

## Sample PWP - Unregister after work done



Modified: 2019-09-11



Juniper Networks, Inc.  
1133 Innovation Way  
Sunnyvale, California 94089  
USA  
408-745-2000  
[www.juniper.net](http://www.juniper.net)

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, 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.

*Sample PWP - Unregister after work done*

Copyright © 2019 Juniper Networks, Inc. All rights reserved.

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

#### YEAR 2000 NOTICE

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

#### END USER LICENSE AGREEMENT

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

# Table of Contents

	About the Documentation . . . . .	xiii
	Documentation and Release Notes . . . . .	xiii
	Using the Examples in This Manual . . . . .	xiii
	Merging a Full Example . . . . .	xiv
	Merging a Snippet . . . . .	xiv
	Documentation Conventions . . . . .	xv
	Documentation Feedback . . . . .	xvii
	Requesting Technical Support . . . . .	xvii
	Self-Help Online Tools and Resources . . . . .	xviii
	Creating a Service Request with JTAC . . . . .	xviii
<b>Part 1</b>	<b>test</b>	
<b>Chapter 1</b>	<b>test . . . . .</b>	<b>3</b>
	Viewing the Configuration . . . . .	3
	Displaying the Current Junos OS Configuration . . . . .	3
	Example: Displaying the Current Junos OS Configuration . . . . .	5
	Displaying Additional Information About the Junos OS Configuration . . . . .	6
	Displaying set Commands from the Junos OS Configuration . . . . .	8
	Example: Displaying set Commands from the Configuration . . . . .	9
	Example: Displaying Required set Commands at the	
	Current Hierarchy Level . . . . .	9
	Example: Displaying set Commands with the match Option . . . . .	10
	Configuration Files Overview . . . . .	11
	Understanding Configuration Files . . . . .	11
	Configuration File Terms . . . . .	11
	Understanding How the Junos OS Configuration Is Stored . . . . .	12
	Autoinstallation of Configuration Files . . . . .	13
	Understanding Autoinstallation of Configuration Files . . . . .	13
	Typical Uses for Autoinstallation . . . . .	13
	Autoinstallation Configuration Files and IP Addresses . . . . .	13
	Typical Autoinstallation Process on a New Device . . . . .	14
	Configuring Autoinstallation of Configuration Files (CLI Procedure) . . . . .	15
	Factory Default Configuration . . . . .	17
	Reverting to the Default Factory Configuration . . . . .	17
	Reverting to the Default Factory Configuration for the EX Series Switch . . . . .	18
	Reverting to the Factory-Default Configuration Using the EX Series	
	Switch LCD Panel . . . . .	19
	Reverting to the EX Series Switch Factory-Default Configuration Using	
	the request system zeroize Command . . . . .	20

Reverting to the EX Series Switch Factory-Default Configuration Using the load factory-default Command . . . . .	20
Reverting to the Factory-Default Configuration Using the Factory Reset/Mode button on EX2300 and EX3400 Switches . . . . .	21
CLI Operational Mode Overview . . . . .	23
Overview of Junos OS CLI Operational Mode Commands . . . . .	23
CLI Command Categories . . . . .	23
Commonly Used Operational Mode Commands . . . . .	24
Junos OS Operational Mode Commands That Combine Other Commands . . . . .	26
Understanding the brief, detail, extensive, and terse Options of Junos OS Operational Commands . . . . .	26
Interface Naming Conventions Used in the Junos OS Operational Commands . . . . .	27
Physical Part of an Interface Name . . . . .	27
Logical Part of an Interface Name . . . . .	28
Channel Identifier Part of an Interface Name . . . . .	28
Using Wildcard Characters in Interface Names . . . . .	29
Using Operational Commands to Monitor a Device . . . . .	29
Using the Junos OS CLI Command Completion . . . . .	30
Controlling the Scope of an Operational Mode Command . . . . .	31
Operational Mode Commands on a TX Matrix Router or TX Matrix Plus Router . . . . .	32
Examples of Routing Matrix Command Options . . . . .	32
Monitoring Who Uses the Junos OS CLI . . . . .	34
Viewing Files and Directories on a Device Running Junos OS . . . . .	34
Directories on the Device . . . . .	35
Listing Files and Directories . . . . .	35
Specifying Filenames and URLs . . . . .	38
Displaying Junos OS Information . . . . .	39
Managing Programs and Processes Using Junos OS Operational Mode Commands . . . . .	41
Showing Software Processes . . . . .	41
Restarting the Junos OS Process . . . . .	43
Stopping Junos OS . . . . .	44
Rebooting Junos OS . . . . .	45
Using the Junos OS CLI Comment Character # for Operational Mode Commands . . . . .	45
Using Comments in Junos OS Operational Mode Commands . . . . .	46
Displaying the Junos OS CLI Command and Word History . . . . .	47
Online Help in the CLI . . . . .	47
Getting Online Help from the Junos OS Command-Line Interface . . . . .	47
Getting Help About Commands . . . . .	48
Getting Help About a String in a Statement or Command . . . . .	49
Getting Help About Configuration Statements . . . . .	50
Getting Help About System Log Messages . . . . .	50
Junos OS CLI Online Help Features . . . . .	50
Help for Omitted Statements . . . . .	50
Using CLI Command Completion . . . . .	51

