JUNOS Configuration Is Stored

When you edit a configuration, you work in a copy of the current configuration to create a candidate configuration. The changes you make to the candidate configuration are visible in the CLI immediately, so if multiple users are editing the configuration at the same time, all users can see all changes.

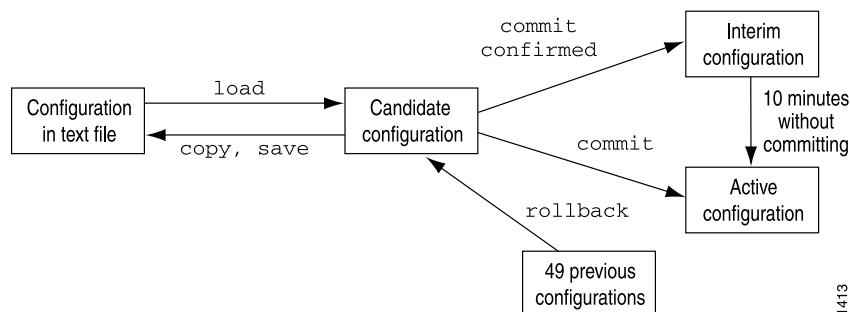
To have a candidate configuration take effect, you *commit* the changes. At this point, the candidate file is checked for proper syntax, activated, and marked as the current, operational software configuration file. If multiple users are editing the configuration, when you commit the candidate configuration, all changes made by all the users take effect.

In addition to saving the current configuration, the CLI saves the current operational version and the previous 49 versions of committed configurations. The most recently committed configuration is version 0 (the current operational version, which is the default configuration that the system returns to if you roll back to a previous configuration), and the oldest saved configuration is version 49.

The currently operational JUNOS Software configuration is stored in the file `juniper.conf`, and the last three committed configurations are stored in the files `juniper.conf.1`, `juniper.conf.2`, and `juniper.conf.3`. These four files are located in the directory `/config`, which is on the router's flash drive. The remaining 46 previous versions of committed configurations, the files `juniper.conf.4` through `juniper.conf.49`, are stored in the directory `/var/db/config` on the hard disk.

Figure 10 on page 100 illustrates the various router configuration states and the configuration mode commands you use to load, commit, copy, save, or roll back the configuration.

Figure 10: Commands for Storing and Modifying the Router Configuration



Returning to the Most Recently Committed Configuration

To return to the most recently committed configuration and load it into configuration mode without activating it, use the `rollback` configuration mode command:

```
[edit]
user@host# rollback
```

load complete

To activate the configuration to which you rolled back, use the `commit` command:

```
[edit]
user@host# rollback
load complete
[edit]
user@host# commit
```

Returning to a Previously Committed JUNOS Configuration

This topic explains how you can return to a configuration prior to the most recently committed one, and contains the following sections:

- Returning to a Configuration Prior to the One Most Recently Committed on page 101
- Displaying Previous Configurations on page 101
- Comparing Configuration Changes with a Prior Version on page 102
- Creating and Returning to a Rescue Configuration on page 104
- Saving a Configuration to a File on page 105

Returning to a Configuration Prior to the One Most Recently Committed

To return to a configuration prior to the most recently committed one, include the number in the `rollback` command. *number* can be a number in the range 0 through 49. The most recently saved configuration is number 0 (which is the default configuration to which the system returns), and the oldest saved configuration is number 49.

```
[edit]
user@host# rollback number
load complete
```

Displaying Previous Configurations

To display previous configurations, including the rollback number, date, time, the name of the user who committed changes, and the method of commit, use the `rollback ?` command.

```
[edit]
user@host# rollback ?
Possible completions:
<[Enter]> Execute this command
<number> Numeric argument
0          2005-02-27 12:52:10 PST by abc via cli
1          2005-02-26 14:47:42 PST by def via cli
2          2005-02-14 21:55:45 PST by ghi via cli
3          2005-02-10 16:11:30 PST by jkl via cli
4          2005-02-10 16:02:35 PST by mno via cli
5          2005-03-16 15:10:41 PST by pqr via cli
6          2005-03-16 14:54:21 PST by stu via cli
7          2005-03-16 14:51:38 PST by vwx via cli
8          2005-03-16 14:43:29 PST by yzz via cli
9          2005-03-16 14:15:37 PST by abc via cli
10         2005-03-16 14:13:57 PST by def via cli
11         2005-03-16 12:57:19 PST by root via other
12         2005-03-16 10:45:23 PST by root via other
13         2005-03-16 10:08:13 PST by root via other
14         2005-03-16 01:20:56 PST by root via other
15         2005-03-16 00:40:37 PST by ghi via cli
16         2005-03-16 00:39:29 PST by jkl via cli
17         2005-03-16 00:32:36 PST by mno via cli
18         2005-03-16 00:31:17 PST by pqr via cli
19         2005-03-15 19:59:00 PST by stu via cli
20         2005-03-15 19:53:39 PST by vwx via cli
21         2005-03-15 18:07:19 PST by yzz via cli
22         2005-03-15 17:59:03 PST by abc via cli
23         2005-03-15 15:05:14 PST by def via cli
24         2005-03-15 15:04:51 PST by ghi via cli
25         2005-03-15 15:03:42 PST by jkl via cli
26         2005-03-15 15:01:52 PST by mno via cli
27         2005-03-15 14:58:34 PST by pqr via cli
28         2005-03-15 13:09:37 PST by root via other
29         2005-03-12 11:01:20 PST by stu via cli
```

```

30      2005-03-12 10:57:35 PST by vwx via cli
31      2005-03-11 10:25:07 PST by yzz via cli
32      2005-03-10 23:40:58 PST by abc via cli
33      2005-03-10 23:40:38 PST by def via cli
34      2005-03-10 23:14:27 PST by ghi via cli
35      2005-03-10 23:10:16 PST by jkl via cli
36      2005-03-10 23:01:51 PST by mno via cli
37      2005-03-10 22:49:57 PST by pqr via cli
38      2005-03-10 22:24:07 PST by stu via cli
39      2005-03-10 22:20:14 PST by vwx via cli
40      2005-03-10 22:16:56 PST by yzz via cli
41      2005-03-10 22:16:41 PST by abc via cli
42      2005-03-10 20:44:00 PST by def via cli
43      2005-03-10 20:43:29 PST by ghi via cli
44      2005-03-10 20:39:14 PST by jkl via cli
45      2005-03-10 20:31:30 PST by root via other
46      2005-03-10 18:57:01 PST by mno via cli
47      2005-03-10 18:56:18 PST by pqr via cli
48      2005-03-10 18:47:49 PST by stu via cli
49      2005-03-10 18:47:34 PST by vw via cli
|Pipe through a command
[edit]

```

Comparing Configuration Changes with a Prior Version

In configuration mode only, when you have made changes to the configuration and want to compare the candidate configuration with a prior version, you can use the **compare** command to display the configuration. The **compare** command compares the candidate configuration with either the current committed configuration or a configuration file and displays the differences between the two configurations. To compare configurations, specify the **compare** command after the pipe:

```

[edit]
user@host# show | compare (filename | rollback n)

```

filename is the full path to a configuration file. The file must be in the proper format: a hierarchy of statements.

n is the index into the list of previously committed configurations. The most recently saved configuration is number 0, and the oldest saved configuration is number 49. If you do not specify arguments, the candidate configuration is compared against the active configuration file (`/config/juniper.conf`).

The comparison output uses the following conventions:

- Statements that are only in the candidate configuration are prefixed with a plus sign (+).
- Statements that are only in the comparison file are prefixed with a minus sign (-).
- Statements that are unchanged are prefixed with a single blank space ().

The following example shows various changes, then a comparison of the candidate configuration with the active configuration, showing only the changes made at the [edit protocols bgp] hierarchy level:

```
[edit]
user@host# edit protocols bgp
[edit protocols bgp]
user@host# show
group my-group {
    type internal;
    hold-time 60;
    advertise-inactive;
    allow 1.1.1.1/32;
}
group fred {
    type external;
    peer-as 33333;
    allow 2.2.2.2/32;
}
group test-peers {
    type external;
    allow 3.3.3.3/32;
}
[edit protocols bgp]
user@host# set group my-group hold-time 90
[edit protocols bgp]
user@host# delete group my-group advertise-inactive
[edit protocols bgp]
user@host# set group fred advertise-inactive
[edit protocols bgp]
user@host# delete group test-peers
[edit protocols bgp]
user@host# show | compare
[edit protocols bgp group my-group]
- hold-time 60;
+ hold-time 90;
- advertise-inactive;
[edit protocols bgp group fred]
+ advertise-inactive;
[edit protocols bgp]
- group test-peers {
-   type external;
-   allow 3.3.3.3/32;
- }
[edit protocols bgp]
user@host# show
group my-group {
    type internal;
    hold-time 90;
    allow 1.1.1.1/32;
}
group fred {
    type external;
    advertise-inactive;
    peer-as 33333;
    allow 2.2.2.2/32;
}
```

```
}
```

Creating and Returning to a Rescue Configuration

A *rescue configuration* allows you to define a known working configuration or a configuration with a known state that you can roll back to at any time. This alleviates the necessity of having to remember the rollback number with the **rollback** command. You use the rescue configuration when you need to roll back to a known configuration or as a last resort if your router configuration and the backup configuration files become damaged beyond repair.

To save the most recently committed configuration as the rescue configuration so that you can return to it at any time, issue the **request system configuration rescue save** command:

```
user@host> request system configuration rescue save
user@host>
```

To return to the rescue configuration, use the **rollback rescue** configuration mode command:

```
[edit]
user@host# rollback rescue
load complete
```



NOTE: If the rescue configuration does not exist, or if the rescue configuration is not a complete, viable configuration, the **rollback** command fails, an error message appears, and the current configuration remains active.

To activate the rescue configuration that you have loaded, use the **commit** command:

```
[edit]
user@host# rollback rescue
load complete
[edit]
user@host# commit
```

To delete an existing rescue configuration, issue the **request system configuration rescue delete** command:

```
user@host> request system configuration rescue delete
user@host>
```

For more information about the **request system configuration rescue delete** and **request system configuration rescue save** commands, see the *JUNOS System Basics and Services Command Reference*.

Saving a Configuration to a File

You might want to save the configuration to a file so that you can edit it with a text editor of your choice. You can save your current configuration to an ASCII file, which saves the configuration in its current form, including any uncommitted changes. If more than one user is modifying the configuration, all changes made by all users are saved.

To save software configuration changes to an ASCII file, use the **save** configuration mode command:

```
[edit]
user@host# save filename
[edit]
user@host#
```

The contents of the current level of the statement hierarchy (and below) are saved, along with the statement hierarchy containing it. This allows a section of the configuration to be saved, while fully specifying the statement hierarchy.

By default, the configuration is saved to a file in your home directory, which is on the flash drive. For information about specifying the filename, see “Specifying Filenames and URLs” on page 51.

When you issue this command from anywhere in the hierarchy (except the top level), a **replace** tag is automatically included at the beginning of the file. You can use the **replace** tag to control how a configuration is loaded from a file. For more information, see “Loading a Configuration from a File” on page 106.

```
user@host> file show /var/home/user/myconf
replace:
protocols {
  bgp {
    disable;
    group int {
      type internal;
    }
  }
  isis {
    disable;
    interface all {
      level 1 disable;
    }
    interface fxp0.0 {
      disable;
    }
  }
  ospf {
    traffic-engineering;
    reference-bandwidth 4g;
    ...
  }
}
```

Loading a Configuration from a File

You can create a file, copy the file to the local router, and then load the file into the CLI. After you have loaded the file, you can commit it to activate the configuration on the router, or you can edit the configuration interactively using the CLI and commit it at a later time.

You can also create a configuration while typing at the terminal and then load it. Loading a configuration from the terminal is generally useful when you are cutting existing portions of the configuration and pasting them elsewhere in the configuration.

To load an existing configuration file that is located on the router, use the **load** configuration mode command:

```
[edit]
user@host# load (factory-default | merge | override | patch | replace | set | update)
          filename <relative>
```

For information about specifying the filename, see “Specifying Filenames and URLs” on page 51.

To load a configuration from the terminal, use the following version of the **load** configuration mode command. Type ^D to end input.

```
[edit]
user@host# load (factory-default | merge | override | patch | replace | set | update)
          terminal <relative>
```

To replace an entire configuration, specify the **override** option at any level of the hierarchy.

An override operation discards the current candidate configuration and loads the configuration in *filename* or the one that you type at the terminal. When you use the **override** option and commit the configuration, all system processes reparse the configuration. For an example, see Figure 11 on page 108.

To replace portions of a configuration, specify the **replace** option. For this operation to work, you must include **replace:** tags in the file or configuration you type at the terminal. The software searches for the **replace:** tags, deletes the existing statements of the same name, if any, and replaces them with the incoming configuration. If there is no existing statement of the same name, the **replace** operation adds to the configuration the statements marked with the **replace:** tag. For an example, see Figure 12 on page 108.

To replace only the configuration that has changed, specify the **update** option at any level of the hierarchy. An update operation compares the current configuration and the current candidate configuration, and loads only the changes between these configurations in *filename* or the one that you type at the terminal. When you use the update operation and commit the configuration, JUNOS Software attempts to notify the smallest set of system processes that are affected by the configuration change.

To combine the current configuration and the configuration in *filename* or the one that you type at the terminal, specify the **merge** option. A merge operation is useful when you are adding a new section to an existing configuration. If the existing configuration and the incoming configuration contain conflicting statements, the statements in the incoming configuration override those in the existing configuration. For an example, see Figure 13 on page 109.

To change part of the configuration with a patch file and mark only those parts as changed, specify the **patch** option. For an example, see Figure 14 on page 109.

To use the **merge**, **replace**, **set**, or **update** option without specifying the full hierarchy level, specify the **relative** option. For example:

```
[edit system]
user@host# show static-host-mapping
bob sysid 987.654.321ab
[edit system]
user@host# load replace terminal relative
[Type ^D at a new line to end input]
replace: static-host-mapping {
    bob sysid 0123.456.789bc;
}
load complete
[edit system]
user@host# show static-host-mapping
bob sysid 0123.456.789bc;
```

If, in an override or merge operation, you specify a file or type text that contains **replace:** tags, the **replace:** tags are ignored and the **override** or **merge** operation is performed.

If you are performing a **replace** operation and the file you specify or text you type does not contain any **replace:** tags, the **replace** operation is effectively equivalent to a **merge** operation. This might be useful if you are running automated scripts and cannot know in advance whether the scripts need to perform a **replace** or a **merge** operation. The scripts can use the **replace** operation to cover either case.

To load a configuration that contains the **set** configuration mode command, specify the **set** option. This option executes the configuration instructions line by line as they are stored in a file or from a terminal. The instructions can contain any configuration mode command, such as **set**, **edit**, **exit**, and **top**. For an example, see Figure 15 on page 110.

To copy a configuration file from another network system to the local router, you can use the SSH and Telnet utilities, as described in the *JUNOS System Basics and Services Command Reference*.



NOTE: If you are using JUNOS Software in a Common Criteria environment, system log messages are created whenever a **secret** attribute is changed (for example, password changes or changes to the RADIUS shared secret). These changes are logged during the following configuration load operations:

load merge
load replace
load override
load update

For more information, see the *Secure Configuration Guide for Common Criteria and JUNOS FIPS*.

Examples: Loading a Configuration from a File

Figure 11: Overriding the Current Configuration



Figure 12: Using the replace Option



Figure 13: Using the merge Option

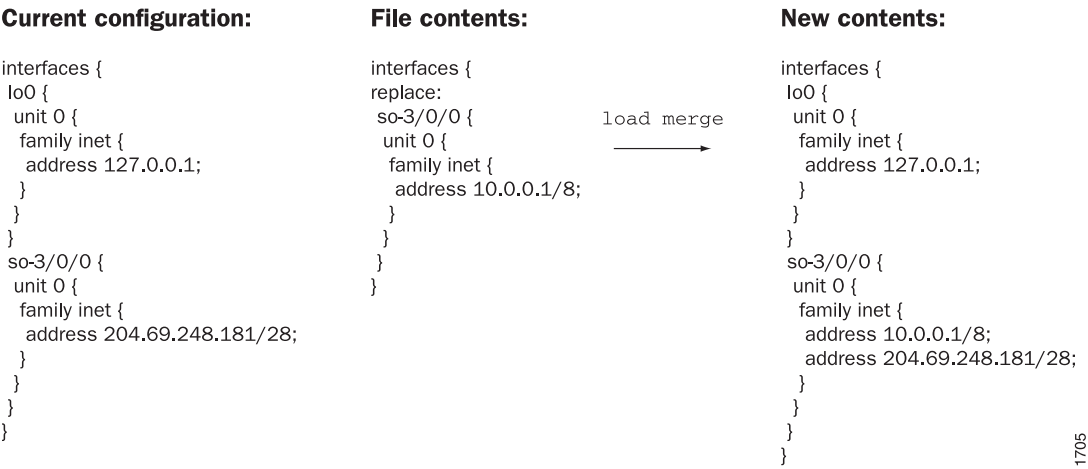


Figure 14: Using a Patch File

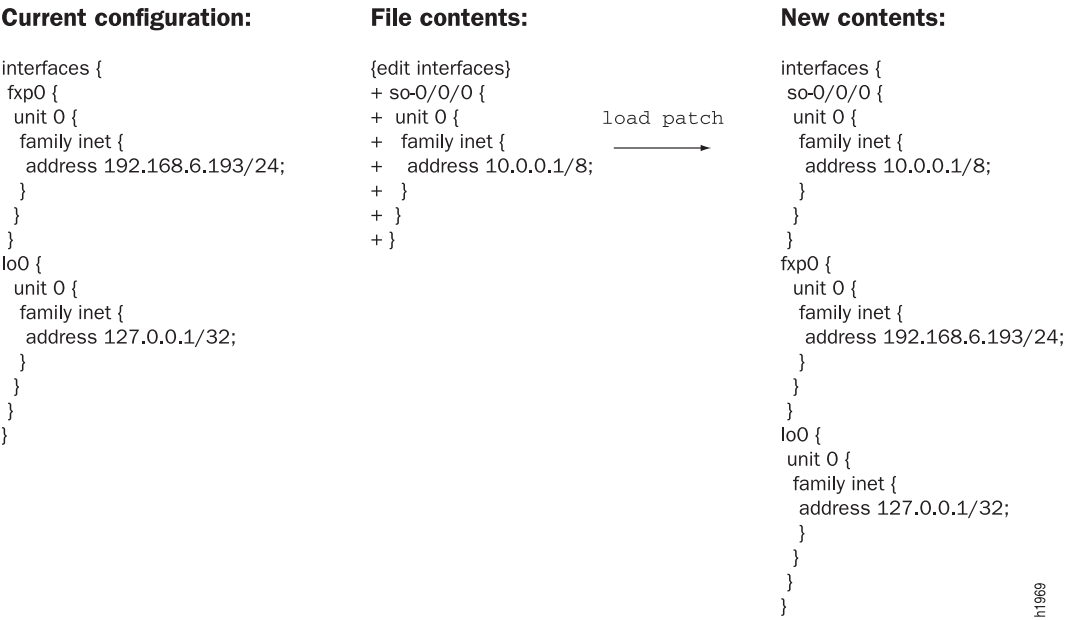


Figure 15: Using the set Option**File contents:**

```
edit access
set profile p1 client cl ike
edit profile p1 client cl ike
set pre-shared-key ascii-text "abcd"
set allowed-proxy-pair local 1.1.1.1 remote 2.2.2.2
exit
deactivate profile p1
top
edit system
set radius-server 1.1.1.1
```

```
load set
```

**New contents:**

```
system {
  radius-server {
    1.1.1.1;
  }
}
access {
  inactive: profile p1 {
    client cl {
      ike {
        allowed-proxy-pair local 1.1.1.1/32 remote 2.2.2.2/32;
        pre-shared-key ascii-text "$9$Ydg4ZDjqf5FVw"; ## SECRET-DATA
      }
    }
  }
}
```

g017216

Additional Details About Specifying JUNOS Statements and Identifiers

This section provides more detailed information about CLI container and leaf statements so that you can better understand how you must specify them when creating ASCII configuration files. It also describes how the CLI performs type-checking to verify that the data you entered is in the correct format.

- Specifying Statements on page 110
- Performing CLI Type-Checking on page 112

Specifying Statements

Statements are shown one of two ways, either with braces or without:

- Statement name and identifier, with one or more lower level statements enclosed in braces:

```
statement-name1 identifier-name {
  statement-name2;
  additional-statements;
}
```

- Statement name, identifier, and a single identifier:

```
statement-name identifier-name1 identifier-name2;
```

The *statement-name* is the name of the statement.

The *identifier-name* is a name or other string that uniquely identifies an instance of a statement. An identifier is used when a statement can be specified more than once in a configuration.

When specifying a statement, you must specify either a statement name or an identifier name, or both, depending on the statement hierarchy.

You specify identifiers in one of the following ways:

- *identifier-name*—The *identifier-name* is a keyword used to uniquely identify a statement when a statement can be specified more than once in a statement.
- *identifier-name value*—The *identifier-name* is a keyword, and the *value* is a required option variable.
- *identifier-name [value1 value2 value3 ...]*—The *identifier-name* is a keyword that accepts multiple values. The brackets are required when you specify a set of values; however, they are optional when you specify only one value.

The following examples illustrate how statements and identifiers are specified in the configuration:

```
protocol {          # Top-level statement (statement-name).
  ospf {           # Statement under "protocol" (statement-name).
    area 0.0.0.0 {  # OSPF area "0.0.0.0" (statement-name identifier-name),
      interface so-0/0/0 { # which contains an interface named "so-0/0/0."
        hello-interval 25; # Identifier and value (identifier-name value).
        priority 2;       # Identifier and value (identifier-name value).
        disable;         # Flag identifier (identifier-name).
      }
      interface so-0/0/1; # Another instance of "interface," named so-0/0/1,
    }                   # this instance contains no data, so no braces
  }                   # are displayed.
}

policy-options {   # Top-level statement (statement-name).
  term term1 {     # Statement under "policy-options"
    # (statement-name value).
    from {         # Statement under "term" (statement-name).
      route-filter 10.0.0.0/8 orlonger reject; # One identifier ("route-filter")
    with
      route-filter 127.0.0.0/8 orlonger reject; # multiple values.
      route-filter 128.0.0.0/16 orlonger reject;
      route-filter 149.20.64.0/24 orlonger reject;
      route-filter 172.16.0.0/12 orlonger reject;
      route-filter 191.255.0.0/16 orlonger reject;
    }
    then {         # Statement under "term" (statement-name).
      next term;   # Identifier (identifier-name).
    }
  }
}
```

When you create an ASCII configuration file, you can specify statements and identifiers in one of the following ways. However, each statement has a preferred style, and the CLI uses that style when displaying the configuration in response to a configuration mode **show** command.