Using Command Completion in Configuration Mode . . . . .	51
Displaying Tips About CLI Commands . . . . .	51
CLI Explorer Overview . . . . .	52
CLI Configuration Mode Overview . . . . .	53
Understanding Junos OS CLI Configuration Mode . . . . .	53
Configuration Mode Commands . . . . .	54
Configuration Statements and Identifiers . . . . .	55
Configuration Statement Hierarchy . . . . .	57
Entering and Exiting the Junos OS CLI Configuration Mode . . . . .	59
Issuing Relative Junos OS Configuration Mode Commands . . . . .	61
Using Command Completion in Configuration Mode . . . . .	62
Notational Conventions Used in Junos OS Configuration Hierarchies . . . . .	64
Backing Up Configurations to an Archive Site . . . . .	65
Configuring the Transfer of the Currently Active Configuration to an Archive Site . . . . .	65
Configuring the Periodic Transfer of the Active Configuration to an Archive Site . . . . .	65
Configuring the Transfer of the Currently Active Configuration When a Configuration Is Committed . . . . .	66
Configuring Archive Sites for the Transfer of Active Configuration Files . . . . .	66
Modifying the Configuration for a Device . . . . .	67
Displaying Users Currently Editing the Junos OS Configuration . . . . .	68
Modifying the Junos OS Configuration . . . . .	68
Adding Junos OS Configuration Statements and Identifiers . . . . .	69
Deleting a Statement from a Junos OS Configuration . . . . .	70
Example: Deleting a Statement from the Junos OS Configuration . . . . .	71
Copying a Junos OS Statement in the Configuration . . . . .	73
Example: Copying a Statement in the Junos Configuration . . . . .	73
Example: Re-Using Configuration . . . . .	75
Inserting a New Identifier in a Junos OS Configuration . . . . .	81
Example: Inserting a New Identifier in a Junos Configuration . . . . .	81
Renaming an Identifier in a Junos OS Configuration . . . . .	85
Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration . . . . .	85
Example: Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration . . . . .	85
Using Global Replace in the Junos OS Configuration . . . . .	87
Common Regular Expressions to Use with the replace Command . . . . .	88
Example: Using Global Replace in a Junos OS Configuration—Using the \n Back Reference . . . . .	89
Example: Using Global Replace in a Junos OS Configuration—Replacing an Interface Name . . . . .	91
Example: Using Global Replace in a Junos OS Configuration—Using the upto Option . . . . .	93
Using Regular Expressions to Delete Related Items from a Junos OS Configuration . . . . .	95
Adding Comments in a Junos OS Configuration . . . . .	96
Adding Comments in the CLI . . . . .	97
Adding Comments in a File . . . . .	98

Example: Including Comments in a Junos OS Configuration by Using the CLI .....	98
CLI Overview .....	101
Introducing the Junos OS Command-Line Interface .....	101
Key Features of the CLI .....	102
Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies .....	103
Junos OS CLI Command Modes .....	103
CLI Command Hierarchy .....	104
Configuration Statement Hierarchy .....	104
Moving Among Hierarchy Levels .....	105
Other Tools to Configure and Monitor Devices Running Junos OS .....	106
Configuring Junos OS in a FIPS Environment .....	106
Getting Started: A Quick Tour of the CLI .....	107
Getting Started with the Junos OS Command-Line Interface .....	107
Switching Between Junos OS CLI Operational and Configuration Modes ..	109
Using Keyboard Sequences to Move Around and Edit the Junos OS CLI ...	111
Configuring a User Account on a Device Running Junos OS .....	112
Using the CLI Editor in Configuration Mode .....	114
Checking the Status of a Device Running Junos OS .....	116
Rolling Back Junos OS Configuration Changes .....	119
Configuring a Routing Protocol .....	120
Shortcut .....	120
Longer Configuration .....	121
Making Changes to a Routing Protocol Configuration .....	123
CLI Environment Settings .....	126
Controlling the Junos OS CLI Environment .....	126
Setting the Terminal Type .....	127
Setting the CLI Prompt .....	127
Setting the CLI Directory .....	127
Setting the CLI Timestamp .....	128
Setting the Idle Timeout .....	128
Setting the CLI to Prompt After a Software Upgrade .....	128
Setting Command Completion .....	128
Displaying CLI Settings .....	129
Setting the Junos OS CLI Screen Length and Width .....	129
Setting the Screen Length .....	129
Setting the Screen Width .....	129
Example: Controlling the CLI Environment .....	129
Example: Enabling Configuration Breadcrumbs .....	136
Configure Command Overview .....	137
Forms of the configure Command .....	138
Using the configure Command .....	139
Using the configure exclusive Command .....	140
Updating the configure private Configuration .....	141
Using Configuration Groups to Quickly Configure Devices .....	142
Understanding Junos OS Configuration Groups .....	143
Configuration Groups Overview .....	143
Inheritance Model .....	143

Configuring Configuration Groups . . . . .	144
Creating a Junos OS Configuration Group . . . . .	144
Applying a Junos OS Configuration Group . . . . .	146
Example: Creating and Applying Junos OS Configuration Groups . . . . .	147
Example: Creating and Applying Configuration Groups on a TX Matrix Router . . . . .	148
Disabling Inheritance of a Junos OS Configuration Group . . . . .	149
Using the junos-defaults Configuration Group . . . . .	151
Using Wildcards with Configuration Groups . . . . .	152
Improving Commit Time When Using Configuration Groups . . . . .	155
Example: Configuring Sets of Statements with Configuration Groups . . . . .	155
Example: Configuring Interfaces Using Configuration Groups . . . . .	156
Example: Configuring a Consistent IP Address for the Management Interface Using Configuration Groups . . . . .	159
Example: Configuring Peer Entities Using Configuration Groups . . . . .	160
Example: Establishing Regional Configurations Using Configuration Groups . . . . .	162
Example: Configuring Wildcard Configuration Group Names . . . . .	163
Example: Referencing the Preset Statement from the Junos OS defaults Group . . . . .	164
Example: Viewing Default Statements That Have Been Applied to the Configuration . . . . .	165
Setting Up Routing Engine Configuration Groups . . . . .	166
Using Conditions to Apply Configuration Groups . . . . .	168
Example: Configuring Conditions for Applying Configuration Groups . . . . .	168
Committing a Configuration . . . . .	171
Junos OS Commit Model for Configurations . . . . .	171
Committing a Junos OS Configuration and Exiting Configuration Mode . . . . .	172
Committing a Junos OS Configuration . . . . .	172
Commit Operation When Multiple Users Configure the Software . . . . .	174
Commit Preparation and Activation Overview . . . . .	175
Committing Junos OS Configurations in Two Steps: Preparation and Activation . . . . .	177
Activating a Junos OS Configuration but Requiring Confirmation . . . . .	178
Scheduling a Junos OS Commit Operation . . . . .	180
Monitoring the Junos OS Commit Process . . . . .	181
Adding a Comment to Describe the Committed Configuration . . . . .	182
Junos OS Batch Commits Overview . . . . .	183
Aggregation and Error Handling . . . . .	183
Example: Configuring Batch Commit Server Properties . . . . .	183
Backing Up the Committed Configuration on the Alternate Boot Drive . . . . .	192





# List of Figures

Part 1	test	
Chapter 1	test .....	3
	Figure 1: EX Series Switch LCD Panel .....	19
	Figure 2: Commands That Combine Other Commands .....	26
	Figure 3: Command Output Options .....	27
	Figure 4: Restarting a Process .....	44
	Figure 5: Configuration Mode Hierarchy of Statements .....	57
	Figure 6: Replacement by Object .....	94
	Figure 7: Monitoring and Configuring Routers .....	102
	Figure 8: Committing a Configuration .....	104
	Figure 9: Configuration Statement Hierarchy Example .....	105
	Figure 10: Confirm a Configuration .....	179



# List of Tables

	<b>About the Documentation . . . . .</b>	<b>xiii</b>
	Table 1: Notice Icons . . . . .	xv
	Table 2: Text and Syntax Conventions . . . . .	xvi
<b>Part 1</b>	<b>test</b>	
<b>Chapter 1</b>	<b>test . . . . .</b>	<b>3</b>
	Table 3: Configuration File Terms . . . . .	11
	Table 4: Commonly Used Operational Mode Commands . . . . .	24
	Table 5: Wildcard Characters for Specifying Interface Names . . . . .	29
	Table 6: Directories on the Device . . . . .	35
	Table 7: show system process extensive Command Output Fields . . . . .	42
	Table 8: Summary of Configuration Mode Commands . . . . .	54
	Table 9: Configuration Mode Top-Level Statements . . . . .	56
	Table 10: Common Regular Expressions to Use with the replace Command . . . .	88
	Table 11: Replacement Examples . . . . .	89
	Table 12: CLI Configuration Mode Navigation Commands . . . . .	105
	Table 13: CLI Keyboard Shortcuts . . . . .	111
	Table 14: Forms of the configure Command . . . . .	138



# About the Documentation

- Documentation and Release Notes on page xiii
- Using the Examples in This Manual on page xiii
- Documentation Conventions on page xv
- Documentation Feedback on page xvii
- Requesting Technical Support on page xvii

## Documentation and Release Notes

---

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

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

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

## Using the Examples in This Manual

---

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

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

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

## Merging a Full Example

To merge a full example, follow these steps:

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

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

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

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

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

## Merging a Snippet

To merge a snippet, follow these steps:

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

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

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

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

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

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

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

For more information about the **load** command, see [CLI Explorer](#).

## Documentation Conventions

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

Table 1: Notice Icons







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

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

Table 2: Text and Syntax Conventions

Convention	Description	Examples
<b>Bold text like this</b>	Represents text that you type.	To enter configuration mode, type the <b>configure</b> command:  user@host> <b>configure</b>
Fixed-width text like this	Represents output that appears on the terminal screen.	user@host> <b>show chassis alarms</b>  No alarms currently active
<i>Italic text like this</i>	<ul style="list-style-type: none"> <li>Introduces or emphasizes important new terms.</li> <li>Identifies guide names.</li> <li>Identifies RFC and Internet draft titles.</li> </ul>	<ul style="list-style-type: none"> <li>A policy <i>term</i> is a named structure that defines match conditions and actions.</li> <li><i>Junos OS CLI User Guide</i></li> <li>RFC 1997, <i>BGP Communities Attribute</i></li> </ul>
<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@# <b>set system domain-name</b> <i>domain-name</i>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none"> <li>To configure a stub area, include the <b>stub</b> statement at the [edit protocols <b>ospf area area-id</b>] hierarchy level.</li> <li>The console port is labeled <b>CONSOLE</b>.</li> </ul>
< > (angle brackets)	Encloses optional keywords or variables.	<b>stub</b> <default-metric <i>metric</i> >;
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	<b>broadcast</b>   <b>multicast</b>  ( <i>string1</i>   <i>string2</i>   <i>string3</i> )
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	<b>rsvp { # Required for dynamic MPLS only</b>
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	<b>community name members</b> [ <b>community-ids</b> ]
Indentation and braces ( { } )	Identifies a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