- Statement followed by identifiers:

```
statement-name identifier-name [...] identifier-name value [...];
```

- Statement followed by identifiers enclosed in braces:

```
statement-name {  
    identifier-name;  
    [...]  
    identifier-name value;  
    [...]  
}
```

- For some repeating identifiers, you can use one set of braces for all the statements:

```
statement-name {  
    identifier-name value1;  
    identifier-name value2;  
}
```

Performing CLI Type-Checking

When you specify identifiers and values, the CLI performs type-checking to verify that the data you entered is in the correct format. For example, for a statement in which you must specify an IP address, the CLI requires you to enter an address in a valid format. If you have not, an error message indicates what you need to type. Table 10 on page 112 lists the data types the CLI checks.

Table 10: CLI Configuration Input Types

Data Type	Format	Examples
Physical interface name (used in the [edit interfaces] hierarchy)	<i>type-fpc/pic/port</i>	Correct: so-0/0/1 Incorrect: so-0
Full interface name	<i>type-fpc/pic/port<:channel>.logical</i>	Correct: so-0/0/1.0 Incorrect: so-0/0/1
Full or abbreviated interface name (used in places other than the [edit interfaces] hierarchy)	<i>type-<fpc/>pic/port>><<:channel>.logical></i>	Correct: so, so-1, so-1/2/3:4.5

Table 10: CLI Configuration Input Types *(continued)*

Data Type	Format	Examples
IP address	<i>Oxhex-bytes</i> octet<.octet<.octet.<octet>>>	<p>Correct: 1.2.3.4, 0x01020304, 128.8.1, 128.8</p> <p>Sample translations:</p> <p>1.2.3 becomes 1.2.3.0 0x01020304 becomes 1.2.3.4 0x010203 becomes 0.1.2.3</p>
IP address (destination prefix) and prefix length	<i>Oxhex-bytes</i> </length>octet<octet <octet.<octet>>></length>	<p>Correct: 10/8, 128.8/16, 1.2.3.4/32, 1.2.3.4</p> <p>Sample translations:</p> <p>1.2.3 becomes 1.2.3.0/32 0x01020304 becomes 1.2.3.4/32 0x010203 becomes 0.1.2.3/32 default becomes 0.0.0.0/0</p>
International Organization for Standardization (ISO) address	<i>hex-nibble</i> <hex-nibble ...>	<p>Correct: 47.1234.2345.3456.00, 47123423453456.00, 47.12.34.23.45.34.56.00</p> <p>Sample translations:</p> <p>47123456 becomes 47.1234.56 47.12.34.56 becomes 47.1234.56 4712.3456 becomes 47.1234.56</p>
OSPF area identifier (ID)	<i>Oxhex-bytes</i> octet<.octet<.octet.< octet >>> <i>decimal-number</i>	<p>Correct: 54, 0.0.0.54, 0x01020304, 1.2.3.4</p> <p>Sample translations:</p> <p>54 becomes 0.0.0.54</p> <p>257 becomes 0.0.1.1 128.8 becomes 128.8.0.0 0x010203 becomes 0.1.2.3</p>

Synchronizing Routing Engines

If your router has two Routing Engines, you can manually direct one Routing Engine to synchronize its configuration with the other by issuing the **commit synchronize** command. The Routing Engine on which you execute this command (requesting Routing Engine) copies and loads its candidate configuration to the other (responding Routing Engine). Both Routing Engines then perform a syntax check on the candidate configuration file being committed. If no errors are found, the configuration is activated and becomes the current operational configuration on both Routing Engines.

The **commit synchronize** command does not work if the responding Routing Engine has uncommitted configuration changes. However, you can enforce commit synchronization on the Routing Engines by using the **force** option. When you issue the **commit synchronize** command with the **force** option from one Routing Engine, the configuration sessions on the other Routing Engine will be terminated and its

configuration synchronized with that on the Routing Engine from which you issued the command.



NOTE: We recommend that you use the **force** option only if you are unable to resolve the issues that caused the **commit synchronize** command to fail.

For example, if you are logged in to **re1** (requesting Routing Engine) and you want **re0** (responding Routing Engine) to have the same configuration as **re1**, issue the **commit synchronize** command on **re1**. **re1** copies and loads its candidate configuration to **re0**. Both Routing Engines then perform a syntax check on the candidate configuration file being committed. If no errors are found, **re1**'s candidate configuration is activated and becomes the current operational configuration on both Routing Engines.



NOTE: When you issue the **commit synchronize** command, you must use the groups **re0** and **re1**. For information about how to use the **apply-groups** statement, see “Applying a JUNOS Configuration Group” on page 148.

The responding Routing Engine must be running JUNOS Release 5.0 or later.

For information about issuing the **commit synchronize** command on a routing matrix, see the *JUNOS System Basics Configuration Guide*.

To synchronize a Routing Engine's current operational configuration file with the other, log in to the Routing Engine from which you want to synchronize and issue the **commit synchronize** command:

```
[edit]
user@host# commit synchronize
commit complete
[edit]
user@host#
```



NOTE: You can also add the **commit synchronize** statement at the **[edit system]** hierarchy level so that a **commit** command automatically invokes a **commit synchronize** command by default. For more information, see the *JUNOS System Basics Configuration Guide*.

To enforce a **commit synchronize** on the Routing Engines, log in to the Routing Engine from which you want to synchronize and issue the **commit synchronize** command with **force** option:

```
[edit]
user@host# commit synchronize force
re0:
re1:
commit complete
```

```
re0:  
commit complete  
[edit]  
user@host#
```



NOTE: If you have nonstop routing enabled on your router, you must issue the `commit synchronize` command from the master Routing Engine after you make any changes to the configuration. If you issue this command on the backup Routing Engine, the JUNOS system software displays a warning and commits the configuration.

Chapter 7

Filtering Command Output

For commands that display output, such as the **show** commands, you can filter the output. This chapter provides information about the following topics:

- Using the Pipe (|) Symbol to Filter JUNOS Command Output on page 117
- Using Regular Expressions with the Pipe (|) Symbol to Filter JUNOS Command Output on page 118
- Pipe (|) Filter Functions in the JUNOS Command-Line Interface on page 119

Using the Pipe (|) Symbol to Filter JUNOS Command Output

The JUNOS Software enables you to filter command output by adding the | (*pipe*) symbol when you enter a command.

For example:

```
user@host> show rip neighbor ?
Possible completions:
<[Enter]>      Execute this command
<name>        Name of RIP neighbor
instance      Name of RIP instance
logical-system Name of logical system, or 'all'
|             Pipe through a command
```

The following example lists the filters that can be used with the pipe symbol:

```
user@host> show rip neighbor | ?
Possible completions:
count          Count occurrences
display        Show additional kinds of information
except         Show only text that does not match a pattern
find           Search for first occurrence of pattern
hold           Hold text without exiting the --More-- prompt
last           Display end of output only
match          Show only text that matches a pattern
no-more        Don't paginate output
request        Make system-level requests
resolve        Resolve IP addresses
save           Save output text to file
trim           Trim specified number of columns from start of line
```

For the **show configuration** command only, an additional compare filter is available:

```

user@host> show configuration | ?
Possible completions:
  compare          Compare configuration changes with prior version
  ...

```

You can enter any of the pipe filters in conjunction. For example:

```
user@host> command | match regular-expression | save filename
```

See “Pipe (|) Filter Functions in the JUNOS Command-Line Interface” on page 119 for a description of each type of filter.



NOTE: This section describes *only* the filters that can be used for operational mode command output. For information about filters that can be used in configuration mode, see the *JUNOS System Basics Configuration Guide*.

Using Regular Expressions with the Pipe (|) Symbol to Filter JUNOS Command Output

The except, find, and match filters used with the pipe symbol employ regular expressions to filter output. Juniper Networks uses the regular expressions as defined in POSIX 1003.2. (See Table 11 on page 118.) If the regular expression contains spaces, operators, or wildcard characters, enclose the expression in quotation marks.

Table 11: Common Regular Expression Operators in Operational Mode Commands

Operator	Function
	Indicates that a match can be one of the two terms on either side of the pipe.
^	Used at the beginning of an expression, denotes where a match should begin.
\$	Used at the end of an expression, denotes that a term must be matched exactly up to the point of the \$ character.
[]	Specifies a range of letters or digits to match. To separate the start and end of a range, use a hyphen (-).
()	Specifies a group of terms to match.

For example, if a command produces the following output:

```

1 2
2 2
3 2 1
4

```

a pipe filter of | match 2 displays the following output:

```
1 2
2 2
3 2 1
```

and a pipe filter of `| except 1` displays the following output:

```
2 2
4
```



NOTE: See the following sections for more examples of using regular expressions:

- “Ignoring Output That Does Not Match a Regular Expression” on page 121
- “Displaying Output from the First Match of a Regular Expression” on page 122
- “Displaying Output That Matches a Regular Expression” on page 123

Pipe (|) Filter Functions in the JUNOS Command-Line Interface

This topic describes the pipe (|) filter functions supported in the JUNOS command-line interface (CLI):

- Comparing Configurations on page 119
- Counting the Number of Lines of Output on page 121
- Displaying Output in XML Tag Format on page 121
- Ignoring Output That Does Not Match a Regular Expression on page 121
- Displaying Output from the First Match of a Regular Expression on page 122
- Retaining Output After the Last Screen on page 122
- Displaying Output Beginning with the Last Entries on page 122
- Displaying Output That Matches a Regular Expression on page 123
- Preventing Output from Being Paginated on page 123
- Sending Command Output to Other Users on page 124
- Resolving IP Addresses on page 124
- Saving Output to a File on page 124
- Trimming Output by Specifying the Starting Column on page 125

Comparing Configurations

The `compare` filter compares the candidate configuration with either the current committed configuration or a configuration file and displays the differences between the two configurations. To compare configurations, enter `compare` after the pipe symbol:

```
[edit]
```

```
user@host# show | compare [filename | rollback n]
```

filename is the full path to a configuration file.

n is the index into the list of previously committed configurations. The most recently saved configuration is 0. If you do not specify arguments, the candidate configuration is compared against the active configuration file (`/config/juniper.conf`).

The comparison output uses the following conventions:

- Statements that are only in the candidate configuration are prefixed with a plus sign (+).
- Statements that are only in the comparison file are prefixed with a minus sign (-).
- Statements that are unchanged are prefixed with a single blank space ().

For example:

```
user@host> show configuration system | compare rollback 9
[edit system]
+ host-name nutmeg;
+ backup-router 192.168.71.254;
- ports {
-     console log-out-on-disconnect;
- }
[edit system name-server]
+ 172.17.28.11;
  172.17.28.101 { ... }
[edit system name-server]
  172.17.28.101 { ... }
+ 172.17.28.100;
+ 172.17.28.10;
[edit system]
- scripts {
-     commit {
-         allow-transients;
-     }
- }
+ services {
+     ftp;
+     rlogin;
+     rsh;
+     telnet;
+ }
```

Starting with JUNOS Release 8.3, output from the `show | compare` command has been enhanced to more accurately reflect configuration changes. This includes more intelligent handling of order changes in lists. For example, consider names in a group that are reordered as follows:

```
groups {      groups {
group_xmp;    group_xmp;
group_cmp;    group_grp:
group_grp;    group_cmp;
}             }
```

In previous releases, output from the `show | compare` command looked like the following:

```
[edit groups]
- group_xmp;
- group_cmp;
- group_grp;
+ group_xmp;
+ group_grp;
+ group_cmp;
```

Now, output from the `show | compare` command looks like the following:

```
[edit groups]
group_xmp {...}
! group_grp {...}
```

Counting the Number of Lines of Output

To count the number of lines in the output from a command, enter `count` after the pipe symbol. For example:

```
user@host> show configuration | count
Count: 269 lines
```

Displaying Output in XML Tag Format

To display command output in XML tag format, enter `display xml` after the pipe symbol.

The following example displays the `show cli directory` command output as XML tags:

```
user@host> show cli directory | display xml
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/7.5I0/junos">
  <cli>
    <working-directory>/var/home/regress</working-directory>
  </cli>
  <cli>
    <banner></banner>
  </cli>
</rpc-reply>
```

Ignoring Output That Does Not Match a Regular Expression

To ignore text that matches a regular expression, specify the `except` command after the pipe symbol. If the regular expression contains any spaces, operators, or wildcard characters, enclose it in quotation marks. For information on common regular expression operators, see Table 11 on page 118.

The following example displays all users who are logged in to the router, except for the user `root`:

```

user@host> show system users | except root
      8:28PM up 1 day, 13:59, 2 users, load averages: 0.01, 0.01, 0.00
USER    TTY FROM                LOGIN@  IDLE WHAT
sheep    p0  baa.juniper.net    7:25PM    - cli

```

Displaying Output from the First Match of a Regular Expression

To display output starting with the first occurrence of text matching a regular expression, enter **find** after the pipe symbol. If the regular expression contains any spaces, operators, or wildcard characters, enclose it in quotation marks. For information on common regular expression operators, see Table 11 on page 118.

The following example displays the routes in the routing table starting at IP address 208.197.169.0:

```

user@host> show route | find 208.197.169.0
208.197.169.0/24    *[Static/5] 1d 13:22:11
                  > to 192.168.4.254 via so-3/0/0.0
224.0.0.5/32       *[OSPF/10] 1d 13:22:12, metric 1
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
47.0005.80ff.f800.0000.0108.0001.1921.6800.4015.00/160
                  *[Direct/0] 1d 13:22:12
                  > via lo0.0

```

The following example displays the first CCC entry in the forwarding table:

```

user@host> show route forwarding-table | find ccc
Routing table: ccc
MPLS:
Interface.Label    Type RtRef Nexthop          Type Index NhRef Netif
default            perm  0          Type Index NhRef Netif
0                  user  0          recv   5      2
1                  user  0          recv   5      2
32769              user  0          ucst   45     1 fe-0/0/0.534
fe-0/0/0. (CCC)    user  0          indr   44     2
                  10.0.16.2  Push 32768, Push

```

Retaining Output After the Last Screen

To not return immediately to the CLI prompt after viewing the last screen of output, enter **hold** after the pipe symbol. The following example prevents returning to the CLI prompt after you have viewed the last screen of output from the **show log log-file-1** command:

```

user@host> show log log-file-1 | hold

```

This filter is useful when you want to scroll or search through output.

Displaying Output Beginning with the Last Entries

To display text starting from the end of the output, enter **last <lines>** after the pipe symbol.

The following example displays the last entries in **log-file-1** file:

```
user@host> show log log-file-1 | last
```

This filter is useful for viewing log files in which the end of the file contains the most recent entries.



NOTE: When the number of lines requested is less than the number of lines that the screen length setting permits you to display, JUNOS returns as many lines as permitted by the screen length setting. That is, if your screen length is set to 20 lines and you have requested only the last 10 lines, JUNOS returns the last 19 lines instead of the last 10 lines.

Displaying Output That Matches a Regular Expression

To display output that matches a regular expression, enter **match *regular-expression*** after the pipe symbol. If the regular expression contains any spaces, operators, or wildcard characters, enclose it in quotation marks. For information on common regular expression operators, see Table 11 on page 118.

The following example matches all the Asynchronous Transfer Mode (ATM) interfaces in the configuration:

```
user@host> show configuration | match at-
at-2/1/0 {
at-2/1/1 {
at-2/2/0 {
at-5/2/0 {
at-5/3/0 {
```

Preventing Output from Being Paginated

By default, if output is longer than the length of the terminal screen, you are provided with a **—(more)—** message to display the remaining output. To display the remaining output, press the Spacebar.

To prevent the output from being paginated, enter **no-more** after the pipe symbol.

The following example displays output from the **show configuration** command all at once:

```
user@host> show configuration | no-more
```

This feature is useful, for example, if you want to copy the entire output and paste it into an e-mail.

Sending Command Output to Other Users

To display command output on the terminal of a specific user logged in to your router, or on the terminals of all users logged in to your router, enter **request message** (all | user *account@terminal*) after the pipe symbol.

If you are troubleshooting your router and, for example, talking with a customer service representative on the phone, you can use the **request message** command to send your representative the command output you are currently viewing on your terminal.

The following example sends the output from the **show interfaces** command you enter on your terminal to the terminal of the user **root@tty1**:

```
user@host> show interfaces | request message user root@tty1
```

The user **root@tty1** sees the following output appear on the terminal screen:

```
Message from user@host on /dev/tty0 at 10:32 PST...
Physical interface: dsc, Enabled, Physical link is Up
  Interface index: 5, SNMP ifIndex: 5
  Type: Software-Pseudo, MTU: Unlimited...
```

Resolving IP Addresses

If the output of a command displays an unresolved IP address, you can enter | **resolve** after the command to display the name associated with the IP address. The **resolve** filter enables the system to perform a reverse DNS lookup of the IP address. If DNS is not enabled, the lookup fails and no substitution is performed.

To perform a reverse DNS lookup of an unresolved IP address, enter **resolve <full-names>** after the pipe symbol. If you do not specify the **full-names** option, the name is truncated to fit whatever field width limitations apply to the IP address.

The following example performs a DNS lookup on any unresolved IP addresses in the output from the **show ospf neighbors** command:

```
user@host> show ospf neighbors | resolve
```

Saving Output to a File

When command output is lengthy, when you need to store or analyze the output, or when you need to send the output in an e-mail or by FTP, you can save the output to a file. By default, the file is placed in your home directory on the router.

To save command output to a file, enter **save filename** after the pipe symbol.

The following example saves the output from the **request support information** command to a file named **my-support-info.txt**:

```
user@host> request support information | save my-support-info.txt
```


Wrote 1143 lines of output to 'my-support-info.txt'
user@host>

Trimming Output by Specifying the Starting Column

Output appears on the terminal screen in terms of rows and columns. The first alphanumeric character starting at the left of the screen is in column 1, the second character is in column 2, and so on. To display output starting from a specific column (thus trimming the leftmost portion of the output), enter **trim columns** after the pipe symbol. The trim filter is useful for trimming the date and time from the beginning of system log messages

The following example displays output from the **show system storage** command, filtering out the first 10 columns:

```
user@host> show system storage | trim 11
```


Chapter 8

Controlling the CLI Environment

This chapter contains the following sections:

- Controlling the JUNOS CLI Environment on page 127
- Example: Controlling the CLI Environment on page 129
- Setting the JUNOS CLI Screen Length and Width on page 130

Controlling the JUNOS CLI Environment

In operational mode, you can control the JUNOS command-line interface (CLI) environment. For example, you can specify the number lines that are displayed on the screen or your terminal type. The following output lists the options that you can use to control the CLI environment:

```
user@host>set cli ?
Possible completions:
complete-on-space  Set whether typing space completes current word
directory          Set working directory
idle-timeout       Set maximum idle time before login session ends
logical-system     Set default logical system
prompt            Set CLI command prompt string
restart-on-upgrade Set whether CLI prompts to restart after software upgrade

screen-length      Set number of lines on screen
screen-width       Set number of characters on a line
terminal          Set terminal type
timestamp          Timestamp CLI output
```



NOTE: When you use SSH to log in to the router or log in from the console when its terminal type is already configured (as described in the *JUNOS System Basics Configuration Guide*), your terminal type, screen length, and screen width are already set.

This chapter discusses the following topics:

- Setting the Terminal Type on page 128
- Setting the CLI Prompt on page 128
- Setting the CLI Directory on page 128
- Setting the CLI Timestamp on page 128

- Setting the Idle Timeout on page 128
- Setting the CLI to Prompt After a Software Upgrade on page 129
- Setting Command Completion on page 129
- Displaying CLI Settings on page 129

Setting the Terminal Type

To set the terminal type, use the **set cli terminal** command:

```
user@host> set cli terminal terminal-type
```

The terminal type can be one of the following: *ansi*, *vt100*, *small-xterm*, or *xterm*.

Setting the CLI Prompt

The default CLI prompt is *user@host>*. To change this prompt, use the **set cli prompt** command. If the prompt string contains spaces, enclose the string in quotation marks (" ").

```
user@host> set cli prompt string
```

Setting the CLI Directory

To set the current working directory, use the **set cli directory** command:

```
user@host> set cli directory directory
```

directory is the pathname of working directory.

Setting the CLI Timestamp

By default, CLI output does not include a timestamp. To include a timestamp in CLI output, use the **set cli timestamp** command:

```
user@host> set cli timestamp [format time-date-format | disable]
```

If you do not specify a timestamp format, the default format is *Mmm dd hh:mm:ss* (for example, Feb 08 17:20:49). Enclose the format in single quotation marks (').

Setting the Idle Timeout

By default, an individual CLI session never times out after extended times, unless the **idle-timeout** statement has been included in the user's login class configuration. To set the maximum time an individual session can be idle before the user is logged off the router, use the **set cli idle-timeout** command:

```
user@host> set cli idle-timeout timeout
```

timeout can be 0 through 100,000 minutes. Setting *timeout* to 0 disables the timeout.

Setting the CLI to Prompt After a Software Upgrade

By default, the CLI prompts you to restart after a software upgrade. To disable the prompt for an individual session, use the `set cli restart-on-upgrade off` command:

```
user@host> set cli restart-on-upgrade off
```

To reenable the prompt, use the `set cli restart-on-upgrade on` command:

```
user@host> set cli restart-on-upgrade on
```

Setting Command Completion

By default, you can press Tab or the Spacebar to have the CLI complete a command.

To have the CLI allow only a tab to complete a command, use the `set cli complete-on-space off` command:

```
user@host> set cli complete-on-space off
Disabling complete-on-space
user@host>
```

To reenable the use of both spaces and tabs for command completion, use the `set cli complete-on-space on` command:

```
user@host> set cli complete-on-space on
Enabling complete-on-space
user@host>
```

Displaying CLI Settings

To display the current CLI settings, use the `show cli` command:

```
user@host> show cli
CLI screen length set to 24
CLI screen width set to 80
CLI complete-on-space set to on
```

Example: Controlling the CLI Environment

The following example shows you how to change the default CLI environment:

```
user@host> set cli screen-length 66
Screen length set to 66
user@host> set cli screen-width 40
Screen width set to 40
user@host> set cli prompt "router1-san-jose > "
router1-san-jose > show cli
CLI complete-on-space set to on
CLI idle-timeout disabled
CLI restart-on-upgrade set to on
CLI screen length set to 66
```

```
CLI screen width set to 40
CLI terminal is 'xterm'
router1-san-jose >
```

Setting the JUNOS CLI Screen Length and Width

You can set the JUNOS command-line interface (CLI) screen length and width according to your specific requirements. This topic contains the following sections:

- Setting the Screen Length on page 130
- Setting the Screen Width on page 130
- Understanding the Screen Length and Width Settings on page 130

Setting the Screen Length

The default CLI screen length is 24 lines. To change the length, use the **set cli screen-length** command:

```
user@host> set cli screen-length length
```

Setting the screen length to 0 lines disables the display of output one screen at a time. Disabling this UNIX **more**-type interface can be useful when you are issuing CLI commands from scripts.