## GUI Conventions



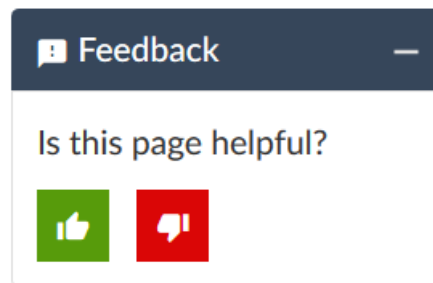
Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
<b>Bold text like this</b>	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none"> <li>In the Logical Interfaces box, select <b>All Interfaces</b>.</li> <li>To cancel the configuration, click <b>Cancel</b>.</li> </ul>
<b>&gt;</b> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select <b>Protocols&gt;Ospf</b> .

## Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the [Juniper Networks TechLibrary](#) site, and do one of the following:



- Click the thumbs-up icon if the information on the page was helpful to you.
- Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.
- E-mail—Send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net). Include the document or topic name, URL or page number, and software version (if applicable).

## Requesting Technical Support

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

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <https://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: <https://www.juniper.net/customers/support/>
- Search for known bugs: <https://prsearch.juniper.net/>
- Find product documentation: <https://www.juniper.net/documentation/>
- Find solutions and answer questions using our Knowledge Base: <https://kb.juniper.net/>
- Download the latest versions of software and review release notes: <https://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum: <https://www.juniper.net/company/communities/>
- Create a service request online: <https://myjuniper.juniper.net>

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

## Creating a Service Request with JTAC

You can create a service request with JTAC on the Web or by telephone.

- Visit <https://myjuniper.juniper.net>.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

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

## PART 1

# test

- [test on page 3](#)



## CHAPTER 1

# test

- [Viewing the Configuration on page 3](#)
- [Configuration Files Overview on page 11](#)
- [Autoinstallation of Configuration Files on page 13](#)
- [Factory Default Configuration on page 17](#)
- [CLI Operational Mode Overview on page 23](#)
- [Using Operational Commands to Monitor a Device on page 29](#)
- [Online Help in the CLI on page 47](#)
- [CLI Configuration Mode Overview on page 53](#)
- [Backing Up Configurations to an Archive Site on page 65](#)
- [Modifying the Configuration for a Device on page 67](#)
- [CLI Overview on page 101](#)
- [Getting Started: A Quick Tour of the CLI on page 107](#)
- [CLI Environment Settings on page 126](#)
- [Configure Command Overview on page 137](#)
- [Using Configuration Groups to Quickly Configure Devices on page 142](#)
- [Committing a Configuration on page 171](#)

## Viewing the Configuration

---

The **show** configuration mode command displays the current configuration for a device running Junos OS.

- [Displaying the Current Junos OS Configuration on page 3](#)
- [Example: Displaying the Current Junos OS Configuration on page 5](#)
- [Displaying Additional Information About the Junos OS Configuration on page 6](#)
- [Displaying set Commands from the Junos OS Configuration on page 8](#)

## Displaying the Current Junos OS Configuration

To display the current configuration for a device running Junos OS, 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, 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: 2018-07-18 11:21:58 PDT by echen  
version 8.3
```

If you have omitted a required statement at a specific hierarchy level, when you issue the **show** command in configuration mode, a message indicates which statement is missing. If 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'  
    }  
  }  
}
```

When you issue the **show configuration** command with the **| display set** pipe option to view the configuration as **set** commands, those portions of the configuration that you do not have permissions to view are substituted with the text **ACCESS-DENIED**.

Unsupported statements included in the CLI configuration are displayed with the “unsupported” text in the configuration. For example, if a statement is configured on an unsupported platform, the CLI displays a message that the statement is ignored in the configuration because it is configured on an unsupported platform. When you issue the **show** command with the **| display xml** option, you can see the **unsupported="unsupported"** attribute for configuration that is unsupported.

The “unsupported” attribute included in text configuration or XML configuration is provided to scripts when the **unsupported="unsupported"** attribute is included in the **<get-configuration>** RPC call.

## Example: Displaying the Current Junos OS Configuration

The following example shows how you can display the current Junos configuration.

Set a configuration:

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

To display the current configuration:

```
[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 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;
}
```

Set and commit a 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
```

Display the last committed configuration:

```
user@host> show configuration
## Last commit: 2018-08-10 11:21:58 PDT by user
version 8.3
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/0 {
        hello-interval 5;
      }
    }
  }
}
```

## Displaying Additional Information About the Junos OS 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 "18.2R1 [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 "$ABC123"; # SECRET-DATA
}
##
## name-server: DNS name servers
## require: system
##
name-server {
    ##
    ## name-server: DNS name server address
    ##
    208.197.1.0;
}
login {
    ##
    ## class: User name (login)
    ## match: ^[[:alnum:]]_-$
    ##
    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;
    }
  }
}
...
```

## Displaying set Commands from the Junos OS 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.

To display the configuration as a series of configuration mode commands, which are 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 9](#)
- [Example: Displaying Required set Commands at the Current Hierarchy Level on page 9](#)
- [Example: Displaying set Commands with the match Option on page 10](#)

### 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
```

## Configuration Files Overview

A configuration file stores the complete configuration of a device. The active (running) configuration is the operational file of the device. The candidate configuration is the working copy storing configuration updates.

- [Understanding Configuration Files on page 11](#)
- [Understanding How the Junos OS Configuration Is Stored on page 12](#)

## Understanding Configuration Files

A configuration file stores the complete configuration of a network device. The current configuration of a device is called the active configuration. You can alter this current configuration and you can also return to a previous configuration or to a rescue configuration.

Juniper Networks Junos OS saves the 50 most recently committed configuration files on a device so that you can return to a previous configuration. The configuration files are named:

- **juniper.conf.gz**—The current active configuration.
- **juniper.conf.1.gz** to **juniper.conf.49.gz**—Rollback configurations.

To make changes to the configuration file, you must use configuration mode in the CLI. When making changes to a configuration file, you are viewing and changing the candidate configuration file. The candidate configuration allows you to make configuration changes without causing operational changes to the active configuration or causing potential damage to your current network operations. Once you commit the changes made to the candidate configuration, the system updates the active configuration.

## Configuration File Terms

**Table 3: Configuration File Terms**

Term	Definition
active configuration	Current committed configuration of a device.
candidate configuration	Working copy of the configuration that allows users to make configurational changes without causing any operational changes until this copy is committed.
configuration group	Group of configuration statements that can be inherited by the rest of the configuration.
commit a configuration	Check configuration for proper syntax, activate and mark as the current configuration file running on the device.
configuration hierarchy	Junos OS configuration consists of a hierarchy of statements. There are two types of statements: Container statements, which contain other statements, and leaf statements, which do not contain other statements. All the container and leaf statements together form the configuration hierarchy.

Table 3: Configuration File Terms (continued)

Term	Definition
default configuration	Default configuration contains the initial values set for each configuration parameter when a device is shipped.
rescue configuration	Well-known configuration that recovers a device from a configuration that denies management access. You set a current committed configuration to be the rescue configuration through the CLI.
roll back a configuration	Return to a previously committed configuration.

- See Also**
- *Uploading a Configuration File*
  - *Reverting to the Rescue Configuration*
  - *Uploading a Configuration File (CLI Procedure)*
  - *Reverting to the Rescue Configuration for the EX Series Switch*

## Understanding How the Junos OS 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, which is the current operational version and the default configuration that the system returns to if you roll back to a previous configuration. The oldest saved configuration is version 49.

By default, Junos OS saves the current configuration and three previous versions of the committed configuration on the CompactFlash card. The currently operational Junos OS configuration is stored in the file **juniper.conf.gz**, and the last three committed configurations are stored in the files **juniper.conf.1.gz**, **juniper.conf.2.gz**, and **conf.3.gz**. These four files are stored on the device's CompactFlash card in the directory **/config**.

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.

- See Also**
- *Using Junos OS to Specify the Number of Configurations Stored on the CompactFlash Card*

- [Returning to the Most Recently Committed Junos OS Configuration](#)
- [Returning to a Previously Committed Junos OS Configuration](#)
- [Loading a Configuration from a File or the Terminal](#)

## Autoinstallation of Configuration Files

---

Autoinstallation is the automatic configuration of devices over the network without manual intervention, or without any need for any configuration.

- [Understanding Autoinstallation of Configuration Files on page 13](#)
- [Configuring Autoinstallation of Configuration Files \(CLI Procedure\) on page 15](#)

## Understanding Autoinstallation of Configuration Files

Autoinstallation is the automatic configuration of a device over the network from a preexisting configuration file that you create and store on a configuration server—typically a Trivial File Transfer Protocol (TFTP) server. You can use autoinstallation to configure new devices automatically and to deploy multiple devices from a central location in the network.

You enable autoinstallation so that the switches in your network implement autoinstallation when they are powered on. To configure autoinstallation, you specify a configuration server, an autoinstallation interface, and a protocol for IP address acquisition.



**NOTE:** The QFX5200 switches only work with HTTP for autoinstallation. TFTP and FTP protocols are not supported.

This topic describes:

- [Typical Uses for Autoinstallation on page 13](#)
- [Autoinstallation Configuration Files and IP Addresses on page 13](#)
- [Typical Autoinstallation Process on a New Device on page 14](#)

### Typical Uses for Autoinstallation

---

Typical uses for autoinstallation of the software include:

- To deploy and update multiple devices from a central location in the network.
- To update a device—Autoinstallation occurs when a device that has been manually configured for autoinstallation is powered on.

### Autoinstallation Configuration Files and IP Addresses

---