Setting the Screen Width

The default CLI screen width is 80 characters. To change the width, use the **set cli screen-width** command:

```
user@host> set cli screen-width width
```

Understanding the Screen Length and Width Settings

The **cli screen-length** and **cli screen-width** settings in combination with each other and the size of the telnet or console window determine the extent of output displayed before each **–more–** prompt appears.

The following examples explain how the **cli screen-length** and **cli screen-width** values determine the appearance of the output:

- When the CLI screen width is set to the default value (80 characters) and the **cli screen-length** to 10 lines, the **–more–** prompt appears on the tenth line of the output.
- When the CLI screen width is set to 20 characters and the **cli screen-length** to 6 lines in a telnet or console window that is wide enough to contain 40 characters, the **–more–** prompt appears on the fourth line of the output. Here each one of the first two lines has more than 20 characters and is counted as two lines. The third line contains the fifth line of output, and the fourth line contains the **–more–** prompt, which has to appear in the sixth line as per the setting.



NOTE: If you have inadvertently set the CLI screen width to a lower value that does not allow you to see the commands that you are typing, reset the CLI screen width with a higher value by entering the `set cli screen-width` command.



TIP: If you are not able to see the command that you are entering, type the command in a text editor and copy it at the command prompt.

Part 3

Advanced Features

- Using Shortcuts, Wildcards, and Regular Expressions on page 135
- Configuration Groups on page 145
- Summary of Configuration Group Statements on page 187

Chapter 9

Using Shortcuts, Wildcards, and Regular Expressions

This chapter provides information on how to use keyboard shortcuts, wildcards, and other advanced techniques to save time when entering commands and configuration statements.

Topics include:

- Using Keyboard Sequences to Move Around and Edit the JUNOS CLI on page 135
- Using Wildcard Characters in Interface Names on page 137
- Using Global Replace in a JUNOS Configuration on page 137
- Common Regular Expressions to Use with the replace Command on page 138
- Using Global Replace in a JUNOS Configuration—Using the \n Back Reference on page 139
- Using Global Replace in a JUNOS Configuration—Replacing an Interface Name on page 140
- Using Global Replace in a JUNOS Configuration—Using the upto Option on page 141
- Using Regular Expressions to Delete Related Items from a JUNOS Configuration on page 142

Using Keyboard Sequences to Move Around and Edit the JUNOS CLI

In the JUNOS command-line interface (CLI), you can use keyboard sequences to move around on a command line and edit the command line. You can also use keyboard sequences to scroll through a list of recently executed commands. Table 12 on page 136 lists some of the CLI keyboard sequences. They are the same as those used in Emacs.

Table 12: CLI Keyboard Sequences

Category	Action	Keyboard Sequence
Move the Cursor	Move the cursor back one character.	Ctrl + b
	Move the cursor back one word.	Esc + b or Alt + b
	Move the cursor forward one character.	Ctrl + f
	Move the cursor forward one word.	Esc + f or Alt + f
	Move the cursor to the beginning of the command line.	Ctrl + a
	Move the cursor to the end of the command line.	Ctrl + e
Delete Characters	Delete the character before the cursor.	Ctrl + h, Delete, or Backspace
	Delete the character at the cursor.	Ctrl + d
	Delete all characters from the cursor to the end of the command line.	Ctrl + k
	Delete all characters on the command line.	Ctrl + u or Ctrl + x
	Delete the word before the cursor.	Ctrl + w, Esc + Backspace, or Alt + Backspace
	Delete the word after the cursor.	Esc + d or Alt + d
Insert Recently Deleted Text	Insert the most recently deleted text at the cursor.	Ctrl + y
Redraw the Screen	Redraw the current line.	Ctrl + l
Display Previous Command Lines	Scroll backward through the list of recently executed commands.	Ctrl + p
	Scroll forward through the list of recently executed commands.	Ctrl + n
	Search the CLI history in reverse order for lines matching the search string.	Ctrl + r
	Search the CLI history by typing some text at the prompt, followed by the keyboard sequence. The CLI attempts to expand the text into the most recent word in the history for which the text is a prefix.	Esc + /

Table 12: CLI Keyboard Sequences (continued)

Category	Action	Keyboard Sequence
Display Previous Command Words	Scroll backward through the list of recently entered words in a command line.	Esc + . or Alt + .
Repeat Keyboard Sequences	Specify the number of times to execute a keyboard sequence. <i>number</i> can be from 1 through 9 and <i>sequence</i> is the keyboard sequence that you want to execute.	Esc + <i>number sequence</i> or Alt + <i>number sequence</i>

Using Wildcard Characters in Interface Names

You can use wildcard characters in the JUNOS Software operational commands to specify groups of interface names without having to type each name individually. Table 13 on page 137 lists the available wildcard characters. You must enclose all wildcard characters except the asterisk (*) in quotation marks (“ ”).

Table 13: Wildcard Characters for Specifying Interface Names

Wildcard Character	Description
* (asterisk)	Match any string of characters in that position in the interface name. For example, so* matches all SONET/SDH interfaces.
"[character<character...>]"	Match one or more individual characters in that position in the interface name. For example, so-"[03]" matches all SONET/SDH interfaces in slots 0 and 3.
"[!character<character...>]"	Match all characters except the ones included in the brackets. For example, so-"[!03]" matches all SONET/SDH interfaces except those in slots 0 and 3.
"[character1-character2]"	Match a range of characters. For example, so-"[0-3]" matches all SONET/SDH interfaces in slots 0, 1, 2, and 3.
"[!character1-character2]"	Match all characters that are not in the specified range of characters. For example, so-"[!0-3]" matches all SONET/SDH interfaces in slots 4, 5, 6, and 7.

Using Global Replace in a JUNOS Configuraton

You can make global changes to variables and identifiers in a JUNOS configuration by using the **replace** configuration mode command. This command replaces a pattern in a configuration with another pattern. For example, you can use this command to find and replace all occurrences of an interface name when a PIC is moved to another slot in the router.

```
user@host# replace pattern pattern1 with pattern2 <upto n>
```

pattern pattern1 is a text string or regular expression that defines the identifiers and values you want to replace in the configuration.

pattern2 is a text string or regular expression that replaces the identifiers and values located with *pattern1*.

Juniper Networks uses standard UNIX-style regular expression syntax (as defined in POSIX 1003.2). If the regular expression contains spaces, operators, or wildcard characters, enclose the expression in quotation marks. Greedy qualifiers (match as much as possible) are supported. Lazy qualifiers (match as little as possible) are not.

The **upto** *n* option specifies the number of objects replaced. The value of *n* controls the total number of objects that are replaced in the configuration (not the total number of times the pattern occurs). Objects at the same hierarchy level (siblings) are replaced first. Multiple occurrences of a pattern within a given object are considered a single replacement. For example, if a configuration contains a **010101** text string, the command

```
replace pattern 01 with pattern 02 upto 2
```

replaces **010101** with **020202** (instead of **020201**). Replacement of **010101** with **020202** is considered a single replacement (*n* = 1), not three separate replacements (*n* = 3).

If you do not specify an **upto** option, all identifiers and values in the configuration that match *pattern1* are replaced.

The **replace** command is available in configuration mode at any hierarchy level. All matches are case-sensitive.

“Common Regular Expressions to Use with the replace Command” on page 138 shows some common regular expressions you can use with the **replace** command.

Common Regular Expressions to Use with the replace Command

Table 14: Common Regular Expressions to Use with the replace Command

Operator	Function
	Indicates that a match can be one of the two terms on either side of the pipe.
^	Used at the beginning of an expression, denotes where a match should begin.
\$	Used at the end of an expression, denotes that a term must be matched exactly up to the point of the \$ character.
[]	Specifies a range of letters or digits to match. To separate the start and end of a range, use a hyphen (-).

Table 14: Common Regular Expressions to Use with the replace Command *(continued)*

Operator	Function
()	Specifies a group of terms to match. Stored as numbered variables. Use for back references as \1 \2 \9.
*	0 or more terms.
+	One or more terms.
.	Any character except for a space (" ").
\	A backslash escapes special characters to suppress their special meaning. For example, \. matches . (period symbol).
\n	Back reference. Matches the <i>n</i> th group.
&	Back reference. Matches the entire match.

Table 15 on page 139 lists some replacement examples:

Table 15: Replacement Examples

Command	Result
replace pattern myrouter with router1	Match: myrouter Result: router1
replace pattern " 192\.\168\.(.*)/24" with " 10.2.\1/28"	Match: 192.168.3.4/24 Result: 10.2.3.4/28
replace pattern " 1.\1" with " abc&def"	Match: 1.1 Result: abc1.1def
replace pattern 1.1 with " abc\&def"	Match: 1#1 Result: abc&def

Using Global Replace in a JUNOS Configuration—Using the \n Back Reference

The following example shows how you can use the \n back reference to replace a pattern:

```
[edit]
user@host# show interfaces
xe-0/0/0 {
  unit 0;
}
```

```

fe-3/0/1 {
  vlan-tagging;
  unit 0 {
    description "inet6 configuration. IP: 2000::c0a8:1bf5";
    vlan-id 100;
    family inet {
      address 17.10.1.1/24;
    }
    family inet6 {
      address 2000::c0a8:1bf5/3;
    }
  }
}
[edit]
user@host# replace pattern "(.*)1bf5" with "\11bf5"
[edit]
user@host# show interfaces
xe-0/0/0 {
  unit 0;
}
fe-3/0/1 {
  vlan-tagging;
  unit 0 {
    description "inet6 configuration. IP: 2000::c0a8:1bf5";
    vlan-id 100;
    family inet {
      address 17.10.1.1/24;
    }
    family inet6 {
      address 2000::c0a8:1bf5/3;
    }
  }
}

```

The pattern 2000::c0a8::1bf5 is replaced with 2000::c0a8:1bf5.

Using Global Replace in a JUNOS Configuration—Replacing an Interface Name

The following example shows how you can replace an interface name in a configuration:

```

[edit]
user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
[edit]
user@host# replace so-0/0/0 with so-1/1/0
[edit]

```



```

user@host# show
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-1/1/0 {
        hello-interval 5;
      }
    }
  }
}

```

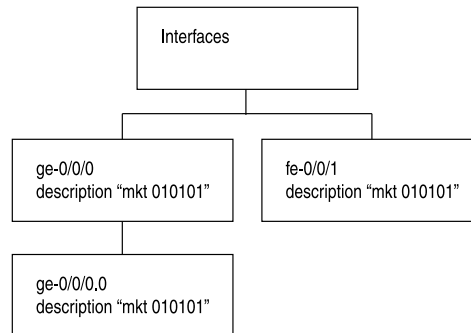
Using Global Replace in a JUNOS Configuration—Using the upto Option

The following example shows how you can use the `upto` option to perform replacements in a JUNOS configuration:

Consider the hierarchy shown in Figure 16 on page 141. The text string `010101` appears in three places (description sections of `ge-0/0/0`, `ge-0/0/0.0`, and `fe-0/0/1`). These three instances are three objects.

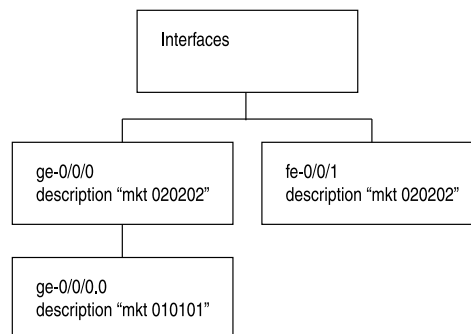
Figure 16: Replacement by Object

Current Configuration:



user@host > **replace pattern 01 with pattern 02 upto 2**

Resulting Configuration:



```

user@host# show interfaces
ge-0/0/0 {

```

g017228

```

description "mkt 010101"; #First instance in the hierarchy
unit 0 {
    description "mkt 010101"; #Third instance in the hierarchy (child of the 1st
    instance)
}
}
fe-0/0/1 {
    description "mkt 010101"; #second instance in the hierarchy (sibling of the first
    instance)
    unit 0 {
        family inet {
            address 200.200.20.2/24;
        }
    }
}
[edit]
user@host# replace pattern 01 with 02 upto 2
[edit]
user@host# commit
commit complete

```

An **upto 2** option in the **replace** command converts **01** to **02** for two object instances. The objects under the main interfaces **ge-0/0/0** and **fe-0/0/1** will be replaced first (since these are siblings in the hierarchy level). Because of the **upto 2** restriction, the **replace** command replaces patterns in the first and second instance in the hierarchy (siblings), but not the third instance (child of the first instance).

```

[edit]
user@host# show interfaces
ge-0/0/0 {
    description "mkt 020202"; #First instance in the hierarchy
    unit 0 {
        description "mkt 010101"; #Third instance in the hierarchy (child of the first
        instance)
    }
}
fe-0/0/1 {
    description "mkt 020202"; #second instance in the hierarchy (sibling of the first
    instance)
    unit 0 {
        family inet {
            address 200.200.20.2/24;
        }
    }
}

```

Using Regular Expressions to Delete Related Items from a JUNOS Configuration

The JUNOS command-line interface (CLI) enables you to delete related configuration items simultaneously, such as channelized interfaces or static routes, by using a single command and regular expressions. Deleting a statement or an identifier

effectively “unconfigures” the functionality associated with that statement or identifier, returning that functionality to its default condition.

You can only delete certain parts of the configuration where you normally put multiple items, for example, interfaces. However, you cannot delete “groups” of different items; for example:

```
user@host# show system services
ftp;
rlogin;
rsh;
ssh {
    root-login allow;
}
telnet;
[edit]
user@host# wildcard delete system services *
syntax error.
```

When you delete a statement, the statement and all its subordinate statements and identifiers are removed from the configuration.

To delete related configuration items, issue the **wildcard** configuration mode command with the **delete** option and specify the statement path, the items to be summarized with a regular expression, and the regular expression.

```
user@host# wildcard delete <statement-path> <identifier> <regular-expression>
```



NOTE: When you use the **wildcard** command to delete related configuration items, the regular expression must be the final statement.

If JUNOS Software matches more than eight related items, the CLI displays only the first eight items.

Deleting Interfaces from the Configuration

Delete multiple T1 interfaces in the range from t1-0/0/0:0 through t1-0/0/0:23:

```
user@host# wildcard delete interfaces t1-0/0/0:. *
matched: t1-0/0/0:0
matched: t1-0/0/0:1
matched: t1-0/0/0:2
Delete 3 objects? [yes,no] (no) no
```

**Deleting Routes from
the Configuration**

Delete static routes in the range from 172.0.0.0 to 172.255.0.0:

```
user@host# wildcard delete routing-options static route 172.*  
matched: 172.16.0.0/12  
matched: 172.16.14.0/24  
matched: 172.16.100.0/24  
matched: 172.16.128.0/19  
matched: 172.16.160.0/24  
matched: 172.17.12.0/23  
matched: 172.17.24.0/23  
matched: 172.17.28.0/23  
...  
Delete 13 objects? [yes,no] (no)
```

Chapter 10

Configuration Groups

This chapter contains the following topics:

- Understanding the JUNOS Configuration Groups on page 146
- Creating a JUNOS Configuration Group on page 147
- Applying a JUNOS Configuration Group on page 148
- Example: Configuring and Applying JUNOS Configuration Groups on page 149
- Example: Creating and Applying Configuration Groups on a TX Matrix Router on page 150
- Disabling Inheritance of a JUNOS Configuration Group on page 151
- Using Wildcards with Configuration Groups on page 152
- Configuring Sets of Statements with Configuration Groups on page 154
- Configuring Interfaces Using JUNOS Configuration Groups on page 155
- Configuring a Consistent Management IP Address on page 157
- Configuring Peer Entities on page 158
- Establishing Regional Configurations on page 160
- Selecting Wildcard Names on page 161
- Using JUNOS Defaults Groups on page 162
- Example: Referencing the Preset Statement on page 163
- Example: Viewing Default Statements That Have Been Applied to the Configuration on page 164
- Inheritance Model on page 164
- Creating and Applying Configuration Groups on page 165
- Configuration Groups Configuration Guidelines on page 165
- Examples: Configuration Groups on page 174
- Using JUNOS Defaults Groups on page 183

Understanding the JUNOS Configuration Groups

This topic provides you an overview of the configuration groups feature and the inheritance model in JUNOS Software, and contains the following sections:

- Configuration Groups Overview on page 146
- Inheritance Model on page 146
- Configuring Configuration Groups on page 147

Configuration Groups Overview

The configuration groups feature in JUNOS Software enables you to create a group containing configuration statements and to direct the inheritance of that group's statements in the rest of the configuration. The same group can be applied to different sections of the configuration, and different sections of one group's configuration statements can be inherited in different places in the configuration.

Configuration groups enable you to create smaller, more logically constructed configuration files, making it easier to configure and maintain JUNOS Software. For example, you can group statements that are repeated in many places in the configuration, such as when configuring interfaces, and thereby limit updates to just the group.

You can also use wildcards in a configuration group to allow configuration data to be inherited by any object that matches a wildcard expression.

The configuration group mechanism is separate from the grouping mechanisms used elsewhere in the configuration, such as BGP groups. Configuration groups provide a generic mechanism that can be used throughout the configuration but that are known only to JUNOS Software command-line interface (CLI). The individual software processes that perform the actions directed by the configuration receive the expanded form of the configuration; they have no knowledge of configuration groups.

Inheritance Model

Configuration groups use true inheritance, which involves a dynamic, ongoing relationship between the source of the configuration data and the target of that data. Data values changed in the configuration group are automatically inherited by the target. The target need not contain the inherited information, although the inherited values can be overridden in the target without affecting the source from which they were inherited.

This inheritance model allows you to see only the instance-specific information without seeing the inherited details. A command pipe in configuration mode allows you to display the inherited data.

Configuring Configuration Groups

For areas of your configuration to inherit configuration statements, you must first put the statements into a configuration group and then apply that group to the levels in the configuration hierarchy that require the statements.

To configure configuration groups and inheritance, you can include the **groups** statement at the **[edit]** hierarchy level:

```
[edit]
groups {
  group-name {
    configuration-data;
  }
}
```

Include the **apply-groups [group-names]** statement anywhere in the configuration that the configuration statements contained in a configuration group are needed.

Creating a JUNOS Configuration Group

To create a configuration group, include the **groups** statement at the **[edit]** hierarchy level:

```
[edit]
groups {
  group-name {
    configuration-data;
  }
  lccn-re0 {
    configuration-data;
  }
  lccn-re1 {
    configuration-data;
  }
}
```

group-name is the name of a configuration group. You can configure more than one configuration group by specifying multiple **group-name** statements. However, you cannot use the prefix **junos-** in a group name because it is reserved for use by JUNOS Software. Similarly, the configuration group **juniper-ais** is reserved exclusively for Juniper Advanced Insight Solutions (AIS)-related configuration. For more information on the **juniper-ais** configuration group, see the Juniper Networks *Advanced Insight Solutions Guide*.

One reason for the naming restriction is a configuration group called **junos-defaults**. This preset configuration group is applied to the configuration automatically. You cannot modify or remove the **junos-defaults** configuration group. For more information about the JUNOS default configuration group, see “Using JUNOS Defaults Groups” on page 183.

On routers that support multiple Routing Engines, you can also specify two special group names:

- **re0**—Configuration statements applied to the Routing Engine in slot 0.
- **re1**—Configuration statements applied to the Routing Engine in slot 1.

The configuration specified in group **re0** is only applied if the current Routing Engine is in slot 0; likewise, the configuration specified in group **re1** is only applied if the current Routing Engine is in slot 1. Therefore, both Routing Engines can use the same configuration file, each using only the configuration statements that apply to it. Each **re0** or **re1** group contains at a minimum the configuration for the hostname and the management interface (**fxp0**). If each Routing Engine uses a different management interface, the group also should contain the configuration for the backup router and static routes.

In addition, the TX Matrix router supports group names for the Routing Engines in each T640 router attached to the routing matrix. Providing special group names for all Routing Engines in the routing matrix allows you to configure the individual Routing Engines in each T640 router differently. Parameters that are not configured at the **[edit groups]** hierarchy level apply to all Routing Engines in the routing matrix.

configuration-data contains the configuration statements applied elsewhere in the configuration with the **apply-groups** statement. To have a configuration inherit the statements in a configuration group, include the **apply-groups** statement. For information about the **apply-groups** statement, see “Applying a Configuration Group” on page 167.

The group names for Routing Engines on the TX Matrix router have the following formats:

- **lccn-re0**—Configuration statements applied to the Routing Engine in slot 0 in a specified T640 router.
- **lccn-re1**—Configuration statements applied to the Routing Engine in slot 1 in a specified T640 router.

n identifies the T640 router and can be from 0 through 3. For example, to configure Routing Engine 1 properties for **lcc3**, you include statements at the

[edit groups lcc3-re1] hierarchy level. For information about the TX Matrix router and routing matrix, see the *JUNOS System Basics Configuration Guide*.

Applying a JUNOS Configuration Group

To have a JUNOS configuration inherit the statements from a configuration group, include the **apply-groups** statement:

```
apply-groups [ group-names ];
```

If you specify more than one group name, list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.

For routers that support multiple Routing Engines, you can specify **re0** and **re1** group names. The configuration specified in group **re0** is only applied if the current Routing Engine is in slot 0; likewise, the configuration specified in group **re1** is only applied if the current Routing Engine is in slot 1. Therefore, both Routing Engines can use the same configuration file, each using only the configuration statements that apply to it. Each **re0** or **re1** group contains at a minimum the configuration for the hostname and the management interface (**fxp0**). If each Routing Engine uses a different management interface, the group also should contain the configuration for the backup router and static routes.

You can include only one **apply-groups** statement at each specific level of the configuration hierarchy. The **apply-groups** statement at a specific hierarchy level lists the configuration groups to be added to the containing statement's list of configuration groups.

Values specified at the specific hierarchy level override values inherited from the configuration group.

Groups listed in nested **apply-groups** statements take priority over groups in outer statements. In the following example, the BGP neighbor **10.0.0.1** inherits configuration data from group **one** first, then from groups **two** and **three**. Configuration data in group **one** overrides data in any other group. Data from group **ten** is used only if a statement is not contained in any other group.

```

apply-groups [ eight nine ten ];
protocols {
  apply-groups seven;
  bgp {
    apply-groups [ five six ];
    group some-bgp-group {
      apply-groups four;
      neighbor 10.0.0.1 {
        apply-groups [ one two three ];
      }
    }
  }
}

```

Example: Configuring and Applying JUNOS Configuration Groups

In this example, the SNMP configuration is divided between the group **basic** and the normal configuration hierarchy.

There are a number of advantages to placing the system-specific configuration (SNMP contact) into a configuration group and thus separating it from the normal configuration hierarchy—the user can replace (using the **load replace** command) either section without discarding data from the other.

In addition, setting a contact for a specific box is now possible because the group data would be hidden by the router-specific data.

```

[edit]
groups {

```

```

basic {# User-defined group name
snmp {# This group contains some SNMP data
contact "My Engineering Group";
community BasicAccess {
    authorization read-only;
}
}
apply-groups basic;# Enable inheritance from group "basic"
snmp {# Some normal (non-group) configuration
location "West of Nowhere";

```

This configuration is equivalent to the following:

```

[edit]
snmp {
    location "West of Nowhere";
    contact "My Engineering Group";
    community BasicAccess {
        authorization read-only;
    }
}

```

For information about how to disable inheritance of a configuration group, see “Disabling Inheritance of a Configuration Group” on page 169.

Example: Creating and Applying Configuration Groups on a TX Matrix Router

```

[edit]
groups {
    re0 { # Routing Engine 0 on TX Matrix router
        system {
            host-name <host-name>;
            backup-router <ip-address>;
        }
        interfaces {
            fxp0 {
                unit 0 {
                    family inet {
                        address <ip-address>;
                    }
                }
            }
        }
    }
    re1 { # Routing Engine 1 on TX Matrix router
        system {
            host-name <host-name>;
            backup-router <ip-address>;
        }
        interfaces {
            fxp0 {
                unit 0 {
                    family inet {
                        address <ip-address>;
                    }
                }
            }
        }
    }
}

```

```

    }
  }
}
lcc0-re0 { # Routing Engine 0 on T640 router numbered 0
system {
  host-name <host-name>;
  backup-router <ip-address>;
}
interfaces {
  fxp0 {
    unit 0 {
      family inet {
        address <ip-address>;
      }
    }
  }
}
lcc0-re1 { # Routing Engine 1 on T640 router numbered 0
system {
  host-name <host-name>;
  backup-router <ip-address>;
}
interfaces {
  fxp0 {
    unit 0 {
      family inet {
        address <ip-address>;
      }
    }
  }
}
}
apply-groups [ re0 re1 lcc0-re0 lcc0-re1 ];
}

```

Disabling Inheritance of a JUNOS Configuration Group

To disable inheritance of a configuration group at any level except the top level of the hierarchy, include the `apply-groups-except` statement:

```
apply-groups-except [ group-names ];
```

This statement is useful when you use the `apply-group` statement at a specific hierarchy level but also want to override the values inherited from the configuration group for a specific parameter.

Example: Disabling Inheritance on Interface so-1/1/0

In the following example, the `apply-groups` statement is applied globally at the interfaces level. The `apply-groups-except` statement is also applied at interface `so-1/1/0` so that it uses the default values `hold-time` and `link-mode`.

```

[edit]
groups { # "groups" is a top-level statement
global { # User-defined group name
interfaces {
  <*> {
    hold-time down 640;
    link-mode full-duplex;
  }
}
}

```

```

    }
  }
  apply-groups global;
  interfaces {
    so-1/1/0 {
      apply-groups-except global; # Disables inheritance from group "global"
      # so-1/1/0 uses default value for "hold-time"
    }
    # and "link-mode"
  }
}

```

For information about applying a configuration group, see “Applying a Configuration Group” on page 167.

Using Wildcards with Configuration Groups

You can use wildcards to identify names and allow one statement to provide data for a variety of statements. For example, grouping the configuration of the `sonet-options` statement over all SONET/SDH interfaces or the dead interval for OSPF over all Asynchronous Transfer Mode (ATM) interfaces simplifies configuration files and eases their maintenance.

Using wildcards in normal configuration data is done in a style that is consistent with that used with traditional UNIX shell wildcards. In this style, you can use the following metacharacters:

- Asterisk (`*`)—Matches any string of characters.
- Question mark (`?`)—Matches any single character.
- Open bracket (`[`)—Introduces a character class.
- Close bracket (`]`)—Indicates the end of a character class. If the close bracket is missing, the open bracket matches a `[` rather than introduce a character class.
- A character class matches any of the characters between the square brackets. Character classes must be enclosed in quotation marks (`" "`).
- Hyphen (`-`)—Specifies a range of characters.
- Exclamation point (`!`)—The character class can be complemented by making an exclamation point the first character of the character class. To include a `]` in a character class, make it the first character listed (after the `!`, if any). To include a minus sign, make it the first or last character listed.

Wildcarding in configuration groups follows the same rules, but the wildcard pattern must be enclosed in angle brackets (`<pattern>`) to differentiate it from other wildcarding in the configuration file. For example:

```

[edit]
groups {
  sonet-default {
    interfaces {
      <so-*> {
        sonet-options {

```

```

        payload-scrambler;
        rfc-2615;
    }
}
}
}
}

```

Wildcard expressions match (and provide configuration data for) existing statements in the configuration that match their expression only. In the example above, the expression `<so-*>` passes its `sonet-options` statement to any interface that matches the expression `so-*`.

Angle brackets allow you to pass normal wildcarding through without modification. In all matching within the configuration, whether it is done with or without wildcards, the first item encountered in the configuration that matches is used. In the following example, data from the wildcarded BGP groups is inherited in the order in which the groups are listed. The preference value from `<*a*>` overrides the preference in `<*b*>`, just as the `p` value from `<*c*>` overrides the one from `<*d*>`. Data values from any of these groups override the data values from `abcd`.

```

[edit]
user@host# show
groups {
  one {
    protocols {
      bgp {
        group <*a*> {
          preference 1;
        }
        group <*b*> {
          preference 2;
        }
        group <*c*> {
          out-delay 3;
        }
        group <*d*> {
          out-delay 4;
        }
        group abcd {
          preference 10;
          hold-time 10;
          out-delay 10;
        }
      }
    }
  }
}
protocols {
  bgp {
    group abcd {
      apply-groups one;
    }
  }
}

```

```
[edit]
user@host# show | display inheritance
protocols {
  bgp {
    group abcd {
      ##
      ## '1' was inherited from group 'one'
      ##
      preference 1;
      ##
      ## '10' was inherited from group 'one'
      ##
      hold-time 10;
      ##
      ## '3' was inherited from group 'one'
      ##
      out-delay 3;
    }
  }
}
```

Configuring Sets of Statements with Configuration Groups

When sets of statements exist in configuration groups, all values are inherited. For example:

```
[edit]
user@host# show
groups {
  basic {
    snmp {
      interface so-1/1/1.0;
    }
  }
}
apply-groups basic;
snmp {
  interface so-0/0/0.0;
}
[edit]
user@host# show | display inheritance
snmp {
  ##
  ## 'so-1/1/1.0' was inherited from group 'basic'
  ##
  interface [ so-0/0/0.0 so-1/1/1.0 ];
}
```

For sets that are not displayed within brackets, all values are also inherited. For example:

```
[edit]
user@host# show
groups {
  worldwide {
```

```

    system {
        name-server {
            10.0.0.100;
            10.0.0.200;
        }
    }
}
apply-groups worldwide;
system {
    name-server {
        10.0.0.1;
        10.0.0.2;
    }
}
[edit]
user@host# show | display inheritance
system {
    name-server {
        ##
        ## '10.0.0.100' was inherited from group 'worldwide'
        ##
        10.0.0.100;
        ##
        ## '10.0.0.200' was inherited from group 'worldwide'
        ##
        10.0.0.200;
        10.0.0.1;
        10.0.0.2;
    }
}

```

Configuring Interfaces Using JUNOS Configuration Groups

You can use configuration groups to separate the common interface media parameters from the interface-specific addressing information. The following example places configuration data for ATM interfaces into a group called `atm-options`:

```

[edit]
user@host# show
groups {
    atm-options {
        interfaces {
            <at-*> {
                atm-options {
                    vpi 0 maximum-vcs 1024;
                }
                unit <*> {
                    encapsulation atm-snap;
                    point-to-point;
                    family iso;
                }
            }
        }
    }
}

```

```

}
apply-groups atm-options;
interfaces {
  at-0/0/0 {
    unit 100 {
      vci 0.100;
      family inet {
        address 10.0.0.100/30;
      }
    }
    unit 200 {
      vci 0.200;
      family inet {
        address 10.0.0.200/30;
      }
    }
  }
}
[edit]
user@host# show | display inheritance
interfaces {
  at-0/0/0 {
    ##
    ## "atm-options" was inherited from group "atm-options"
    ##
    atm-options {
      ##
      ## "1024" was inherited from group "atm-options"
      ##
      vpi 0 maximum-vcs 1024;
    }
    unit 100 {
      ##
      ## "atm-snap" was inherited from group "atm-options"
      ##
      encapsulation atm-snap;
      ##
      ## "point-to-point" was inherited from group "atm-options"
      ##
      point-to-point;
      vci 0.100;
      family inet {
        address 10.0.0.100/30;
      }
      ##
      ## "iso" was inherited from group "atm-options"
      ##
      family iso;
    }
    unit 200 {
      ##
      ## "atm-snap" was inherited from group "atm-options"
      ##
      encapsulation atm-snap;
      ##
      ## "point-to-point" was inherited from group "atm-options"

```



```

    ##
    point-to-point;
    vci 0.200;
    family inet {
        address 10.0.0.200/30;
    }
    ##
    ## "iso" was inherited from group "atm-options"
    ##
    family iso;
}
}
}
[edit]
user@host# show | display inheritance | except ##
interfaces {
  at-0/0/0 {
    atm-options {
      vpi 0 maximum-vcs 1024;
    }
    unit 100 {
      encapsulation atm-snap;
      point-to-point;
      vci 0.100;
      family inet {
        address 10.0.0.100/30;
      }
      family iso;
    }
    unit 200 {
      encapsulation atm-snap;
      point-to-point;
      vci 0.200;
      family inet {
        address 10.0.0.200/30;
      }
      family iso;
    }
  }
}
}

```

Configuring a Consistent Management IP Address

On routers with multiple Routing Engines, each Routing Engine is configured with a separate IP address for the management interface (fxp0). To access the master Routing Engine, you must know which Routing Engine is active and use the appropriate IP address.

Optionally, for consistent access to the master Routing Engine, you can configure an additional IP address and use this address for the management interface regardless of which Routing Engine is active. This additional IP address is active only on the management interface for the master Routing Engine. During switchover, the address moves to the new master Routing Engine.

In the following example, address **10.17.40.131** is configured for both Routing Engines and includes a **master-only** statement. With this configuration, the **10.17.40.131** address is active only on the master Routing Engine. The address remains consistent regardless of which Routing Engine is active. Address **10.17.40.132** is assigned to **fxp0** on **re0**, and **10.17.40.133** is assigned to **fxp0** on **re1**.

```
[edit groups re0 interfaces fxp0]
unit 0 {
  family inet {
    address 10.17.40.131/25 {
      master-only;
    }
    address 10.17.40.132/25;
  }
}
[edit groups re1 interfaces fxp0]
unit 0 {
  family inet {
    address 10.17.40.131/25 {
      master-only;
    }
    address 10.17.40.133/25;
  }
}
```

This feature is available on all routers that include dual Routing Engines. On the TX Matrix router, this feature is applicable to the switch-card chassis (SCC) only.

Configuring Peer Entities

In this example, we create a group **some-isp** that contains configuration data relating to another Internet service provider (ISP). We can then insert **apply-group** statements at any point to allow any location in the configuration hierarchy to inherit this data.

```
[edit]
user@host# show
groups {
  some-isp {
    interfaces {
      <xe-*> {
        gigether-options {
          flow-control;
        }
      }
    }
    protocols {
      bgp {
        group <*> {
          neighbor <*> {
            remove-private;
          }
        }
      }
    }
  }
}
```

```

    }
  }
  pim {
    interface <*> {
      version 1;
    }
  }
}
}
interfaces {
  xe-0/0/0 {
    apply-groups some-isp;
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
protocols {
  bgp {
    group main {
      neighbor 10.254.0.1 {
        apply-groups some-isp;
      }
    }
  }
  pim {
    interface xe-0/0/0.0 {
      apply-groups some-isp;
    }
  }
}
[edit]
user@host# show | display inheritance
interfaces {
  xe-0/0/0 {
    ##
    ## "gigether-options" was inherited from group "some-isp"
    ##
    gigether-options {
      ##
      ## "flow-control" was inherited from group "some-isp"
      ##
      flow-control;
    }
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
protocols {
  bgp {

```

```

group main {
  neighbor 10.254.0.1 {
    ##
    ## "remove-private" was inherited from group "some-isp"
    ##
    remove-private;
  }
}
pim {
  interface xe-0/0/0.0 {
    ##
    ## "1" was inherited from group "some-isp"
    ##
    version 1;
  }
}
}

```

Establishing Regional Configurations

In this example, one group is populated with configuration data that is standard throughout the company, while another group contains regional deviations from this standard:

```

[edit]
user@host# show
groups {
  standard {
    interfaces {
      <t3-*> {
        t3-options {
          compatibility-mode larscom subrate 10;
          idle-cycle-flag ones;
        }
      }
    }
  }
  northwest {
    interfaces {
      <t3-*> {
        t3-options {
          long-buildout;
          compatibility-mode kentrox;
        }
      }
    }
  }
}
apply-groups standard;
interfaces {
  t3-0/0/0 {
    apply-groups northwest;
  }
}

```

```
[edit]
user@host# show | display inheritance
interfaces {
  t3-0/0/0 {
    ##
    ## "t3-options" was inherited from group "northwest"
    ##
    t3-options {
      ##
      ## "long-buildout" was inherited from group "northwest"
      ##
      long-buildout;
      ##
      ## "kentrox" was inherited from group "northwest"
      ##
      compatibility-mode kentrox;
      ##
      ## "ones" was inherited from group "standard"
      ##
      idle-cycle-flag ones;
    }
  }
}
```

Selecting Wildcard Names

You can combine wildcarding and thoughtful use of names in statements to tailor statement values:

```
[edit]
user@host# show
groups {
  mpls-conf {
    protocols {
      mpls {
        label-switched-path <*-major> {
          retry-timer 5;
          bandwidth 155m;
          optimize-timer 60;
        }
        label-switched-path <*-minor> {
          retry-timer 15;
          bandwidth 64k;
          optimize-timer 120;
        }
      }
    }
  }
}
apply-groups mpls-conf;
protocols {
  mpls {
    label-switched-path metro-major {
      to 10.0.0.10;
    }
  }
}
```

```

        label-switched-path remote-minor {
            to 10.0.0.20;
        }
    }
}
[edit]
user@host# show | display inheritance
protocols {
    mpls {
        label-switched-path metro-major {
            to 10.0.0.10;
            ##
            ## "5" was inherited from group "mpls-conf"
            ##
            retry-timer 5;
            ## "155m" was inherited from group "mpls-conf"
            ##
            bandwidth 155m;
            ##
            ## "60" was inherited from group "mpls-conf"
            ##
            optimize-timer 60;
        }
        label-switched-path remote-minor {
            to 10.0.0.20;
            ##
            ## "15" was inherited from group "mpls-conf"
            ##
            retry-timer 15;
            ##
            ## "64k" was inherited from group "mpls-conf"
            ##
            bandwidth 64k;
            ##
            ## "120" was inherited from group "mpls-conf"
            ##
            optimize-timer 120;
        }
    }
}

```

Using JUNOS Defaults Groups

JUNOS Software provides a hidden and immutable configuration group called `junos-defaults` that is automatically applied to the configuration of your router. The `junos-defaults` group contains preconfigured statements that contain predefined values for common applications. Some of the statements must be referenced to take effect, such as definitions for applications (for example, FTP or telnet settings). Other statements are applied automatically, such as terminal settings.



NOTE: Many identifiers included in the `junos-defaults` configuration group begin with the name `junos-`. Because identifiers beginning with the name `junos-` are reserved for use by Juniper Networks, you cannot define any configuration objects using this name.

You cannot include `junos-defaults` as a configuration group name in an `apply-groups` statement.

To view the full set of available preset statements from the JUNOS defaults group, issue the `show groups junos-defaults` configuration mode command at the top level of the configuration. The following example displays a partial list of JUNOS defaults groups:

```
user@host# show groups junos-defaults
# Make vt100 the default for the console port
system {
  ports {
    console type vt100;
  }
}
applications {
  # File Transfer Protocol
  application junos-ftp {
    application-protocol ftp;
    protocol tcp;
    destination-port 21;
  }
  # Trivial File Transfer Protocol
  application junos-tftp {
    application-protocol tftp;
    protocol udp;
    destination-port 69;
  }
  # RPC port mapper on TCP
  application junos-rpc-portmap-tcp {
    application-protocol rpc-portmap;
    protocol tcp;
    destination-port 111;
  }
  # RPC port mapper on UDP
}
```

To reference statements available from the `junos-defaults` group, include the selected `junos- default-name` statement at the applicable hierarchy level.

Example: Referencing the Preset Statement

The following example is a preset statement from the JUNOS defaults group that is available for FTP in a stateful firewall:

```
[edit]
groups {
```

```

junos-defaults {
  applications {
    application junos-ftp {# Use FTP default configuration
      application-protocol ftp;
      protocol tcp;
      destination-port 21;
    }
  }
}

```

To reference a preset JUNOS default statement from the JUNOS defaults group, include the `junos- default-name` statement at the applicable hierarchy level. For example, to reference the JUNOS default statement for FTP in a stateful firewall, include the `junos-ftp` statement at the `[edit services stateful-firewall rule rule-name term term-name from applications]` hierarchy level:

```

[edit]
services {
  stateful-firewall {
    rule my-rule {
      term my-term {
        from {
          applications junos-ftp; #Reference predefined statement, junos-ftp,
        }
      }
    }
  }
}

```

Example: Viewing Default Statements That Have Been Applied to the Configuration

To view the JUNOS defaults that have been applied to the configuration, issue the `show | display inheritance defaults` command. For example, to view the inherited JUNOS defaults at the `[edit system ports]` hierarchy level:

```

user@host# show system ports | display inheritance defaults
## ## 'console' was inherited from group 'junos-defaults'
## 'vt100' was inherited from group 'junos-defaults'
## console type vt100;

```

If you choose not to use existing JUNOS default statements, you can create your own configuration groups manually. For more information about manually creating of configuration groups, see “Configuration Groups” on page 145 and “Configuring Configuration Groups” on page 147.

Inheritance Model

Configuration groups use true inheritance, which involves a dynamic, ongoing relationship between the source of the configuration data and the target of that data. Data values changed in the configuration group are automatically inherited by the target. The target need not contain the inherited information, although the inherited values can be overridden in the target without affecting the source from which they were inherited.

This inheritance model allows you to see only the instance-specific information without seeing the inherited details. A command pipe in configuration mode allows you to display the inherited data.

Creating and Applying Configuration Groups

To configure configuration groups and inheritance, you can include the **groups** statement at the **[edit]** hierarchy level:

```
[edit]
groups {
  group-name {
    configuration-data;
  }
}
```

Include the **apply-groups [group-names]** statement anywhere in the configuration that the configuration statements contained in a configuration group are needed.

Configuration Groups Configuration Guidelines

For areas of your configuration to inherit configuration statements, you must first put the statements into a configuration group and then apply that group to the levels in the configuration hierarchy that require the statements. This section covers the following topics:

- Creating a Configuration Group on page 165
- Applying a Configuration Group on page 167
- Disabling Inheritance of a Configuration Group on page 169
- Displaying Inherited Values on page 170
- Using Wildcards with Configuration Groups on page 171

Creating a Configuration Group

To create a configuration group, include the **groups** statement at the **[edit]** hierarchy level:

```
[edit]
groups {
  group-name {
    configuration-data;
  }
  lccn-re0 {
    configuration-data;
  }
  lccn-re1 {
    configuration-data;
  }
}
```

group-name is the name of a configuration group. You can configure more than one configuration group by specifying multiple *group-name* statements. However, you cannot use the prefix **junos-** in a group name because it is reserved for use by JUNOS Software. Similarly, the configuration group **juniper-ais** is reserved exclusively for Juniper Advanced Insight Solutions (AIS)-related configuration. For more information on the **juniper-ais** configuration group, see the Juniper Networks *Advanced Insight Solutions Guide*.

One reason for the naming restriction is a configuration group called **junos-defaults**. This preset configuration group is applied to the configuration automatically. You cannot modify or remove the **junos-defaults** configuration group. For more information about the JUNOS default configuration group, see “Using JUNOS Defaults Groups” on page 183.

On routers that support multiple Routing Engines, you can also specify two special group names:

- **re0**—Configuration statements applied to the Routing Engine in slot 0.
- **re1**—Configuration statements applied to the Routing Engine in slot 1.

The configuration specified in group **re0** is only applied if the current Routing Engine is in slot 0; likewise, the configuration specified in group **re1** is only applied if the current Routing Engine is in slot 1. Therefore, both Routing Engines can use the same configuration file, each using only the configuration statements that apply to it. Each **re0** or **re1** group contains at a minimum the configuration for the hostname and the management interface (**fxp0**). If each Routing Engine uses a different management interface, the group also should contain the configuration for the backup router and static routes.

In addition, the TX Matrix router supports group names for the Routing Engines in each T640 router attached to the routing matrix. Providing special group names for all Routing Engines in the routing matrix allows you to configure the individual Routing Engines in each T640 router differently. Parameters that are not configured at the **[edit groups]** hierarchy level apply to all Routing Engines in the routing matrix.

configuration-data contains the configuration statements applied elsewhere in the configuration with the **apply-groups** statement. To have a configuration inherit the statements in a configuration group, include the **apply-groups** statement. For information about the **apply-groups** statement, see “Applying a Configuration Group” on page 167.

The group names for Routing Engines on the TX Matrix router have the following formats:

- **lccn-re0**—Configuration statements applied to the Routing Engine in slot 0 in a specified T640 router.
- **lccn-re1**—Configuration statements applied to the Routing Engine in slot 1 in a specified T640 router.

n identifies the T640 router and can be from 0 through 3. For example, to configure Routing Engine 1 properties for **lcc3**, you include statements at the **[edit groups]**

lcc3-re1] hierarchy level. For information about the TX Matrix router and routing matrix, see the *JUNOS System Basics Configuration Guide*

Applying a Configuration Group

To have a configuration inherit the statements in a configuration group, include the `apply-groups` statement:

```
apply-groups [ group-names ];
```

If you specify more than one group name, list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.

For routers that support multiple Routing Engines, you can specify `re0` and `re1` group names. The configuration specified in group `re0` is only applied if the current Routing Engine is in slot 0; likewise, the configuration specified in group `re1` is only applied if the current Routing Engine is in slot 1. Therefore, both Routing Engines can use the same configuration file, each using only the configuration statements that apply to it. Each `re0` or `re1` group contains at a minimum the configuration for the hostname and the management interface (`fxp0`). If each Routing Engine uses a different management interface, the group also should contain the configuration for the backup router and static routes.

You can include only one `apply-groups` statement at each specific level of the configuration hierarchy. The `apply-groups` statement at a specific hierarchy level lists the configuration groups to be added to the containing statement's list of configuration groups.

Values specified at the specific hierarchy level override values inherited from the configuration group.