For the autoinstallation process to work, you must store one or more host-specific or default configuration files on a configuration server in the network and have a service available—typically Dynamic Host Configuration Protocol (DHCP)—to assign an IP address to the switch.

You can set up the following configuration files for autoinstallation on the switch:

- **network.conf**—Default configuration file for autoinstallation, in which you specify IP addresses and associated hostnames for devices on the network.
- **switch.conf**—Default configuration file for autoinstallation with a minimum configuration sufficient for you to telnet to the device and configure it manually.
- **hostname.conf**—Host-specific configuration file for autoinstallation on a device that contains all the configuration information necessary for the switch. In the filename, *hostname* is replaced with the hostname assigned to the switch.

If the server with the autoinstallation configuration file is not on the same LAN segment as the new device, or if a specific device is required by the network, you must configure an intermediate device directly attached to the new switch, through which the new switch can send TFTP, Boot Protocol (BOOTP), and Domain Name System (DNS) requests. In this case, you specify the IP address of the intermediate device as the location to receive TFTP requests for autoinstallation.

### Typical Autoinstallation Process on a New Device

---

When the device configured for autoinstallation is powered on, it performs the following autoinstallation tasks:

1. The device sends out DHCP or BOOTP requests on each connected interface simultaneously to obtain an IP address.

If a DHCP server responds to these requests, it provides the device with some or all of the following information:

- An IP address and subnet mask for the autoinstallation interface.
- The location of the (typically) TFTP server, Hypertext Transfer Protocol (HTTP) server, or FTP server on which the configuration file is stored.
- The name of the configuration file to be requested from the TFTP server.
- The IP address or hostname of the TFTP server.

If the DHCP server provides the server's hostname, a DNS server must be available on the network to resolve the name to an IP address.

- The IP address of an intermediate device if the configuration server is on a different LAN segment from the device.
2. After the device acquires an IP address, the autoinstallation process on the device attempts to download a configuration file in the following ways:
    - a. If the DHCP server specifies the host-specific configuration file **hostname.conf**, the device uses that filename in the TFTP server request. The autoinstallation process on the new device makes three unicast TFTP requests for **hostname.conf**. If these attempts fail, the device broadcasts three requests to any available TFTP server for the file.
    - b. If the device does not locate a **hostname.conf** file, the autoinstallation process sends three unicast TFTP requests for a **network.conf** file that contains the device's



hostname-to-IP-address mapping information. If these attempts fail, the device broadcasts three requests to any available TFTP server for the file.

- c. If the device fails to find a **network.conf** file that contains a hostname entry for the device, the autoinstallation process sends out a DNS request and attempts to resolve the device's IP address to a hostname.
  - d. If the device determines its hostname, it sends a TFTP request for the **hostname.conf** file.
  - e. If the device is unable to map its IP address to a hostname, it sends TFTP requests for the default configuration file **device.conf**. The TFTP request procedure is the same as for the **network.conf** file.
3. After the device locates a configuration file on a TFTP server, the autoinstallation process downloads the file, installs the file on the device, and commits the configuration.

- See Also**
- *Connecting and Configuring an EX Series Switch (CLI Procedure)*
  - *Connecting and Configuring an EX Series Switch (J-Web Procedure)*
  - *Configuration Files Terms*

## Configuring Autoinstallation of Configuration Files (CLI Procedure)

Autoinstallation is the automatic configuration of a device over the network from a pre-existing configuration file that you create and store on a configuration server—typically a Trivial File Transfer Protocol (TFTP) server. You can use autoinstallation to automatically deploy multiple devices from a central location in the network.

To specify autoinstallation to run when you power on a device already installed in your network, you can enable it by specifying one or more interfaces, protocols, and configuration servers to be used for autoinstallation.

Before you explicitly enable and configure autoinstallation on the device, perform these tasks as needed for your network's configuration:

- Have a service available—typically Dynamic Host Configuration Protocol (DHCP)—to assign an IP address to the device.
- Configure a DHCP server on your network to meet your network requirements. You can configure a switch to operate as a DHCP server.
- Create one of the following configuration files, and store it on a TFTP server (or HTTP server or FTP server) in the network:
  - A host-specific file with the name **hostname.conf** for each device undergoing autoinstallation. Replace **hostname** with the name of a device. The **hostname.conf**

file typically contains all the configuration information necessary for the device with this hostname.

- A default configuration file named **device.conf** with the minimum configuration necessary to enable you to telnet into the new device for further configuration.
- Physically attach the device to the network using a Gigabit Ethernet port.
- If you configure the DHCP server to provide only the TFTP server hostname, add an IP address-to-hostname mapping entry for the TFTP server to the DNS database file on the Domain Name System (DNS) server in the network.
- If the device is not on the same network segment as the DHCP server (or other device providing IP address resolution), configure an existing device as an intermediate device to receive TFTP and DNS requests and forward them to the TFTP server and the DNS server. You must configure the LAN or serial interface on the intermediate device with the IP addresses of the hosts providing TFTP and DNS services. Connect this interface to the device.
- If you are using **hostname.conf** files for autoinstallation, you must also complete the following tasks:
  - Configure the DHCP server to provide a **hostname.conf** filename to each device. Each device uses its **hostname.conf** filename to request a configuration file from the TFTP server. Copy the necessary **hostname.conf** configuration files to the TFTP server.
  - Create a default configuration file named **network.conf**, and copy it to the TFTP server. This file contains IP-address-to-hostname mapping entries. If the DHCP server does not send a **hostname.conf** filename to a new device, the device uses **network.conf** to resolve its hostname based on its IP address.

Alternatively, you can add the IP-address-to-hostname mapping entry for the device to a DNS database file.

The device uses the hostname to request a **hostname.conf** file from the TFTP server.

To configure autoinstallation:

1. Specify the URL address of one or more servers from which to obtain configuration files.