Groups listed in nested `apply-groups` statements take priority over groups in outer statements. In the following example, the BGP neighbor `10.0.0.1` inherits configuration data from group `one` first, then from groups `two` and `three`. Configuration data in group `one` overrides data in any other group. Data from group `ten` is used only if a statement is not contained in any other group.

```
apply-groups [ eight nine ten ];
protocols {
  apply-groups seven;
  bgp {
    apply-groups [ five six ];
    group some-bgp-group {
      apply-groups four;
      neighbor 10.0.0.1 {
        apply-groups [ one two three ];
      }
    }
  }
}
```

Example: Configuring and Applying Configuration Groups

In this example, the SNMP configuration is divided between the group **basic** and the normal configuration hierarchy.

There are a number of advantages to placing the system-specific configuration (SNMP contact) into a configuration group and thus separating it from the normal configuration hierarchy—the user can replace (using the **load replace** command) either section without discarding data from the other.

In addition, setting a contact for a specific box is now possible because the group data would be hidden by the router-specific data.

```
[edit]
groups {
  basic {# User-defined group name
    snmp {# This group contains some SNMP data
      contact "My Engineering Group";
      community BasicAccess {
        authorization read-only;
      }
    }
  }
  apply-groups basic;# Enable inheritance from group "basic"
  snmp {# Some normal (non-group) configuration
    location "West of Nowhere";
  }
```

This configuration is equivalent to the following:

```
[edit]
snmp {
  location "West of Nowhere";
  contact "My Engineering Group";
  community BasicAccess {
    authorization read-only;
  }
}
```

For information about how to disable inheritance of a configuration group, see “Disabling Inheritance of a Configuration Group” on page 169.

Example: Creating and Applying Configuration Groups on a TX Matrix Router

```
[edit]
groups {
  re0 { # Routing Engine 0 on TX Matrix router
    system {
      host-name <host-name>;
      backup-router <ip-address>;
    }
    interfaces {
      fxp0 {
        unit 0 {
          family inet {
```

```

        address <ip-address>;
    }
}
}
}
re1 { # Routing Engine 1 on TX Matrix router
system {
    host-name <host-name>;
    backup-router <ip-address>;
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address <ip-address>;
            }
        }
    }
}
}
lcc0-re0 { # Routing Engine 0 on T640 router numbered 0
system {
    host-name <host-name>;
    backup-router <ip-address>;
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address <ip-address>;
            }
        }
    }
}
lcc0-re1 { # Routing Engine 1 on T640 router numbered 0
system {
    host-name <host-name>;
    backup-router <ip-address>;
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address <ip-address>;
            }
        }
    }
}
}
apply-groups [ re0 re1 lcc0-re0 lcc0-re1 ];
}

```

Disabling Inheritance of a Configuration Group

To disable inheritance of a configuration group at any level except the top level of the hierarchy, include the `apply-groups-except` statement:

```
apply-groups-except [ group-names ];
```

This statement is useful when you use the **apply-group** statement at a specific hierarchy level but also want to override the values inherited from the configuration group for a specific parameter.

Example: Disabling Inheritance on Interface so-1/1/0

In the following example, the **apply-groups** statement is applied globally at the interfaces level. The **apply-groups-except** statement is also applied at interface **so-1/1/0** so that it uses the default values **hold-time** and **link-mode**.

```
[edit]
groups { # "groups" is a top-level statement
global { # User-defined group name
interfaces {
  <*> {
    hold-time down 640;
    link-mode full-duplex;
  }
}
apply-groups global;
interfaces {
  so-1/1/0 {
    apply-groups-except global; # Disables inheritance from group "global"
    # so-1/1/0 uses default value for "hold-time"
  }
  # and "link-mode"
}
```

For information about applying a configuration group, see “Applying a Configuration Group” on page 167.

Displaying Inherited Values

Configuration groups can add some confusion regarding the actual values used by the router, because configuration data can be inherited from configuration groups. To view the actual values used by the router, use the **display inheritance** command after the pipe in a **show** command. This command displays the inherited statements at the level at which they are inherited and the group from which they have been inherited.

```
[edit]
user@host# show | display inheritance
snmp {
  location "West of Nowhere";
  ##
  ## 'My Engineering Group' was inherited from group 'basic'
  ##
  contact "My Engineering Group";
  ##
  ## 'BasicAccess' was inherited from group 'basic'
  ##
  community BasicAccess {
```

```

##
## 'read-only' was inherited from group 'basic'
##
authorization read-only;
}
}

```

To display the expanded configuration (the configuration, including the inherited statements) without the `##` lines, use the `except` command after the pipe in a `show` command:

```

[edit]
user@host# show | display inheritance | except ##
snmp {
  location "West of Nowhere";
  contact "My Engineering Group";
  community BasicAccess {
    authorization read-only;
  }
}

```

Using Wildcards with Configuration Groups

You can use wildcards to identify names and allow one statement to provide data for a variety of statements. For example, grouping the configuration of the `sonet-options` statement over all SONET/SDH interfaces or the dead interval for OSPF over all Asynchronous Transfer Mode (ATM) interfaces simplifies configuration files and eases their maintenance.

Using wildcards in normal configuration data is done in a style that is consistent with that used with traditional UNIX shell wildcards. In this style, you can use the following metacharacters:

- Asterisk (`*`)—Matches any string of characters.
- Question mark (`?`)—Matches any single character.
- Open bracket (`[`)—Introduces a character class.
- Close bracket (`]`)—Indicates the end of a character class. If the close bracket is missing, the open bracket matches a `[` rather than introduce a character class.
- A character class matches any of the characters between the square brackets. Character classes must be enclosed in quotation marks (`" "`).
- Hyphen (`-`)—Specifies a range of characters.
- Exclamation point (`!`)—The character class can be complemented by making an exclamation point the first character of the character class. To include a `]` in a character class, make it the first character listed (after the `!`, if any). To include a minus sign, make it the first or last character listed.

Wildcarding in configuration groups follows the same rules, but the wildcard pattern must be enclosed in angle brackets (`<pattern>`) to differentiate it from other wildcarding in the configuration file. For example:

```

[edit]

```

```

groups {
  sonet-default {
    interfaces {
      <so-*> {
        sonet-options {
          payload-scrambler;
          rfc-2615;
        }
      }
    }
  }
}

```

Wildcard expressions match (and provide configuration data for) existing statements in the configuration that match their expression only. In the example above, the expression `<so-*>` passes its `sonet-options` statement to any interface that matches the expression `so-*`.

Angle brackets allow you to pass normal wildcarding through without modification. In all matching within the configuration, whether it is done with or without wildcards, the first item encountered in the configuration that matches is used. In the following example, data from the wildcarded BGP groups is inherited in the order in which the groups are listed. The preference value from `<*a*>` overrides the preference in `<*b*>`, just as the `p` value from `<*c*>` overrides the one from `<*d*>`. Data values from any of these groups override the data values from `abcd`.

```

[edit]
user@host# show
groups {
  one {
    protocols {
      bgp {
        group <*a*> {
          preference 1;
        }
        group <*b*> {
          preference 2;
        }
        group <*c*> {
          out-delay 3;
        }
        group <*d*> {
          out-delay 4;
        }
        group abcd {
          preference 10;
          hold-time 10;
          out-delay 10;
        }
      }
    }
  }
}
protocols {
  bgp {

```



```

        group abcd {
            apply-groups one;
        }
    }
}
[edit]
user@host# show | display inheritance
protocols {
    bgp {
        group abcd {
            ##
            ## '1' was inherited from group 'one'
            ##
            preference 1;
            ##
            ## '10' was inherited from group 'one'
            ##
            hold-time 10;
            ##
            ## '3' was inherited from group 'one'
            ##
            out-delay 3;
        }
    }
}

```

Example: Using Wildcards with Configuration Groups

The following example demonstrates the use of wildcarding. The interface `so-0/0/0` inherits data from the various SONET/SDH interface wildcard patterns in group `one`.

```

[edit]
user@host# show
groups {
    one {
        interfaces {
            <so-*> {
                sonet-options {
                    rfc-2615;
                }
            }
            <so-0/*> {
                sonet-options {
                    fcs 32;
                }
            }
            <so-*/0/*> {
                sonet-options {
                    fcs 16;
                }
            }
            <so-*/*/0> {
                sonet-options {
                    payload-scrambler;
                }
            }
        }
    }
}

```

```
}  
}  
}  
}  
}  
} apply-groups one;  
interfaces {  
    so-0/0/0 {  
        unit 0 {  
            family inet {  
                address 10.0.0.1/8;  
            }  
        }  
    }  
}  
}  
[edit]  
user@host# show | display inheritance  
interfaces {  
    so-0/0/0 {  
        ##  
        ## 'sonet-options' was inherited from group 'one'  
        ##  
        sonet-options {  
            ##  
            ## '32' was inherited from group 'one'  
            ##  
            fcs 32;  
            ##  
            ## 'payload-scrambler' was inherited from group 'one'  
            ##  
            payload-scrambler;  
            ##  
            ## 'rfc-2615' was inherited from group 'one'  
            ##  
            rfc-2615;  
        }  
        unit 0 {  
            family inet {  
                address 10.0.0.1/8;  
            }  
        }  
    }  
}
```

Examples: Configuration Groups

The following examples illustrate ways to use configuration groups and inheritance:

- Configuring Sets of Statements with Configuration Groups on page 175
- Configuring Interfaces on page 176
- Configuring a Consistent Management IP Address on page 178
- Configuring Peer Entities on page 179
- Establishing Regional Configurations on page 181
- Selecting Wildcard Names on page 182

Configuring Sets of Statements with Configuration Groups

When sets of statements exist in configuration groups, all values are inherited. For example:

```
[edit]
user@host# show
groups {
  basic {
    snmp {
      interface so-1/1/1.0;
    }
  }
}
apply-groups basic;
snmp {
  interface so-0/0/0.0;
}
[edit]
user@host# show | display inheritance
snmp {
  ##
  ## 'so-1/1/1.0' was inherited from group 'basic'
  ##
  interface [ so-0/0/0.0 so-1/1/1.0 ];
}
```

For sets that are not displayed within brackets, all values are also inherited. For example:

```
[edit]
user@host# show
groups {
  worldwide {
    system {
      name-server {
        10.0.0.100;
        10.0.0.200;
      }
    }
  }
}
apply-groups worldwide;
system {
  name-server {
    10.0.0.1;
    10.0.0.2;
  }
}
[edit]
user@host# show | display inheritance
system {
  name-server {
    ##
```

```

    ## '10.0.0.100' was inherited from group 'worldwide'
    ##
    10.0.0.100;
    ##
    ## '10.0.0.200' was inherited from group 'worldwide'
    ##
    10.0.0.200;
    10.0.0.1;
    10.0.0.2;
  }
}

```

Configuring Interfaces

You can use configuration groups to separate the common interface media parameters from the interface-specific addressing information. The following example places configuration data for ATM interfaces into a group called **atm-options**:

```

[edit]
user@host# show
groups {
  atm-options {
    interfaces {
      <at-*> {
        atm-options {
          vpi 0 maximum-vcs 1024;
        }
        unit <*> {
          encapsulation atm-snap;
          point-to-point;
          family iso;
        }
      }
    }
  }
}
apply-groups atm-options;
interfaces {
  at-0/0/0 {
    unit 100 {
      vci 0.100;
      family inet {
        address 10.0.0.100/30;
      }
    }
    unit 200 {
      vci 0.200;
      family inet {
        address 10.0.0.200/30;
      }
    }
  }
}
[edit]
user@host# show | display inheritance

```

```

interfaces {
  at-0/0/0 {
    ##
    ## "atm-options" was inherited from group "atm-options"
    ##
    atm-options {
      ##
      ## "1024" was inherited from group "atm-options"
      ##
      vpi 0 maximum-vcs 1024;
    }
    unit 100 {
      ##
      ## "atm-snap" was inherited from group "atm-options"
      ##
      encapsulation atm-snap;
      ##
      ## "point-to-point" was inherited from group "atm-options"
      ##
      point-to-point;
      vci 0.100;
      family inet {
        address 10.0.0.100/30;
      }
      ##
      ## "iso" was inherited from group "atm-options"
      ##
      family iso;
    }
    unit 200 {
      ##
      ## "atm-snap" was inherited from group "atm-options"
      ##
      encapsulation atm-snap;
      ##
      ## "point-to-point" was inherited from group "atm-options"
      ##
      point-to-point;
      vci 0.200;
      family inet {
        address 10.0.0.200/30;
      }
      ##
      ## "iso" was inherited from group "atm-options"
      ##
      family iso;
    }
  }
}
[edit]
user@host# show | display inheritance | except ##
interfaces {
  at-0/0/0 {
    atm-options {
      vpi 0 maximum-vcs 1024;
    }
  }
}

```

```

    unit 100 {
        encapsulation atm-snap;
        point-to-point;
        vci 0.100;
        family inet {
            address 10.0.0.100/30;
        }
        family iso;
    }
    unit 200 {
        encapsulation atm-snap;
        point-to-point;
        vci 0.200;
        family inet {
            address 10.0.0.200/30;
        }
        family iso;
    }
}

```

Configuring a Consistent Management IP Address

On routers with multiple Routing Engines, each Routing Engine is configured with a separate IP address for the management interface (**fxp0**). To access the master Routing Engine, you must know which Routing Engine is active and use the appropriate IP address.

Optionally, for consistent access to the master Routing Engine, you can configure an additional IP address and use this address for the management interface regardless of which Routing Engine is active. This additional IP address is active only on the management interface for the master Routing Engine. During switchover, the address moves to the new master Routing Engine.

In the following example, address **10.17.40.131** is configured for both Routing Engines and includes a **master-only** statement. With this configuration, the **10.17.40.131** address is active only on the master Routing Engine. The address remains consistent regardless of which Routing Engine is active. Address **10.17.40.132** is assigned to **fxp0** on **re0**, and **10.17.40.133** is assigned to **fxp0** on **re1**.

```

[edit groups re0 interfaces fxp0]
unit 0 {
    family inet {
        address 10.17.40.131/25 {
            master-only;
        }
        address 10.17.40.132/25;
    }
}
[edit groups re1 interfaces fxp0]
unit 0 {
    family inet {
        address 10.17.40.131/25 {

```

```

        master-only;
    }
    address 10.17.40.133/25;
}
}

```

This feature is available on all routers that include dual Routing Engines. On the TX Matrix router, this feature is applicable to the switch-card chassis (SCC) only.

Configuring Peer Entities

In this example, we create a group **some-isp** that contains configuration data relating to another Internet service provider (ISP). We can then insert **apply-group** statements at any point to allow any location in the configuration hierarchy to inherit this data.

```

[edit]
user@host# show
groups {
  some-isp {
    interfaces {
      <xe-*> {
        giggerther-options {
          flow-control;
        }
      }
    }
    protocols {
      bgp {
        group <*> {
          neighbor <*> {
            remove-private;
          }
        }
      }
      pim {
        interface <*> {
          version 1;
        }
      }
    }
  }
}
interfaces {
  xe-0/0/0 {
    apply-groups some-isp;
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
protocols {

```

```

bgp {
  group main {
    neighbor 10.254.0.1 {
      apply-groups some-isp;
    }
  }
}
pim {
  interface xe-0/0/0.0 {
    apply-groups some-isp;
  }
}
}
[edit]
user@host# show | display inheritance
interfaces {
  xe-0/0/0 {
    ##
    ## "gigether-options" was inherited from group "some-isp"
    ##
    gigether-options {
      ##
      ## "flow-control" was inherited from group "some-isp"
      ##
      flow-control;
    }
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
protocols {
  bgp {
    group main {
      neighbor 10.254.0.1 {
        ##
        ## "remove-private" was inherited from group "some-isp"
        ##
        remove-private;
      }
    }
  }
}
pim {
  interface xe-0/0/0.0 {
    ##
    ## "1" was inherited from group "some-isp"
    ##
    version 1;
  }
}
}

```


Establishing Regional Configurations

In this example, one group is populated with configuration data that is standard throughout the company, while another group contains regional deviations from this standard:

```
[edit]
user@host# show
groups {
  standard {
    interfaces {
      <t3-*> {
        t3-options {
          compatibility-mode larscom subrate 10;
          idle-cycle-flag ones;
        }
      }
    }
  }
  northwest {
    interfaces {
      <t3-*> {
        t3-options {
          long-buildout;
          compatibility-mode kentrox;
        }
      }
    }
  }
}
apply-groups standard;
interfaces {
  t3-0/0/0 {
    apply-groups northwest;
  }
}
[edit]
user@host# show | display inheritance
interfaces {
  t3-0/0/0 {
    ##
    ## "t3-options" was inherited from group "northwest"
    ##
    t3-options {
      ##
      ## "long-buildout" was inherited from group "northwest"
      ##
      long-buildout;
      ##
      ## "kentrox" was inherited from group "northwest"
      ##
      compatibility-mode kentrox;
      ##
      ## "ones" was inherited from group "standard"
```

```

        ##
        idle-cycle-flag ones;
    }
}

```

Selecting Wildcard Names

You can combine wildcarding and thoughtful use of names in statements to tailor statement values:

```

[edit]
user@host# show
groups {
  mpls-conf {
    protocols {
      mpls {
        label-switched-path <*-major> {
          retry-timer 5;
          bandwidth 155m;
          optimize-timer 60;
        }
        label-switched-path <*-minor> {
          retry-timer 15;
          bandwidth 64k;
          optimize-timer 120;
        }
      }
    }
  }
}
apply-groups mpls-conf;
protocols {
  mpls {
    label-switched-path metro-major {
      to 10.0.0.10;
    }
    label-switched-path remote-minor {
      to 10.0.0.20;
    }
  }
}
[edit]
user@host# show | display inheritance
protocols {
  mpls {
    label-switched-path metro-major {
      to 10.0.0.10;
      ##
      ## "5" was inherited from group "mpls-conf"
      ##
      retry-timer 5;
      ## "155m" was inherited from group "mpls-conf"
      ##
      bandwidth 155m;
    }
  }
}

```

```

##
## "60" was inherited from group "mpls-conf"
##
optimize-timer 60;
}
label-switched-path remote-minor {
to 10.0.0.20;
##
## "15" was inherited from group "mpls-conf"
##
retry-timer 15;
##
## "64k" was inherited from group "mpls-conf"
##
bandwidth 64k;
##
## "120" was inherited from group "mpls-conf"
##
optimize-timer 120;
}
}
}

```

Using JUNOS Defaults Groups

JUNOS Software provides a hidden and immutable configuration group called `junos-defaults` that is automatically applied to the configuration of your router. The `junos-defaults` group contains preconfigured statements that contain predefined values for common applications. Some of the statements must be referenced to take effect, such as definitions for applications (for example, FTP or telnet settings). Other statements are applied automatically, such as terminal settings.



NOTE: Many identifiers included in the `junos-defaults` configuration group begin with the name `junos-`. Because identifiers beginning with the name `junos-` are reserved for use by Juniper Networks, you cannot define any configuration objects using this name.

You cannot include `junos-defaults` as a configuration group name in an `apply-groups` statement.

To view the full set of available preset statements from the JUNOS defaults group, issue the `show groups junos-defaults` configuration mode command at the top level of the configuration. The following example displays a partial list of JUNOS defaults groups:

```

user@host# show groups junos-defaults
# Make vt100 the default for the console port
system {
  ports {
    console type vt100;
  }
}

```

```

}
applications {
  # File Transfer Protocol
  application junos-ftp {
    application-protocol ftp;
    protocol tcp;
    destination-port 21;
  }
  # Trivial File Transfer Protocol
  application junos-tftp {
    application-protocol tftp;
    protocol udp;
    destination-port 69;
  }
  # RPC port mapper on TCP
  application junos-rpc-portmap-tcp {
    application-protocol rpc-portmap;
    protocol tcp;
    destination-port 111;
  }
  # RPC port mapper on UDP
}

```

To reference statements available from the `junos-defaults` group, include the selected `junos- default-name` statement at the applicable hierarchy level.

Example: Referencing a Preset Statement

The following example is a preset statement from the JUNOS defaults group that is available for FTP in a stateful firewall:

```

[edit]
groups {
  junos-defaults {
    applications {
      application junos-ftp {# Use FTP default configuration
        application-protocol ftp;
        protocol tcp;
        destination-port 21;
      }
    }
  }
}

```

To reference a preset JUNOS default statement from the JUNOS defaults group, include the `junos- default-name` statement at the applicable hierarchy level. For example, to reference the JUNOS default statement for FTP in a stateful firewall, include the `junos-ftp` statement at the `[edit services stateful-firewall rule rule-name term term-name from applications]` hierarchy level:

```

[edit]
services {
  stateful-firewall {
    rule my-rule {
      term my-term {
        from {

```

```

        applications junos-ftp; #Reference predefined statement, junos-ftp,
    }
}
}
}
}

```

Example: Viewing Default Statements That Have Been Applied to the Configuration

To view the JUNOS defaults that have been applied to the configuration, issue the `show | display inheritance defaults` command. For example, to view the inherited JUNOS defaults at the `[edit system ports]` hierarchy level:

```

user@host# show system ports | display inheritance defaults
## ## 'console' was inherited from group 'junos-defaults'
## 'vt100' was inherited from group 'junos-defaults'
## console type vt100;

```

If you choose not to use existing JUNOS default statements, you can create your own configuration groups manually. For more information about manually creating of configuration groups, see “Configuration Groups” on page 145 and “Configuring Configuration Groups” on page 147.

Chapter 11

Summary of Configuration Group Statements

The following sections explain each of the configuration group statements. The statements are organized alphabetically.

apply-groups

Syntax	<code>apply-groups [<i>group-names</i>];</code>
Hierarchy Level	All hierarchy levels
Release Information	Statement introduced before JUNOS Release 7.4.
Description	<p>Apply a configuration group to a specific hierarchy level in a configuration, to have a configuration inherit the statements in the configuration group.</p> <p>You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.</p>
Options	<i>group-names</i> —One or more names specified in the groups statement.
Usage Guidelines	See “Applying a Configuration Group” on page 167.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	groups

apply-groups-except

Syntax	<code>apply-groups-except [<i>group-names</i>];</code>
Hierarchy Level	All hierarchy levels except the top level
Release Information	Statement introduced before JUNOS Release 7.4.
Description	Disable inheritance of a configuration group.
Options	<i>group-names</i> —One or more names specified in the groups statement.
Usage Guidelines	See “Disabling Inheritance of a Configuration Group” on page 169.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	groups

groups

Syntax

```
groups {
  group-name {
    configuration-data;
  }
  lccn-re0 {
    configuration-data;
  }
  lccn-re1 {
    configuration-data;
  }
}
```

Hierarchy Level [edit]

Release Information Statement introduced before JUNOS Release 7.4.

Description Create a configuration group.

Options *configuration-data*—The configuration statements that are to be applied elsewhere in the configuration with the **apply-groups** statement, to have the target configuration inherit the statements in the group.

group-name—Name of the configuration group. To configure multiple groups, specify more than one *group-name*. On routers that support multiple Routing Engines, you can also specify two special group names:

- **re0**—Configuration statements that are to be applied to the Routing Engine in slot 0.
- **re1**—Configuration statements that are to be applied to the Routing Engine in slot 1.

The configuration specified in group **re0** is applied only if the current Routing Engine is in slot 0; likewise, the configuration specified in group **re1** is applied only if the current Routing Engine is in slot 1. Therefore, both Routing Engines can use the same configuration file, each using only the configuration statements that apply to it. Each **re0** or **re1** group contains at a minimum the configuration for the hostname and the management interface (**fxp0**). If each Routing Engine uses a different management interface, the group also should contain the configuration for the backup router and static routes.

(Routing matrix only) The TX Matrix router supports group names for the Routing Engines in each connected T640 router in the following formats:

- **lccn-re0**—Configuration statements applied to the Routing Engine in slot 0 of the specified T640 router that is connected to a TX Matrix router.
- **lccn-re1**—Configuration statements applied to the specified to the Routing Engine in slot 1 of the specified T640 router that is connected to a TX Matrix router.

n identifies the T640 router and can be from 0 through 3.

Usage Guidelines See “Creating a Configuration Group” on page 165.

Required Privilege Level configure—To enter configuration mode.

Related Topics apply-groups
apply-groups-except

Part 4

CLI Command Summaries

- Summary of CLI Environment Commands on page 193
- Summary of CLI Configuration Mode Commands on page 209
- Summary of CLI Operational Mode Commands on page 243

Chapter 12

Summary of CLI Environment Commands

The following sections explain each of the command-line interface (CLI) environment commands described in this book. The commands are organized alphabetically.

set cli complete-on-space

Syntax	set cli complete-on-space (off on)
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set the CLI to complete a partial command entry when you type a space or a tab. This is the default behavior of the CLI.
Sample Output	<p>In the following example, pressing the Spacebar changes the partial command entry from <code>com</code> to <code>complete-on-space</code>. The example shows how adding the keyword <code>off</code> at the end of the command disables command completion.</p> <pre>user@host> set cli com<Space> user@host>set cli complete-on-space off Disabling complete-on-space</pre>
Options	<p><code>off</code>—Turn off command completion.</p> <p><code>on</code>—Allow either a space or a tab to be used for command completion.</p>
Usage Guidelines	See “Setting Command Completion” on page 129.
Required Privilege Level	view

set cli directory

Syntax	set cli directory <i>directory</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set the current working directory.
Sample Output	user@host> set cli directory /var/home/regress Current directory: /var/home/regress
Options	<i>directory</i> —Pathname of the working directory.
Usage Guidelines	See “Setting the CLI Directory” on page 128.
Required Privilege Level	view

set cli idle-timeout

Syntax `set cli idle-timeout <minutes>`

Release Information Command introduced before JUNOS Release 7.4.

Description Set the maximum time that an individual session can be idle before the user is logged off the router.

Sample Output

```
user@host> set cli idle-timeout 60
Idle timeout set to 60 minutes
```

Options *minutes*—(Optional) Maximum idle time. The range of values, in minutes, is 0 through 100,000. If you do not issue this command, and the user's login class does not specify this value, the user is never forced off the system after extended idle times. Setting the value to 0 disables the timeout.

Usage Guidelines See "Setting the Idle Timeout" on page 128.

Required Privilege Level view

set cli prompt

Syntax	set cli prompt <i>string</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set the prompt so that it is displayed within the CLI.
Sample Output	<pre>user@host> set cli prompt lab1-router></pre>
Options	<i>string</i> —CLI prompt string. To include spaces in the prompt, enclose the string in quotation marks. By default, the string is <i>username@hostname</i> .
Usage Guidelines	See “Setting the CLI Prompt” on page 128.
Required Privilege Level	view

set cli restart-on-upgrade

Syntax	set cli restart-on-upgrade string (off on)
Release Information	Command introduced before JUNOS Release 7.4.
Description	For an individual session, set the CLI to prompt you to restart the router after upgrading the software.
Sample Output	<pre>user@host> set cli restart-on-upgrade on Enabling restart-on-upgrade</pre>
Options	off—Disables the prompt. on—Enables the prompt.
Usage Guidelines	See “Setting the CLI to Prompt After a Software Upgrade” on page 129.
Required Privilege Level	view

set cli screen-length

Syntax	set cli screen-length <i>length</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set terminal screen length.
Sample Output	<pre>user@host> set cli screen-length 75 Screen length set to 75</pre>
Options	<p><i>length</i>—Number of lines of text that the terminal screen displays. The range of values, in number of lines, is 24 through 100,000. The default is 24.</p> <p>The point at which the —(more)— prompt appears on the screen is a function of this setting and the settings for the set cli screen-width and set cli terminal commands.</p>
Usage Guidelines	See “Setting the Screen Length” on page 130 and “Understanding the Screen Length and Width Settings” on page 130.
Required Privilege Level	view
Related Topics	<p>set cli screen-width</p> <p>set cli terminal</p> <p>show cli</p>

set cli screen-width

Syntax	set cli screen-width <i>width</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set the terminal screen width.
Sample Output	<pre>user@host> set cli screen-width Screen width set to 132</pre>
Options	<p><i>width</i> —Number of characters in a line. The range of values is 80 through 100,000. The default is 80.</p> <p>The point at which the <code>—(more)—</code> prompt appears on the screen is a function of this setting and the settings for the <code>set cli screen-length</code> and <code>set cli terminal</code> commands.</p>
Usage Guidelines	See “Setting the Screen Width” on page 130 and “Understanding the Screen Length and Width Settings” on page 130.
Required Privilege Level	view
Related Topics	<p>set cli screen-length</p> <p>set cli terminal</p> <p>show cli</p>

set cli terminal

Syntax	set cli terminal <i>terminal-type</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set the terminal type.
Sample Output	user@host> set cli terminal xterm
Options	<i>terminal-type</i> —Type of terminal that is connected to the Ethernet management port: <ul style="list-style-type: none">■ ansi—ANSI-compatible terminal (80 characters by 24 lines)■ small-xterm—Small xterm window (80 characters by 24 lines)■ vt100—VT100-compatible terminal (80 characters by 24 lines)■ xterm—Large xterm window (80 characters by 65 lines)
Usage Guidelines	See “Setting the Terminal Type” on page 128.
Required Privilege Level	view

set cli timestamp

Syntax	set cli timestamp (format <i>timestamp-format</i> disable)
Release Information	Command introduced before JUNOS Release 7.4.
Description	Set a timestamp for CLI output.
Sample Output	<pre>user@host> set cli timestamp format '%m-%d-%T' '04-21-17:39:13' CLI timestamp set to: '%m-%d-%T'</pre>
Options	<p>format <i>timestamp-format</i>—Set the data and time format for the timestamp. The timestamp format you specify can include the following placeholders in any order:</p> <ul style="list-style-type: none"> ■ %m—Two-digit month ■ %d—Two-digit date ■ %T—Six-digit hour, minute, and seconds <p>Enclose the format in single quotation marks ('). Do not use spaces. Use a hyphen (-) or similar character to separate placeholders.</p> <p>disable—Remove the timestamp from the CLI.</p>
Usage Guidelines	See “Setting the CLI Timestamp” on page 128.
Required Privilege Level	view

set date

Syntax `set date (date-time | ntp <ntp-server> <source-address source-address>)`

Release Information Command introduced before JUNOS Release 7.4.

Description Set the date and time.

Sample Output `user@host> set date ntp`
`21 Apr 17:22:02 ntpdate[3867]: step time server 172.17.27.46 offset 8.759252 sec`

- Options**
- *date-time*—Specify date and time in one of the following formats:
 - `YYYYMMDDHHMM.SS`
 - `"month DD, YYYY HH:MM(am | pm)"`
 - *ntp*—Configure the router to synchronize the current date and time setting with a Network Time Protocol (NTP) server.
 - *ntp-server*—(Optional) Specify the IP address of one or more NTP servers.
 - *source-address* *source-address*—(Optional) Specify the source address that is used by the router to contact the remote NTP server.

Required Privilege Level view

show cli

Syntax show cli

Release Information Command introduced before JUNOS Release 7.4.

Description Display configured CLI settings.

Sample Output

```
user@host> show cli
CLI complete-on-space set to on
CLI idle-timeout disabled
CLI restart-on-upgrade set to on
CLI screen-length set to 47
CLI screen-width set to 132
CLI terminal is 'vt100'
CLI is operating in enhanced mode
CLI timestamp disabled
CLI working directory is '/var/home/regress'
```

Required Privilege Level view

show cli authorization

Syntax show cli authorization

Release Information Command introduced before JUNOS Release 7.4.

Description Display the permissions for the current user.

Sample Output

```
user@host> show cli authorization
Current user: 'root' login: 'boojum' class '(root)'
Permissions:
admin          -- Can view user accounts
admin-control  -- Can modify user accounts
clear          -- Can clear learned network information
configure      -- Can enter configuration mode
control        -- Can modify any config
edit           -- Can edit full files
field          -- Special for field (debug) support
floppy         -- Can read and write from the floppy
interface      -- Can view interface config
interface-control -- Can modify interface config
network        -- Can access the network
reset          -- Can reset/restart interfaces and daemons
routing        -- Can view routing config
routing-control -- Can modify routing config
shell          -- Can start a local shell
snmp           -- Can view SNMP config
snmp-control   -- Can modify SNMP config
system         -- Can view system config
system-control -- Can modify system config
trace          -- Can view trace file settings
trace-control  -- Can modify trace file settings
view           -- Can view current values and statistics
maintenance    -- Can become the super-user
firewall       -- Can view firewall config
firewall-control -- Can modify firewall config
secret         -- Can view secret config
secret-control  -- Can modify secret config
```

Required Privilege Level view

show cli directory

Syntax show cli directory

Release Information Command introduced before JUNOS Release 7.4.

Description Display the current working directory.

Sample Output user@host> **show cli directory**
Current directory: /var/home/regress

Required Privilege Level view

show cli history

Syntax	show cli history <count>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Display a list of previous CLI commands.
Sample Output	<pre>user@host> show cli history 11:14:14 -- show arp 11:22:10 -- show cli authorization 11:27:12 -- show cli history</pre>
Options	none—Display all previous CLI commands. count—(Optional) Maximum number of commands to display.
Usage Guidelines	See “Displaying the JUNOS CLI Command and Word History” on page 32.
Required Privilege Level	view

Chapter 13

Summary of CLI Configuration Mode Commands

The following sections explain each of the command-line interface (CLI) configuration mode commands described in this book. The commands are organized alphabetically.

activate

Syntax	<code>activate (statement identifier)</code>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Remove the inactive: tag from a statement, effectively adding the statement or identifier back to the configuration. Statements or identifiers that have been activated take effect when you next issue the commit command.
Options	<p><i>identifier</i>—Identifier from which you are removing the inactive tag. It must be an identifier at the current hierarchy level.</p> <p><i>statement</i>—Statement from which you are removing the inactive tag. It must be a statement at the current hierarchy level.</p>
Usage Guidelines	See “Deactivating and Reactivating Statements and Identifiers” on page 79.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	deactivate

annotate

Syntax	<code>annotate statement "comment-string"</code>
Release Information	Command introduced before JUNOS Release 7.4.
Description	<p>Add comments to a configuration. You can add comments only at the current hierarchy level.</p> <p>Any comments you add appear only when you view the configuration by entering the show command in configuration mode or the show configuration command in operational mode.</p>
Options	<p><i>comment-string</i>—Text of the comment. You must enclose it in quotation marks. In the comment string, you can include the comment delimiters <code>/* */</code> or <code>#</code>. If you do not specify any, the comment string is enclosed with the <code>/* */</code> comment delimiters. If a comment for the specified <i>statement</i> already exists, it is deleted and replaced with the new comment.</p> <p><i>statement</i>—Statement to which you are attaching the comment.</p>
Usage Guidelines	See “Adding Comments in a JUNOS Configuration” on page 80.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	The description statement in the <i>JUNOS Network Interfaces Configuration Guide</i> .

commit

Syntax `commit <<at <"string">> <and-quit> <check> <comment <"comment-string">>
<confirmed> <display detail> <minutes> <synchronize<force>>`

Release Information Command introduced before JUNOS Release 7.4.

Description Commit the set of changes to the database and cause the changes to take operational effect.

Options `at <"string">`—(Optional) Save software configuration changes and activate the configuration at a future time, or upon reboot.

`string` is `reboot` or the future time to activate the configuration changes. Enclose the `string` value (including `reboot`) in quotation marks (" "). You can specify time in two formats:

- A time value in the form `hh:mm[:ss]` (hours, minutes, and optionally seconds)—Commit the configuration at the specified time, which must be in the future but before 11:59:59 PM on the day the `commit at` configuration command is issued. Use 24-hour time for the `hh` value; for example, `04:30:00` is 4:30:00 AM, and `20:00` is 8:00 PM. The time is interpreted with respect to the clock and time zone settings on the router.
- A date and time value in the form `yyy-mm-dd hh:mm[:ss]` (year, month, date, hours, minutes, and, optionally, seconds)—Commit the configuration at the specified day and time, which must be after the `commit at` command is issued. Use 24-hour time for the `hh` value. For example, `2003-08-21 12:30:00` is 12:30 PM on August 21, 2003. The time is interpreted with respect to the clock and time zone settings on the router.

For example, `commit at "18:00:00"`. For date and time, include both values in the same set of quotation marks. For example, `commit at "2005-03-10 14:00:00"`.

A *commit check* is performed when you issue the `commit at` configuration mode command. If the result of the check is successful, then the current user is logged out of configuration mode, and the configuration data is left in a read-only state. No other commit can be performed until the scheduled commit is completed.



NOTE: If JUNOS Software fails before the configuration changes become active, all configuration changes are lost.

You cannot issue the **commit at** configuration command when there is a pending reboot.

You cannot issue the **request system reboot** command once you schedule a commit operation for a specific time in the future.

You cannot commit a configuration when a scheduled commit is pending. For information about how to use the **clear** command to cancel a scheduled configuration, see the *JUNOS System Basics and Services Command Reference*.

and-quit—(Optional) Commit the configuration and, if the configuration contains no errors and the commit succeeds, exit from configuration mode.

check—(Optional) Verify the syntax of the configuration, but do not activate it.

comment <"comment-string"> —(Optional) Add a comment that describes the committed configuration. The comment can be as long as 512 bytes and must be typed on a single line. You cannot include a comment with the **commit check** command. Enclose *comment-string* in quotation marks (" "). For example, **commit comment "Includes changes recommended by SW Lab"**.

confirmed <minutes>—(Optional) Require that the commit be confirmed within the specified amount of time. To confirm a commit, enter either a **commit** or **commit check** command. If the commit is not confirmed within the time limit, the configuration rolls back automatically to the precommit configuration and a broadcast message is sent to all logged-in users. To show when a rollback is scheduled, enter the **show system commit** command. The allowed range is 1 through 65,535 minutes, and the default is 10 minutes.

display detail—(Optional) Monitors the commit process.

synchronize <force>—(Optional) If your router has two Routing Engines, you can manually direct one Routing Engine to synchronize its configuration with the other by issuing the **commit synchronize** command. The Routing Engine on which you execute this command (request Routing Engine) copies and loads its candidate configuration to the other (responding Routing Engine). Both Routing Engines then perform a syntax check on the candidate configuration file being committed. If no errors are found, the configuration is activated and becomes the current operational configuration on both Routing Engines. The **commit synchronize** command does not work if the responding Routing Engine has uncommitted configuration changes. However, you can enforce commit synchronization on the Routing Engines by using the **force** option. When you issue the **commit synchronize** command with the **force** option from one Routing Engine, the configuration sessions on the other Routing Engine will be terminated and its configuration synchronized with that on the Routing Engine from which you issued the command.



NOTE: When you issue the `commit synchronize` command, you must use the `apply-groups re0` and `re1` commands. For information about how to use groups, see “Applying a Configuration Group” on page 167.

The responding Routing Engine must use JUNOS Release 5.0 or later.

Usage Guidelines See “Verifying a JUNOS Configuration” on page 82, “Committing a JUNOS Configuration” on page 83, “Scheduling a JUNOS Commit Operation” on page 85, “Deactivating and Reactivating Statements and Identifiers” on page 79, “Monitoring the JUNOS Commit Process” on page 86, and “Adding a Comment to Describe the Committed Configuration” on page 87.

Required Privilege Level `configure`—To enter configuration mode.



NOTE: If you are using JUNOS Software in a Common Criteria environment, system log messages are created whenever a **secret** attribute is changed (for example, password changes or changes to the RADIUS shared secret). These changes are logged during the following configuration load operations:

`load merge`
`load replace`
`load override`
`load update`

For more information, see the *Secure Configuration Guide for Common Criteria and JUNOS FIPS*.

copy

Syntax	<code>copy</code> <i>existing-statement</i> to <i>new-statement</i>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Make a copy of an existing statement in the configuration.
Options	<i>existing-statement</i> —Statement to copy. <i>new-statement</i> —Copy of the statement.
Usage Guidelines	See “Copying a JUNOS Statement in the Configuration” on page 75.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.

deactivate

Syntax	deactivate (<i>statement</i> <i>identifier</i>)
Release Information	Command introduced before JUNOS Release 7.4.
Description	Add the inactive: tag to a statement, effectively commenting out the statement or identifier from the configuration. Statements or identifiers marked as inactive do not take effect when you issue the commit command.
Options	<p><i>identifier</i>—Identifier to which you are adding the inactive: tag. It must be an identifier at the current hierarchy level.</p> <p><i>statement</i>—Statement to which you are adding the inactive: tag. It must be a statement at the current hierarchy level.</p>
Usage Guidelines	See “Deactivating and Reactivating Statements and Identifiers” on page 79.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	activate, delete

delete

Syntax delete <*statement-path*> <*identifier*>

Release Information Command introduced before JUNOS Release 7.4.

Description Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it.

Deleting a statement or an identifier effectively “unconfigures” or disables the functionality associated with that statement or identifier.

If you do not specify *statement-path* or *identifier*, the entire hierarchy starting at the current hierarchy level is removed.

Options *statement-path*—(Optional) Path to an existing statement or identifier. Include this if the statement or identifier to be deleted is not at the current hierarchy level.

identifier—(Optional) Name of the statement or identifier to delete.

Usage Guidelines See “Deleting a Statement from a JUNOS Configuration” on page 73.

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

Related Topics deactivate

edit

Syntax	<code>edit <i>statement-path</i></code>
Release Information	Command introduced before JUNOS Release 7.4.
Description	<p>Move inside the specified statement hierarchy. If the statement does not exist, it is created.</p> <p>You cannot use the edit command to change the value of identifiers. You must use the set command.</p>
Options	<i>statement-path</i> —Path to the statement.
Usage Guidelines	See “Displaying the Current Configuration” on page 70.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	set

exit

Syntax	exit <configuration-mode>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Exit the current level of the statement hierarchy, returning to the level prior to the last edit command, or exit from configuration mode. The quit and exit commands are synonyms.
Options	<p>none—Return to the previous edit level. If you are at the top of the statement hierarchy, exit configuration mode.</p> <p>configuration-mode—(Optional) Exit from configuration mode.</p>
Usage Guidelines	See “Displaying the Current Configuration” on page 70.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	top, up

help

Syntax `help <(apropos string | reference <statement-name> | syslog <syslog-tag> | tip cli number | topic <word>)>`

Release Information Command introduced before JUNOS Release 7.4.

Description Display help about available configuration statements or general information about getting help.

Options `apropos string`—(Optional) Display statement names and help text that matches the string specified. If the string contains spaces, enclose it in quotation marks (" "). You can also specify a regular expression for the string, using standard UNIX-style regular expression syntax.

`reference <statement-name>`—(Optional) Display summary information for the statement. This information is based on summary descriptions that appear in the JUNOS configuration guides.

`syslog <syslog-tag>`—(Optional) Display information about system log messages.

`tip cli number`—(Optional) Display a tip about using the CLI. Specify the number of the tip you want to view.

`topic <word>`—(Optional) Display usage guidelines for a topic or configuration statement. This information is based on subjects that appear in the JUNOS configuration guides.

Entering the **help** command without an option provides introductory information about how to use the **help** command.

Usage Guidelines See “Getting Online Help” on page 25.

Required Privilege Level `configure`—To enter configuration mode.

insert

Syntax	insert <statement-path> identifier1 (before after) identifier2
Release Information	Command introduced before JUNOS Release 7.4.
Description	Insert an identifier in to an existing hierarchy.
Options	<p>after—Place <i>identifier1</i> after <i>identifier2</i>.</p> <p>before—Place <i>identifier1</i> before <i>identifier2</i>.</p> <p><i>identifier1</i>—Existing identifier.</p> <p><i>identifier2</i>—New identifier to insert.</p> <p><i>statement-path</i>—(Optional) Path to the existing identifier.</p>
Usage Guidelines	See “Inserting a New Identifier” on page 77.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.

load

Syntax load (factory-default | merge | override | patch | replace | set | update) (*filename* | terminal) <relative>

Release Information Command introduced before JUNOS Release 7.4.

Description Load a configuration from an ASCII configuration file, from terminal input, or from the factory default. Your current location in the configuration hierarchy is ignored when the load operation occurs.

Options **factory-default**—Loads the factory configuration. The factory configuration contains the manufacturer’s suggested configuration settings. The factory configuration is the router’s first configuration and is loaded when the router is first installed and powered on.

On J Series Services Routers, pressing and holding down the Config button on the router for 15 seconds causes the factory configuration to be loaded and committed. However, this operation deletes all other configurations on the router; using the **load factory-default** command does not.

filename—Name of the file to load. For information about specifying the filename, see “Specifying Filenames and URLs” on page 51.

merge—Combine the configuration that is currently shown in the CLI and the configuration in **filename**.

override—Discard the entire configuration that is currently shown in the CLI and load the entire configuration in **filename**. Marks every object as changed.

patch—Change part of the configuration and mark only those parts as changed.

replace—Look for a **replace** tag in **filename**, delete the existing statement of the same name, and replace it with the configuration in **filename**.

set—Merge a set of commands with an existing configuration. This option executes the configuration instructions line-by-line as they are stored in a file or from a terminal. The instructions can contain any configuration mode command, such as **set**, **edit**, **exit**, and **top**.

relative—(Optional) Use the **merge** or **replace** option without specifying the full hierarchy level.

terminal—Use the text you type at the terminal as input to the configuration. Type **Ctrl+d** to end terminal input.

update—Discard the entire configuration that is currently shown in the CLI, and load the entire configuration in **filename**. Marks changed objects only.



NOTE: If you are using JUNOS Software in a Common Criteria environment, system log messages are created whenever a **secret** attribute is changed (for example, password changes or changes to the RADIUS shared secret). These changes are logged during the following configuration load operations:

```
load merge
load replace
load override
load update
```

For more information, see the *Secure Configuration Guide for Common Criteria and JUNOS FIPS*.

Usage Guidelines See “Loading a Configuration from a File” on page 106.

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

quit

Syntax	quit <configuration-mode>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Exit the current level of the statement hierarchy, returning to the level prior to the last edit command, or exit from configuration mode. The quit and exit commands are synonyms.
Options	<p>none—Return to the previous edit level. If you are at the top of the statement hierarchy, exit configuration mode.</p> <p>configuration-mode—(Optional) Exit from configuration mode.</p>
Usage Guidelines	See “Displaying the Current Configuration” on page 70.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.
Related Topics	top, up

rename

Syntax `rename <statement-path> identifier1 to identifier2`

Release Information Command introduced before JUNOS Release 7.4.

Description Rename an existing configuration statement or identifier.

Options *identifier1*—Existing identifier to rename.

identifier2—New name of identifier.

statement-path—(Optional) Path to an existing statement or identifier.

Usage Guidelines See “Renaming an Identifier” on page 76.

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

replace

Syntax `replace pattern pattern1 with pattern2 <upto n>`

Release Information Command introduced in JUNOS Release 7.6.

Description Replace identifiers or values in a configuration.

Options *pattern1*—Text string or regular expression that defines the identifiers or values you want to match.

pattern2—Text string or regular expression that replaces the identifiers and values located with *pattern1*.

Juniper Networks uses standard UNIX-style regular expression syntax (as defined in POSIX 1003.2). If the regular expression contains spaces, operators, or wildcard characters, enclose the expression in quotation marks. Greedy qualifiers (match as much as possible) are supported. Lazy qualifiers (match as little as possible) are not.

upto *n*—Number of objects replaced. The value of *n* controls the total number of objects that are replaced in the configuration (not the total number of times the pattern occurs). Objects at the same hierarchy level (siblings) are replaced first. Multiple occurrences of a pattern within a given object are considered a single replacement. If you do not specify an **upto** option, all identifiers and values in the configuration that match *pattern1* are replaced.

Usage Guidelines See “Using Global Replace in a JUNOS Configuraton” on page 137.

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

rollback

Syntax `rollback <number | rescue>`

Release Information Command introduced before JUNOS Release 7.4.

Description Return to a previously committed configuration. The software saves the last 50 committed configurations, including the rollback number, date, time, and name of the user who issued the `commit` configuration command.

The currently operational JUNOS Software configuration is stored in the file `juniper.conf`, and the last three committed configurations are stored in the files `juniper.conf.1`, `juniper.conf.2`, and `juniper.conf.3`. These four files are located in the directory `/config`, which is on the router's flash drive. The remaining 46 previous committed configurations, the files `juniper.conf.4` through `juniper.conf.49`, are stored in the directory `/var/db/config`, which is on the router's hard disk.

During rollback, the configuration you specify is loaded from the associated file. Only objects in the rollback configuration that differ from the previously loaded configuration are marked as changed (equivalent to `load update`).

Options `none`—Return to the most recently saved configuration.

number—Configuration to return to. The range of values is from 0 through 49. The most recently saved configuration is number 0, and the oldest saved configuration is number 49. The default is 0.

rescue—Return to the rescue configuration.

Usage Guidelines See “Returning to a Previously Committed JUNOS Configuration” on page 100 and “Creating and Returning to a Rescue Configuration” on page 104

Required Privilege Level `rollback`—To roll back to configurations other than the one most recently committed.

run

Syntax `run command`

Release Information Command introduced before JUNOS Release 7.4.

Description Run a top-level CLI command without exiting from configuration mode.

Options *command*—CLI top-level command.

Required Privilege Level configure—To enter configuration mode.

save

Syntax `save filename`

Release Information Command introduced before JUNOS Release 7.4.

Description Save the configuration to an ASCII file. The contents of the current level of the statement hierarchy (and below) are saved, along with the statement hierarchy containing it. This allows a section of the configuration to be saved, while fully specifying the statement hierarchy.

When saving a file to a remote system, the software uses the **scp/ssh** protocol.

Options *filename*—Name of the saved file. You can specify a filename in one of the following ways:

- *filename*—File in the user's home directory (the current directory) on the local flash drive.
- *path/filename*—File on the local flash drive.
- */var/filename* or */var/path/filename*—File on the local hard disk.
- *a:filename* or *a:path/filename*—File on the local drive. The default path is */* (the root-level directory). The removable media can be in MS-DOS or UNIX (UFS) format.
- *hostname:/path /filename*, *hostname:filename*, *hostname:path/filename*, or *scp://hostname/path/filename*—File on an scp/ssh client. This form is not available in the worldwide version of JUNOS Software. The default path is the user's home directory on the remote system. You can also specify *hostname* as *username@hostname*.
- *ftp://hostname/path/ filename*—File on an FTP server. You can also specify *hostname* as *username @hostname* or *username :password @hostname*. The default path is the user's home directory. To specify an absolute path, the path must start with the string *%2F*; for example, *ftp://hostname/%2Fpath/filename*. To have the system prompt you for the password, specify *prompt* in place of the password. If a password is required, and you do not specify the password or *prompt*, an error message is displayed:

```
user@host> file copy ftp://username@ftp.hostname.net//filename
```

```
file copy ftp.hostname.net: Not logged in.
```

```
user@host> file copy ftp://username:prompt@ftp.hostname.net//filename
```

Password for *username@ftp.hostname.net*:

- *http://hostname/path/ filename*—File on a Hypertext Transfer Protocol (HTTP) server. You can also specify *hostname* as *username@hostname* or *username:password@hostname*. If a password is required and you omit it, you are prompted for it.

- `re0:/path/filename` or `re1:/path/filename`—File on a local Routing Engine.

Usage Guidelines See “Deactivating and Reactivating Statements and Identifiers” on page 79.

Required Privilege Level `configure`—To enter configuration mode.

set

Syntax `set <statement-path> identifier`

Release Information Command introduced before JUNOS Release 7.4.

Description Create a statement hierarchy and set identifier values. This is similar to `edit` except that your current level in the hierarchy does not change.

Options *identifier*—Name of the statement or identifier to set.

statement-path—(Optional) Path to an existing statement hierarchy level. If that hierarchy level does not exist, it is created.

Usage Guidelines See “Displaying the Current Configuration” on page 70.

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

Related Topics `edit`

show

Syntax `show <statement-path> <identifier>`

Release Information Command introduced before JUNOS Release 7.4.

Description Display the current configuration.

Options `none`—Display the entire configuration at the current hierarchy level.

identifier—(Optional) Display the configuration for the specified identifier.

statement-path—(Optional) Display the configuration for the specified statement hierarchy path.

Usage Guidelines See “Displaying the Current Configuration” on page 70.

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

Related Topics `show | display inheritance`

`show | display omit`

`show | display set`

`show | display set relative`

`show groups junos-defaults`

show | display inheritance

Syntax show | display inheritance <brief | defaults | terse>

Release Information Command introduced before JUNOS Release 7.4.

Description Show the inherited configuration data and information about the source group from which the configuration has been inherited.

Sample Output user@host# **show system ports | display inheritance defaults**
 ## ## 'console' was inherited from group 'junos-defaults'
 ## 'vt100' was inherited from group 'junos-defaults'
 ## console type vt100;

- Options**
- brief—Display brief output for the command.
 - defaults—Display the JUNOS Software defaults that have been applied to the configuration.
 - terse—Display terse output with inheritance details as inline comment.

Usage Guidelines For information about the show | display inheritance defaults command, see “Using JUNOS Defaults Groups” on page 183.

Required Privilege Level view

show | display omit

Syntax show | display omit

Release Information Command introduced in JUNOS Release 8.2.

Description Display configuration statements (including those marked as hidden by the `apply-flags omit` configuration statement).

Sample Output

```
user@host# show | display omit
system {
  apply-flags omit;
  login {
    message lengthy-login-message;
  }
}
```

Required Privilege Level view

show | display set

Syntax	show display set
Release Information	Command introduced before JUNOS Release 7.4.
Description	Display the configuration as a series of configuration mode commands required to re-create the configuration from the top level of the hierarchy as set commands
Sample Output	<pre>user@host# show display set set interfaces fe-0/0/0 unit 0 family inet address 192.168.1.230/24 set interfaces fe-0/0/0 unit 0 family iso set interfaces fe-0/0/0 unit 0 family mpls set interfaces fe-0/0/0 unit 1 family inet address 10.0.0.1/8 deactivate interfaces fe-0/0/0 unit 1</pre>
Usage Guidelines	See “Displaying set Commands from the Configuration” on page 93.
Required Privilege Level	view

show | display set relative

Syntax show | display set relative

Release Information Command introduced before JUNOS Release 7.4.

Description Display the configuration as a series of configuration mode commands required to re-create the configuration from the current hierarchy level.

Sample Output

```
[edit interfaces fe-0/0/0]
user@host# show
unit 0 {
  family inet {
    address 192.107.1.230/24;
  }
  family iso;
  family mpls;
}
inactive: unit 1 {
  family inet {
    address 10.0.0.1/8;
  }
}
user@host# show | display set relative
set unit 0 family inet address 192.107.1.230/24
set unit 0 family iso
set unit 0 family mpls
set unit 1 family inet address 10.0.0.1/8
deactivate unit 1
```

Usage Guidelines See “Displaying set Commands from the Configuration” on page 93.

Required Privilege Level view

show groups junos-defaults

Syntax show groups junos-defaults**Release Information** Command introduced before JUNOS Release 7.4.**Description** Display the full set of available preset statements from the JUNOS Software defaults group.

Sample Output user@host# **show groups junos-defaults**

```

groups {
  junos-defaults {
    applications {
      # File Transfer Protocol
      application junos-ftp {
        application-protocol ftp;
        protocol tcp;
        destination-port 21;
      }
      # Trivial File Transfer Protocol
      application junos-tftp {
        application-protocol tftp;
        protocol udp;
        destination-port 69;
      }
      # RPC port mapper on TCP
      application junos-rpc-portmap-tcp {
        application-protocol rpc-portmap;
        protocol tcp;
        destination-port 111;
      }
      # RPC port mapper on UDP
    }
  }
}

```

Usage Guidelines See “Using JUNOS Defaults Groups” on page 183.**Required Privilege Level** view

status

Syntax status

Release Information Command introduced before JUNOS Release 7.4.

Description Display the users currently editing the configuration.

Usage Guidelines See “Displaying Users Currently Editing the Configuration” on page 90.

Required Privilege Level configure—To enter configuration mode.

top

Syntax	<code>top <configuration-command></code>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Return to the top level of configuration command mode, which is indicated by the [edit] banner.
Options	<i>configuration-command</i> —(Optional) Issue configuration mode commands from the top of the hierarchy.
Usage Guidelines	See “Displaying the Current Configuration” on page 70.
Required Privilege Level	configure—To enter configuration mode.
Related Topics	exit up

up

Syntax up <number> <configuration-command>

Release Information Command introduced before JUNOS Release 7.4.

Description Move up one level in the statement hierarchy.

Options none—Move up one level in the configuration hierarchy.

configuration-command—(Optional) Issue configuration mode commands from a location higher in the hierarchy.

number—(Optional) Move up the specified number of levels in the configuration hierarchy.

Usage Guidelines See “Displaying the Current Configuration” on page 70.

Required Privilege Level configure—To enter configuration mode.

Related Topics exit
 top

update

Syntax update

Release Information Command introduced in JUNOS Release 7.5.

Description Update private candidate configuration with a copy of the most recently committed configuration, including your private changes.

Usage Guidelines See “Updating the Configure Private Configuration” on page 92.



NOTE: The `update` command is available only when you are in `configure private` mode.

wildcard delete

Syntax	wildcard delete <statement-path> <identifier> <regular-expression>
Release Information	Command introduced before JUNOS Release 7.4.
Description	<p>Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it.</p> <p>Deleting a statement or an identifier effectively “unconfigures” or disables the functionality associated with that statement or identifier.</p> <p>If you do not specify <i>statement-path</i> or <i>identifier</i>, the entire hierarchy starting at the current hierarchy level is removed.</p>
Options	<p><i>identifier</i>—(Optional) Name of the statement or identifier to delete.</p> <p><i>regular-expression</i>—(Optional) The pattern based on which you want to delete multiple items. When you use the <i>wildcard</i> command to delete related configuration items, the <i>regular-expression</i> must be the final statement.</p> <p><i>statement-path</i>—(Optional) Path to an existing statement or identifier. Include this if the statement or identifier to be deleted is not at the current hierarchy level.</p>
Usage Guidelines	See “Using Global Replace in a JUNOS Configuration” and “Using the upto Option” on page 141.
Required Privilege Level	configure—To enter configuration mode; other required privilege levels depend on where the statement is located in the configuration hierarchy.

Chapter 14

Summary of CLI Operational Mode Commands

The following sections explain each of the command-line interface (CLI) operational mode commands described in this book. The commands are organized alphabetically.

configure

Syntax	configure <exclusive> <private>
Release Information	Command introduced before JUNOS Release 7.4.
Description	Enter configuration mode. When this command is entered without any optional keywords, everyone can make configuration changes and commit all changes made to the configuration.
Options	<p>exclusive—(Optional) Lock the candidate configuration for as long as you remain in configuration mode, allowing you to make changes without interference from other users. Other users can enter and exit configuration mode, but they cannot change the configuration.</p> <p>private—(Optional) Allow multiple users to edit different parts of the configuration at the same time and to commit only their own changes, or to roll back without interfering with one another's changes. You cannot commit changes in configure private mode when another user is in configure exclusive mode.</p>
Additional Information	For more information about the different methods of entering configuration mode and the restrictions that apply, see the <i>JUNOS System Basics Configuration Guide</i> .
Required Privilege Level	configure
Related Topics	show configuration
List of Sample Output	configure on page 244
Output Fields	When you enter this command, you are placed in configuration mode and the system prompt changes from <i>hostname></i> to <i>hostname#</i> .
configure	<pre> user@host> configure Entering configuration mode [edit] user@host# </pre>

file

Syntax file <archive | checksum | compare | copy | delete | list | rename | show | source address>

Release Information Command introduced before JUNOS Release 7.4.

Description Archive files from the device, copy files to and from the router, calculate the file checksum, compare files, delete a file from the device, list files on the device, rename a file, show file contents, or show the local address to initiate a connection.

Additional Information See “Viewing Files and Directories on a Device Running JUNOS Software” on page 47. See also the *JUNOS System Basics and Services Command Reference*.

Required Privilege Level maintenance

help

Syntax	<code>help < (apropos <i>string</i> reference <<i>statement-name</i>> syslog <<i>syslog-tag</i>> tip cli <i>number</i> topic <<i>word</i>>)></code>
Release Information	Command introduced before JUNOS Release 7.4. apropos option added in JUNOS Release 8.0.
Description	Display help about available operational commands, configuration statements, or general information about getting help. Entering the help command without an option provides introductory information about how to use the help and ? commands.
Options	<p>apropos <i>string</i>—(Optional) Display command names and help text that matches the string specified. If the string contains spaces, enclose it in quotation marks (" "). You can also specify a regular expression for the string, using standard UNIX-style regular expression syntax.</p> <p>reference <<i>statement-name</i>>—(Optional) Display summary information for a configuration statement. This information is based on summary descriptions that appear in the JUNOS configuration guides.</p> <p>syslog <<i>syslog-tag</i>>—(Optional) Display information about system log messages.</p> <p>tip cli <i>number</i>—(Optional) Display a tip about using the CLI. Specify the number of the tip you want to view.</p> <p>topic <<i>word</i>>—(Optional) Display usage guidelines for a topic or configuration statement. This information is based on subjects that appear in the JUNOS configuration guides.</p>
Additional Information	See “Getting Online Help” on page 25.
Required Privilege Level	None

| (pipe)

Syntax | (compare | count | display (changed | commit-scripts | detail | display set | inheritance | omit | xml) | except *pattern* | find *pattern* | hold | last *lines* | match *pattern* | no-more | request message (all | *account@terminal*) resolve <full-names> | save *filename* | trim *columns*)

Release Information Command introduced before JUNOS Release 7.4.
display commit-scripts option added in JUNOS Release 7.4.

Description Filter the output of an operational mode or a configuration mode command.

Options compare (filename | rollback *n*)—(Configuration mode only, and only with the show command) Compare configuration changes with another configuration file.

count—Display the number of lines in the output.

display—Display additional information about the configuration contents.

- changed—Tag changes with junos:changed attribute (XML only).
- commit-scripts—(Configuration mode only) Display all statements that are in a configuration, including statements that were generated by transient changes. For more information, see the *JUNOS Configuration and Diagnostic Automation Guide*.
- detail—(Configuration mode only) Display configuration data detail.
- inheritance <brief | default | groups | terse>—(Configuration mode only) Display inherited configuration data and source group.
- omit—(Configuration mode only) Display configuration statements omitted by the apply-flags omit configuration statement.
- set—Display the configuration as a series of configuration mode commands required to re-create the configuration.
- xml—(Operational mode only) Display the command output as JUNOScript (Extensible Markup Language [XML]) tags.

except *pattern*—Ignore text matching a regular expression when searching the output. If the regular expression contains spaces, operators, or wildcard characters, enclose it in quotation marks.

find *pattern*—Display the output starting at the first occurrence of text matching a regular expression. If the regular expression contains spaces, operators, or wildcard characters, enclose it in quotation marks (" ").

last *lines*—Display the last number of lines you want to view from the end of the configuration. However, when the number of lines requested is less than the number of lines that the screen length setting permits you to display, JUNOS returns as many

lines as permitted by the screen length setting. For more information on using the **last *lines*** option, see “Displaying Output Beginning with the Last Entries” on page 122.

hold—Hold text without exiting the **-More-** prompt.

match *pattern*—Search for text matching a regular expression. If the regular expression contains spaces, operators, or wildcard characters, enclose it in quotation marks.

no-more—Display output all at once rather than one screen at a time.

resolve—Convert IP addresses into Domain Name System (DNS) names. Truncates to fit original size unless **full-names** is specified. To prevent the names from being truncated, use the **full-names** option.

request message (all | *account@terminal*)—Display command output on the terminal of a specific user logged in to your router, or on the terminals of all users logged in to your router.

save *filename*—Save the output to a file or URL. For information about specifying the filename, see “Specifying Filenames and URLs” on page 51.

trim *columns*—Trim specified number of columns from the start line.

Additional Information “Filtering Command Output” on page 117 and “Displaying the Current Configuration” on page 70.

request

Syntax request <chassis | ipsec switch | message | mpls | routing-engine | security | services | system | flow-collector | support information>

Release Information Command introduced before JUNOS Release 7.4.

Description Stop or reboot router components, switch between primary and backup components, display messages, and display system information.



NOTE: If your router contains two Routing Engines and you want to shut the power off to the router or remove a Routing Engine, you must first halt the backup Routing Engine (if it has been upgraded) and then the master Routing Engine. To halt a Routing Engine, issue the **request system halt** command. You can also halt both Routing Engines at the same time by issuing the **request system halt both-routing-engines** command.

If you want to reboot a router that has two Routing Engines, reboot the backup Routing Engine (if you have upgraded it) and then the master Routing Engine.



CAUTION: Halt the backup Routing Engine before you remove it or shut off the power to the router; otherwise you might need to reinstall JUNOS Software.



NOTE: If you reboot the TX Matrix router, all the T640 master Routing Engines connected to the TX Matrix router reboot. If you halt both Routing Engines on a TX Matrix router, all the T640s Routing Engines connected to the TX Matrix router are also halted.



NOTE: If you insert a Flexible PIC Concentrator (FPC) into your router, you may need to issue the **request chassis fpc** command (or press the **online** button) to bring the FPC online. This applies to FPCs in M20, M40, M40e, M160, M320, and T Series routers. For command usage, see the **request chassis fpc** command description in the *JUNOS System Basics and Services Command Reference*.

Additional Information Most **request** commands are discussed in the *JUNOS System Basics and Services Command Reference*. The following **request** commands are discussed in the *JUNOS Interfaces Command Reference*: **request ipsec switch** and **request services**.

Required Privilege Level maintenance

restart

Syntax	<pre>restart <adaptive-services audit-process chassis-control class-of-service disk-monitoring dynamic-flow-capture ecc-error-logging event-processing firewall interface-control ipsec-key-management kernel-replication l2-learning l2tp-service lACP mib-process pgcp-service pgm pic-services-logging ppp pppoe protected-system-domain-service redundancy-interface-process remote-operations root-system-domain-service routing <logical-system <i>logical-system-name</i>> sampling service-deployment services pgcp gateway <i>gateway-name</i> sbc-configuration-process snmp> <gracefully immediately soft></pre>
Syntax (TX Matrix Router)	<pre>restart <adaptive-services audit-process chassis-control class-of-service disk-monitoring dynamic-flow-capture ecc-error-logging event-processing firewall interface-control ipsec-key-management kernel-replication l2-learning l2tp-service lACP link-management mib-process pgm pic-services-logging ppp pppoe redundancy-interface-process remote-operations routing <logical-system <i>logical-system-name</i>> sampling service-deployment snmp> <all-chassis all-lcc lcc <i>number</i> scc> <gracefully immediately soft></pre>
Syntax (TX Matrix Plus Router)	<pre>restart <adaptive-services audit-process chassis-control class-of-service disk-monitoring dynamic-flow-capture ecc-error-logging event-processing firewall interface-control ipsec-key-management kernel-replication l2-learning l2tp-service lACP link-management mib-process pgm pic-services-logging ppp pppoe redundancy-interface-process remote-operations routing <logical-system <i>logical-system-name</i>> sampling service-deployment snmp> <all-chassis all-lcc all-sfc lcc <i>number</i> sfc <i>number</i>> <gracefully immediately soft></pre>
Syntax (J Series Router)	<pre>restart <adaptive-services audit-process chassis-control class-of-service dhcp dialer-services dlsw event-processing firewall interface-control ipsec-key-management isdn-signaling l2ald l2-learning l2tp-service mib-process network-access-service pgm ppp pppoe remote-operations routing <logical-system <i>logical-system-name</i>> sampling service-deployment snmp usb-control web-management> <gracefully immediately soft></pre>
Release Information	<p>Command introduced before JUNOS Release 7.4.</p> <p>dynamic-flow-capture option added in JUNOS Release 7.4.</p> <p>dlsw option added in JUNOS Release 7.5.</p> <p>event-processing option added in JUNOS Release 7.5.</p> <p>ppp option added in JUNOS Release 7.5.</p> <p>l2ald option added in JUNOS Release 8.0.</p> <p>link-management option added in Release 8.0.</p> <p>pgcp-service option added in JUNOS Release 8.4.</p> <p>sbc-configuration-process option added in JUNOS Release 9.5.</p>

`services pgcp gateway` option added in JUNOS Release 9.6
`sfc` and `all-sfc` options introduced for the TX Matrix Router in JUNOS Release 9.6.

Description Restart a JUNOS Software process.



CAUTION: Never restart a software process unless instructed to do so by a customer support engineer. A restart might cause the router to drop calls and interrupt transmission, resulting in possible loss of data.

Options none—Same as `gracefully`.

adaptive-services—(Optional) Restart the configuration management process that manages the configuration for stateful firewall, Network Address Translation (NAT), intrusion detection services (IDS), and IP Security (IPsec) services on the Adaptive Services PIC.

audit-process—(Optional) Restart the RADIUS accounting process.

chassis-control—(Optional) Restart the chassis management process.

class-of-service—(Optional) Restart the class-of-service (CoS) process, which controls the router's CoS configuration.

dhcp—(J Series router only) (Optional) Restart the software process for a Dynamic Host Configuration Protocol (DHCP) server. A DHCP server allocates network IP addresses and delivers configuration settings to client hosts without user intervention.

dialer-services—(J Series router only) (Optional) Restart the ISDN dial out process.

disk-monitoring—(Optional) Restart disk monitoring, which checks the health of the hard disk drive on the Routing Engine.

dlsu—(J Series router only) (Optional) Restart the data link switching (DLSw) service.

dynamic-flow-capture—(Optional) Restart the dynamic flow capture (DFC) process, which controls DFC configurations on Monitoring Services III PICs.

ecc-error-logging—(Optional) Restart the error checking and correcting (ECC) process, which logs ECC parity errors in memory on the Routing Engine.

event-processing—(Optional) Restart the event process (eventd).

firewall—(Optional) Restart the firewall management process, which manages firewall configuration.

interface-control—(Optional) Restart the interface process, which controls the router's physical interface devices and logical interfaces.

ipsec-key-management—(Optional) Restart the IPSEC key management process.

isdn-signaling—(J Series router only) (Optional) Restart the ISDN signaling process, which initiates ISDN connections.

kernel-replication—(Optional) Restart the kernel replication process, which replicates the state of the backup Routing Engine when graceful Routing Engine switchover is configured.

l2-learning—(Optional) Restart the Layer 2 address flooding and learning process.

l2tp-service—(Optional) (M10, M10i, and M7i routers only) Restart the Layer 2 Tunneling Protocol (L2TP) process, which establishes L2TP tunnels and Point-to-Point Protocol (PPP) sessions through L2TP tunnels.

lacc—(Optional) Restart the link aggregation control protocol process.

link-management—(TX Matrix and TX Matrix Plus routers only) (Optional) Restart the Link Management Protocol (LMP) process, which establishes and maintains LMP control channels.

mib-process—(Optional) Restart the Management Information Base (MIB) II process, which provides the router's MIB II agent.

network-access-service—(J Series router only) (Optional) Restart the network access process, which provides the router's Challenge Handshake Authentication Protocol (CHAP) authentication service.

pgcp-service—(Optional) Restart the pgcpd service process running on the Routing Engine. This option does not restart pgcpd processes running on mobile station Physical Interface Cards (PICs). To restart pgcpd processes running on mobile station PICs, use the **services pgcp gateway** option.

pgm—(Optional) Restart the process that implements the Pragmatic General Multicast (PGM) protocol for assisting in the reliable delivery of multicast packets.

pic-services-logging—(Optional) Restart the logging process for some Physical Interface Cards (PICs). With this process, also known as fsad (the file system access daemon), PICs send special logging information to the Routing Engine for archiving on the hard disk.

ppp—(Optional) Restart the Point-to-Point Protocol (PPP) process.

pppoe—(Optional) Restart the Point-to-Point Protocol over Ethernet (PPPoE) process.

protected-system-domain-service—(Optional) Restart the Protected System Domain (PSD) process.

redundancy-interface-process—(Optional) Restart the ASP redundancy process.

remote-operations—(Optional) Restart the remote operations process, which provides the ping and traceroute MIBs.

root-system-domain-service—(Optional) Restart the Root System Domain (RSD) service.

routing <logical-system *logical-system-name*>—(Optional) Restart the routing protocol process, which controls the routing protocols that run on the router and maintains the routing tables. Optionally, restart the routing protocol process for the specified logical system only.

sampling—(Optional) Restart the sampling process, which performs packet sampling and cflowd export.

service-development—(Optional) Restart the service deployment service process.

services pgcp gateway *gateway-name*—(Optional) Restart the pgcpd process for a specific BGP running on an MS-PIC. This option does not restart the pgcpd process running on the Routing Engine. To restart the pgcpd process on the Routing Engine, use the **pgcp-service** option.

snmp—(Optional) Restart the SNMP process, which provides the router's SNMP master agent.

usb-control—(J Series router only) (Optional) Restart the USB control process.

web-management—(J Series router only) (Optional) Restart the Web management process.

all-chassis—(TX Matrix and TX Matrix Plus routers only) (Optional) Restart the software process on all chassis.

all-lcc—(TX Matrix and TX Matrix Plus routers only) (Optional) For a TX Matrix router, restart the software process on all T640 routers connected to the TX Matrix router. For a TX Matrix Plus router, restart the software process on all T1600 routers connected to the TX Matrix Plus router.

all-sfc—(TX Matrix Plus routers only) (Optional) For a TX Matrix Plus router, restart the software processes for the TX Matrix Plus router (or switch-fabric chassis).

lcc *number*—(TX Matrix and TX Matrix Plus routers only) (Optional) For a TX Matrix router, restart the software process for a specific T640 router that is connected to the TX Matrix router. For a TX Matrix Plus router, restart the software process for a specific T1600 router that is connected to the TX Matrix Plus router. Replace *number* with a value from 0 through 3.

scc—(TX Matrix routers only) Restart the software process on the TX Matrix router (or switch-card chassis).

sfc *number*—(TX Matrix Plus routers only) Restart the software process on the TX Matrix Plus router (or switch-fabric chassis). Replace *number* with 0.

gracefully—(Optional) Restart the software process.

immediately—(Optional) Immediately restart the software process.

soft—(Optional) Reread and reactivate the configuration without completely restarting the software processes. For example, BGP peers stay up and the routing table

stays constant. Omitting this option results in a graceful restart of the software process.

Required Privilege Level	reset
List of Sample Output	restart interfaces on page 255
Output Fields	When you enter this command, you are provided feedback on the status of your request.
restart interfaces	<pre>user@host> restart interfaces interfaces process terminated interfaces process restarted</pre>

set

Syntax `set <statement-path> identifier`

Release Information Command introduced before JUNOS Release 7.4.

Description Create a statement hierarchy and set identifier values. This is similar to **edit** except that your current level in the hierarchy does not change.

Options *identifier*—Name of the statement or identifier to set.

statement-path—(Optional) Path to an existing statement hierarchy level. If that hierarchy level does not exist, it is created.

Usage Guidelines See “Displaying the Current Configuration” on page 70.

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

Related Topics `edit`

show

Syntax `show <statement-path> <identifier>`

Release Information Command introduced before JUNOS Release 7.4.

Description Display the current configuration.

Options `none`—Display the entire configuration at the current hierarchy level.

identifier—(Optional) Display the configuration for the specified identifier.

statement-path—(Optional) Display the configuration for the specified statement hierarchy path.

Usage Guidelines See “Displaying the Current Configuration” on page 70.

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

Related Topics `show | display inheritance`

`show | display omit`

`show | display set`

`show | display set relative`

`show groups junos-defaults`

Part 5

Index

- Index on page 261
- Index of Statements and Commands on page 269

Index

Symbols

!	
in interface names.....	137
regular expression operator.....	118
" ", configuration group wildcards.....	152, 171
#, comments in configuration statements.....	xxvi, 80
\$	
regular expression operator.....	118
(), in syntax descriptions.....	xxvi
regular expression operator.....	118
*	
in interface names.....	137
regular expression operator.....	139
wildcard character.....	152, 171
+	
in statement lists.....	73
regular expression operator.....	139
. (period)	
regular expression operator.....	139
/* */, comment delimiters.....	80
< >, in syntax descriptions.....	xxv
?	
regular expression operator.....	152, 171
wildcard.....	152, 171
[], in configuration statements.....	xxvi
\	
in interface names.....	137
regular expression operator.....	118
wildcard characters.....	152, 171
^	
regular expression operator.....	118
{ }, in configuration statements.....	xxvi
specifying statements.....	110
(pipe).....	247
command output, filtering.....	247
(pipe), in syntax descriptions.....	xxvi, 247

A

access privilege levels	
entering configuration mode.....	68
activate command.....	210
usage guidelines.....	62
active configuration.....	5

addresses

machine name.....	12
annotate command.....	63, 211
usage guidelines.....	80
apply-groups statement.....	187
usage guidelines.....	148, 167
apply-groups-except statement.....	188
usage guidelines.....	169
authorization	<i>See</i> permissions

B

braces, in configuration statements.....	xxvi
brackets	
angle, in syntax descriptions.....	xxv
square, in configuration statements.....	xxvi

C

candidate configuration.....	5
clear command	
usage guidelines.....	37
CLI	
command completion.....	194
command history.....	32
displaying.....	207
comparing configuration versions.....	102
configuration mode	
description.....	62
navigation commands, table.....	7
current working directory	
displaying.....	206
setting.....	195
date	
setting.....	203
editing command line.....	135
idle timeout, setting.....	196
keyboard sequences.....	136
overview.....	3
permissions, displaying.....	205
prompt strings.....	128
prompt, setting.....	197
restart, after software upgrade.....	198
screen length, setting.....	199
screen width, setting.....	200
settings, displaying.....	204

terminal type, setting.....	201	commit synchronize command.....	212
timestamp.....	128	usage guidelines.....	79
timestamp, setting.....	202	commit display detail command	
tutorial.....	9	usage guidelines.....	86
type checking.....	112	committing configuration	
users, monitoring.....	45	and exiting configuration mode.....	84
word history.....	33	basic.....	83
working directory.....	128	confirmation required.....	84
command history		logging message about.....	87
operational mode.....	32	monitoring.....	86
command output		scheduling for later.....	85
configuration details.....	95	synchronizing on Routing Engines.....	113
configuration, comparing files.....	119	compare command.....	247
end of, displaying from.....	122	usage guidelines.....	102
filtering		compare filter.....	119
comparing configuration versions.....	102	completing partial command entry.....	194
number of lines, counting.....	121	configuration	
pagination, preventing.....	123	activating.....	100
regular expressions		adding comments.....	80
first match, displaying from.....	122	candidate.....	5
matching output, displaying.....	123	committing.....	83
nonmatching output, ignoring.....	121	and exiting configuration mode.....	84
retaining.....	122	confirmation required.....	84
saving to a file.....	124	logging message about.....	87
sending to users.....	124	monitoring process.....	86
XML format, displaying.....	121	scheduling for later.....	85
command shell.....	3	synchronizing on Routing Engines.....	113
command-summary.....	233	comparing with previous.....	102
show groups junos-defaults.....	237	deleting	
show display omit.....	234	statements.....	73
show display set.....	235	displaying	
show display set relative.....	236	details.....	95
commands		edit command, using.....	70
completion.....	29, 129	global replacement.....	137
configure.....	129	groups configuration groups <i>See</i> configuration	
filenames, specifying.....	51	groups	
help about commands.....	26, 27	locking.....	91
history.....	32, 33	merging current and new.....	107
options.....	41	modifying.....	70
overview.....	37	previous, displaying.....	101
URLs, specifying.....	51	replacing.....	106
comments		saving to file.....	105
adding to configuration file.....	80	storage of previous.....	99
comments, in configuration statements.....	xxvi	configuration files	
commit and-quit command		filename, specifying.....	51
usage guidelines.....	84	saving to files.....	105
commit at command		URL, specifying.....	51
usage guidelines.....	85	configuration groups	
commit command.....	212	applying.....	148, 167
usage guidelines.....	63, 83	creating.....	147, 165
commit comment command		example configuration groups.....	174
usage guidelines.....	87	inheritance model.....	146, 164
commit confirmed command		inherited values.....	170
usage guidelines.....	84	interface parameters.....	155, 157, 176, 178
commit scripts.....	8	nested groups.....	149, 167
		overview.....	146

- peer entities.....158, 179
 - re0, re1 groups.....147, 165
 - regional configurations.....160, 181
 - sets of statements.....154, 175
 - wildcards.....152, 161, 171, 182
 - configuration mode, CLI.....83
 - +73
 - command completion.....29
 - commands
 - activate.....62
 - annotate.....63
 - commit.....63
 - copy.....63
 - deactivate.....63
 - delete.....63
 - edit.....63
 - exit.....63
 - extension.....63
 - help.....63
 - insert.....63
 - load.....63
 - paste.....63
 - quit.....63
 - rollback.....22, 63
 - run.....63
 - save.....64
 - set.....64
 - show.....64
 - status.....64
 - top.....64
 - up.....64
 - update.....64
 - configuration hierarchy, description.....66
 - description.....62
 - entering.....68
 - example16
 - exiting.....69
 - global replacement.....137
 - identifier, description.....64
 - locking.....91
 - statement
 - container.....66
 - description.....64
 - leaf.....66
 - switching to operational mode.....11
 - users editing configuration
 - displaying.....90
 - multiple simultaneous users.....88
 - configuration mode, entering.....244
 - configuration statements
 - adding comments about.....80
 - deleting.....73
 - inheriting from groups.....154, 175
 - overviews.....72
 - structure and components.....110
 - configure command.....244
 - names and addresses.....12
 - usage guidelines.....38, 68
 - configure exclusive command
 - usage guidelines.....91
 - container hierarchy *See* hierarchy
 - conventions
 - text and syntax.....xxv
 - copy command.....215
 - usage guidelines.....38, 39, 63
 - count command.....247
 - count filter.....121
 - curly braces, in configuration statements.....xxvi
 - current working directory
 - displaying.....206
 - setting.....195
 - cursor, moving.....136
 - customer support.....xxvii
 - contacting JTAC.....xxvii
- D**
- data types, CLI.....112
 - date
 - setting from CLI.....203
 - deactivate command.....216
 - usage guidelines.....63
 - default configuration group.....162, 183
 - delete command.....217
 - usage guidelines.....63, 73
 - directories
 - working, displaying.....206
 - disable statement
 - usage guidelines.....79
 - display detail command
 - usage guidelines.....95
 - display inheritance command
 - usage guidelines.....170
 - display set command
 - usage guidelines.....93
 - display xml filter.....121
 - documentation set
 - comments on.....xxvi
- E**
- edit command.....218
 - usage guidelines.....63
 - editing command line.....135
 - Emacs keyboard sequences.....135
 - environment settings, CLI
 - command completion.....129
 - displaying.....129
 - example configuration.....129
 - idle timeout.....128
 - prompt string.....128

screen dimensions.....	127, 130
software upgrade, restarting after.....	129
terminal type.....	128
timestamp.....	128
working directory.....	128
except command.....	247
except filter.....	121
exit command.....	219
from configuration mode.....	11, 12
usage guidelines.....	63, 69
exit configuration-mode command.....	219
usage guidelines.....	69
extension command	
usage guidelines.....	63

F

file command.....	245
usage guidelines.....	38, 39, 47
filenames, specifying in commands.....	51
files	
listing.....	48
saving command output to.....	124
saving configurations to files.....	105
viewing.....	47
find command.....	247
find filter.....	122
font conventions.....	xxv
FreeBSD UNIX kernel.....	4

G

groups statement.....	189
usage guidelines.....	147, 165

H

help apropos command	
usage guidelines.....	27
help command.....	220, 246
usage guidelines.....	27, 63
help reference command	
usage guidelines.....	27
help tip cli command	
usage guidelines.....	29
history, CLI commands	
displaying.....	207
operational mode.....	32, 33
hold command.....	247
hold filter.....	122

I

icons defined, notice.....	xxiv
----------------------------	------

identifiers	
inserting in sequential lists.....	77
renaming.....	76
specifying.....	110
idle timeout	
user, setting.....	196
idle timeout values	
CLI sessions.....	128
ignore filter.....	121
inheritance model, configuration groups.....	146, 164
inherited values, configuration groups.....	170
insert command.....	221
usage guidelines.....	63, 77
interface	
configuration example.....	16
interface names	
conventions.....	46
interfaces	
media parameters.....	155, 157, 176, 178
issuing relative configuration commands.....	76

J

J-Web graphical user interface (GUI).....	8
juniper-ais configuration group	
usage guidelines.....	147, 166
JUNOS FIPS software environment.....	8
junos-defaults configuration group.....	237
displaying.....	162, 183, 233, 237
JUNOScript Application Programming Interface (API).....	8

K

keyboard sequences	
editing command line.....	135

L

last command.....	247
last filter.....	122
load command.....	222
usage guidelines.....	63
load merge command	
usage guidelines.....	107
load override command	
usage guidelines.....	106
load set command	
usage guidelines.....	107
locking configuration.....	91
logical interfaces	
unit numbers.....	46

M

manuals
 comments on.....xxvi
 match command.....247
 match filter.....123
 monitor command.....37

N

names
 wildcard161, 182
 naming conventions, interface.....46
 nested configuration groups.....149, 167
 no-more command.....247, 248
 no-more filter.....123
 notice icons defined.....xxiv

O

operational mode, CLI
 command history.....32
 command overview.....37
 switching to configuration mode.....11
 users, monitoring.....45
 word history.....33
 operators
 used in regular expressions.....118

P

parentheses, in syntax descriptions.....xxvi
 partial command entry, completing.....194
 paste command
 usage guidelines.....63
 peer entities.....158, 179
 permissions, CLI, displaying.....205
 ping command
 usage guidelines.....37
 pipe (|)
 command output, filtering.....247
 processes
 managing.....54
 restarting.....251
 programs
 managing.....54
 prompt
 setting to display in CLI.....197
 to restart.....198
 prompt strings
 CLI.....128

Q

quit command.....39, 224
 usage guidelines.....63

R

re0 configuration group.....147, 165
 re1 configuration group.....147, 165
 redrawing screen.....136
 redundancy
 synchronize Routing Engines.....79
 regional configurations.....160, 181
 regular expression operators.....118
 in operational mode commands.....118
 regular expressions
 first match, displaying from.....122
 matching output, displaying.....123
 nonmatching output, ignoring.....121
 relative option.....107
 rename command.....225
 usage guidelines.....76
 renaming identifiers.....76
 replace command.....226
 usage guidelines.....137
 replace option.....106
 request command.....249
 usage guidelines.....38
 request message filter.....124
 request system configuration rescue delete
 command.....104
 request system configuration rescue save
 command.....104
 request system halt command.....58
 request system logout pid pid_number command.....91
 request system reboot command.....58
 resolve command.....247
 restart command.....251
 usage guidelines.....38
 restart routing command.....57
 restarting
 after software upgrade.....129, 198
 software processes.....251
 rollback command.....22, 227
 usage guidelines.....63
 Routing Engines
 synchronizing configuration.....113
 run command.....228
 usage guidelines.....63

S

save command.....229, 247
 usage guidelines.....64, 105
 screen
 dimensions.....127, 130
 redrawing.....136
 screen length, setting.....199
 screen width, setting.....200
 set cli complete-on-space command.....194
 usage guidelines.....129

set cli directory command.....	195
usage guidelines.....	128
set cli idle-timeout command.....	196
usage guidelines.....	128
set cli prompt command.....	197
usage guidelines.....	128
set cli restart-on-upgrade command.....	198
usage guidelines.....	129
set cli screen-length command.....	199
usage guidelines.....	127, 130
set cli screen-width command.....	200
set cli terminal command.....	201
usage guidelines.....	128
set cli timestamp command.....	202
usage guidelines.....	128
set command.....	70
configuration mode.....	231, 256
usage guidelines.....	64
set date command.....	203
set option.....	107
show cli authorization command.....	205
show cli command.....	204
usage guidelines.....	129
show cli directory command.....	206
show cli history command.....	207
usage guidelines.....	32
show command	
configuration mode.....	232, 257
usage guidelines.....	64
show groups junos-defaults	
command-summary.....	237
show groups junos-defaults command.....	237
usage guidelines.....	162, 183
show system processes extensive command.....	54
output, table.....	56
show version command	
JUNOS Software.....	52, 53
show display inheritance command.....	233
show display inheritance defaults command	
usage guidelines.....	162, 183
show display omit	
command-summary.....	234
show display omit command.....	234
show display set	
command-summary.....	235
show display set command.....	235
usage guidelines.....	93
show display set relative	
command-summary.....	236
show display set relative command.....	236
usage guidelines.....	93
software upgrade	
restarting after.....	198
ssh command	
usage guidelines.....	38

status command.....	238
usage guidelines.....	64, 90
storing previous configurations.....	99
support, technical <i>See</i> technical support	
symbol.....	124
synchronizing Routing Engines.....	79
syntax conventions.....	xxv

T

technical support	
contacting JTAC.....	xxvii
telnet command	
usage guidelines.....	38
terminal screen length, setting.....	199
terminal screen width, setting.....	200
terminal type.....	128
setting.....	201
test command	
usage guidelines.....	37
timeout, user, setting.....	196
timestamp, CLI output, setting.....	202
top command.....	239
usage guidelines.....	64, 76
traceroute command	
usage guidelines.....	37
trim command.....	247
TX Matrix router	
configuration groups.....	147, 165
configuration groups example.....	150, 168
type checking, CLI.....	112

U

UNIX operating system.....	3, 4
UNIX shell.....	4
up command.....	240
usage guidelines.....	64, 76
update command.....	241
usage guidelines.....	64, 92
updating configure private configuration.....	92
upgrade, restarting after.....	129
upgrading software.....	129
prompt to restart after.....	198
URLs, specifying in commands.....	51
user accounts	
configuration example.....	12
user timeout, setting.....	196
users	
CLI permissions, displaying.....	205
editing configuration	
displaying.....	90
multiple simultaneous users.....	88
of CLI, monitoring.....	45

W

- wildcard characters.....152, 171
- wildcard command.....242
- wildcard delete command
 - usage guidelines.....142
- wildcard names.....152, 161, 171, 182
- word history
 - operational mode.....33
- working directory
 - displaying.....206
 - setting.....195

X

- XML format
 - displaying command output in.....121

Index of Statements and Commands

Symbols

| (pipe).....247

A

activate command.....210
annotate command.....63, 211
apply-groups statement187
apply-groups-except statement188

C

commit command.....212
compare command.....247
configure command.....244
copy command.....215

D

deactivate command.....216
delete command.....217

E

edit command.....218
exit command.....219

F

file command.....245

G

groups statement.....189

H

help command.....220, 246

I

insert command.....221

L

load command.....222

N

no-more command.....247, 248

Q

quit command.....39, 224

R

rename command.....225
replace command.....226
request command.....249
restart command.....251
rollback command.....22, 227
run command.....228

S

save command.....229, 247
set cli complete-on-space command.....194
set cli directory command.....195
set cli idle-timeout command.....196
set cli prompt command.....197
set cli restart-on-upgrade command.....198
set cli screen-length command.....199
set cli screen-width command.....200
set cli terminal command.....201
set cli timestamp command.....202
set command
 configuration mode.....231, 256
set date command.....203
show cli authorization command.....205
show cli command.....204
show cli directory command.....206
show cli history command.....207
show command
 configuration mode.....232, 257
show groups junos-defaults command.....237
show | display inheritance command.....233
show | display omit command.....234

show display set command.....	235
show display set relative command.....	236
status command.....	238

T

top command.....	239
------------------	-----

U

up command.....	240
update command.....	241

W

wildcard command.....	242
-----------------------	-----