```
[edit system]
```

```
user@host# set autoinstallation configuration-servers tftp://tftpconfig.example.com
```



**NOTE:** You can also use an FTP address, for example,  
**ftp://user:password@sftpconfig.example.com.**

2. Configure one or more Ethernet interfaces to perform autoinstallation and one or two procurement protocols for each interface. The switch uses the protocols to send a request for an IP address for the interface:

```
[edit system]
```

```
user@host# set autoinstallation interfaces ge-0/0/0 bootp
```

To verify autoinstallation, from the CLI, enter the **show system autoinstallation status** command.

```
user@host> show system autoinstallation status

Autoinstallation status:
Master state: Active
Last committed file: None
Configuration server of last committed file: 10.25.100.1
Interface:
  Name: ge-0/0/0
  State: Configuration Acquisition
  Acquired:
    Address: 192.168.124.75
    Hostname: host-ge-000
    Hostname source: DNS
    Configuration filename: device-ge-000.conf
    Configuration filename server: 10.25.100.3
  Address acquisition:
    Protocol: DHCP Client
    Acquired address: None
    Protocol: RARP Client
    Acquired address: None
Interface:
  Name: ge-0/0/1
  State: None
  Address acquisition:
    Protocol: DHCP Client
    Acquired address: None
    Protocol: RARP Client
    Acquired address: None
```

**See Also** • [Understanding DHCP Services for Switches](#)

## Factory Default Configuration

The default factory configuration which contains the basic configuration settings is the first configuration of the device and is loaded when the device is first installed and powered on. For more information, see the following topics:

- [Reverting to the Default Factory Configuration on page 17](#)
- [Reverting to the Default Factory Configuration for the EX Series Switch on page 18](#)

## Reverting to the Default Factory Configuration

If for any reason the current active configuration fails, you can revert to the default factory configuration. The default factory configuration contains the basic configuration settings. This is the first configuration of the switch, and it is loaded when the switch is first installed and powered on.

The **load factory default** command is a standard Junos OS configuration command. This configuration command replaces the current active configuration with the default factory configuration.

To revert the switch to the rescue configuration:

```
[edit]
user@switch# load factory-default
[edit]
user@switch# delete system commit factory-settings
[edit]
user@switch# commit
```

- See Also**
- [Understanding Configuration Files on page 11](#)
  - [Reverting to the Rescue Configuration](#)

## Reverting to the Default Factory Configuration for the EX Series Switch

With EX Series switches, if for any reason the current active configuration fails, you can revert to the factory-default configuration.

You can also roll back to a previous configuration, as described in [“Rolling Back Junos OS Configuration Changes” on page 119](#), or revert to the rescue configuration, as described in [Reverting to the Rescue Configuration for the EX Series Switch](#).



**TIP:** If you have lost the root password, it is not necessary to revert to the factory-default configuration to reset it. See [Troubleshooting Loss of the Root Password](#).

The factory-default configuration contains the basic configuration settings for the switch. This is the first configuration of the switch and it is loaded when the switch is first powered on. For the factory-default configuration file for your switch, see the hardware documentation for your switch.



**TIP:** You can run the EZsetup script to complete the initial configuration of the switch *after* reverting to the factory-default configuration. (The EZsetup script is available only on fixed configuration switches, it is not available on modular switches.) For information on completing the initial configuration using either the CLI or the J-Web interface, see [Connecting and Configuring an EX Series Switch \(CLI Procedure\)](#) or [Connecting and Configuring an EX Series Switch \(J-Web Procedure\)](#).

You can revert to the factory-default configuration by using the **Menu** button to the right of the LCD panel on switches with LCD panel or by using the **request system zeroize** operational command or the **load factory-default** configuration command. (If your switch model does not have an LCD panel, use these commands.) You can also use the **load**

**factory-default** command to revert to the factory-default configuration file that contains all default settings *except* the root password setting, which is retained.

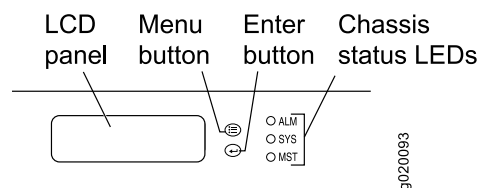
These procedures are described in the following sections:

- [Reverting to the Factory-Default Configuration Using the EX Series Switch LCD Panel on page 19](#)
- [Reverting to the EX Series Switch Factory-Default Configuration Using the request system zeroize Command on page 20](#)
- [Reverting to the EX Series Switch Factory-Default Configuration Using the load factory-default Command on page 20](#)
- [Reverting to the Factory-Default Configuration Using the Factory Reset/Mode button on EX2300 and EX3400 Switches on page 21](#)

### Reverting to the Factory-Default Configuration Using the EX Series Switch LCD Panel

To set the switch to the factory-default configuration, for EX Series switches, you can use the LCD panel and buttons on the front panel of the switch. If the EX Series switch model does not have an LCD panel, use one of the procedures described in the following sections.

*Figure 1: EX Series Switch LCD Panel*



**NOTE:** To revert a member switch of a Virtual Chassis to the factory-default configuration, first disconnect the cables connected to the Virtual Chassis ports (VCPs) to avoid affecting Virtual Chassis configuration parameters (member ID, mastership priority, and setting of VCP uplinks) on other members. See *Disconnecting a Fiber-Optic Cable*, *Disconnecting a Virtual Chassis Cable from an EX4200 Switch*, or *Disconnecting a Virtual Chassis Cable from an EX4500 Switch*.

To revert to the factory-default configuration by using the LCD panel:

1. Press the **Menu** button until you see MAINTENANCE MENU on the panel.
2. Press the **Enter** button.
3. Press **Menu** until you see FACTORY DEFAULT.
4. Press **Enter**. The display says RESTORE DEFAULT?

5. Press **Enter**. The screen flashes **FACTORY DEFAULT IN PROGRESS** and returns to the idle menu.
6. Complete the initial configuration of the switch. See *Connecting and Configuring an EX Series Switch (CLI Procedure)* or *Connecting and Configuring an EX Series Switch (J-Web Procedure)*.

### Reverting to the EX Series Switch Factory-Default Configuration Using the `request system zeroize` Command

---

The `request system zeroize` command is a standard Junos OS operational mode command that removes all configuration information and resets all key values. The operation unlinks all user-created data files, including customized configuration and log files, from their directories. The switch then reboots and reverts to the factory-default configuration.

To completely erase user-created data so that it is unrecoverable, use the `request system zeroize media` command.



**CAUTION:** Before issuing `request system zeroize`, use the `request system snapshot` command to back up the files currently used to run the switch to a secondary device.

---

To revert to the factory-default configuration by using the `request system zeroize` command:

1. `user@switch> request system zeroize`  
warning: System will be rebooted and may not boot without configuration  
Erase all data, including configuration and log files? [yes,no] (yes)
2. Type **yes** to remove configuration and log files and revert to the factory-default configuration.



**NOTE:** The `auto-image-upgrade` statement is added under the `[edit chassis]` hierarchy level when you use this procedure, and thus the automatic image upgrade feature is made available on the switch.

---

### Reverting to the EX Series Switch Factory-Default Configuration Using the `load factory-default` Command

---

The `load factory-default` command is a standard Junos OS configuration command that replaces the current active configuration with the factory-default configuration (except the root password setting, which by default is not set but which you must set in order to commit the new configuration in this procedure).

If you want to run the EZsetup script to complete the initial configuration of the switch after you revert to the factory-default configuration, do not use the `load factory-default` command. Instead do the reversion using either the LCD panel or the `request system`

**zeroize** command. If you use the **load factory-default** command to revert to the factory-default configuration, the configuration for the root password is retained and the EZsetup script will not run. (The EZsetup script is available only on fixed configuration switches, it is not available on modular switches.)



**NOTE:** The **load factory-default** command by itself is not supported on EX3300, EX4200, EX4500, and EX4550 switches configured in a Virtual Chassis.

To revert to the factory-default configuration by using the **load factory-default** command:



**NOTE:** If you use this procedure, you must delete the system commit factory settings, set the root password, and commit the configuration. These steps are not required when you revert to the factory-default configuration by using **request system zeroize**. Also, the **auto-image-upgrade** statement is not added to the configuration when you use this procedure; it *is* added to the configuration when you use **request system zeroize**.

1. [edit]  
user@switch# **load factory-default**
2. [edit]  
user@switch# **delete system commit factory-settings**
3. [edit]  
user@switch# **set system root-authentication plain-text-password**
4. [edit]  
user@switch# **commit**
5. Check the member ID and mastership priority with the **show virtual-chassis** command and check to see whether there are remaining settings for uplink VCPs by using the **show virtual-chassis vc-port** command.

### Reverting to the Factory-Default Configuration Using the Factory Reset/Mode button on EX2300 and EX3400 Switches

To set the EX2300 switches except the EX2300-24MP and EX2300-48MP switches, EX2300-C switches, and EX3400 switches to the factory-default configuration, use the Factory Reset/Mode button located on the far right side of the front panel.



**NOTE:** To revert a member switch of a Virtual Chassis to the factory-default configuration, disconnect the cables connected to the VCPs to avoid affecting Virtual Chassis configuration parameters (member ID, mastership priority, and setting of VCP uplinks) on other members. See *Disconnecting a Fiber-Optic Cable*.

To revert to the factory-default configuration by using the Factory Reset/Mode button:

1. Press the Factory Reset/Mode button for 10 seconds. The switch transitions into factory-default configuration, the console displays **committing factory default configuration**, and the Link/Activity LED on the RJ-45 network ports and the uplink ports is lit steadily in green color.
2. Press the Factory Reset/Mode button for 10 more seconds. The switch transitions into initial setup mode, the console displays **committing ezsetup config**, and the Link/Activity LED on the RJ-45 network ports and the uplink ports blink in green color.

The Factory Reset/Mode button is enabled by default. You can disable the button using the CLI.

To disable the Factory Reset/Mode button, run the commands:

1. [edit]  
user@switch# **set chassis config-button no-clear**
2. [edit]  
user@switch# **commit**

To enable the Factory Reset/Mode button, run the commands:

1. [edit]  
user@switch# **delete chassis config-button no-clear**
2. [edit]  
user@switch# **commit**

- See Also**
- *Connecting and Configuring an EX Series Switch (CLI Procedure)*
  - *Connecting and Configuring an EX Series Switch (J-Web Procedure)*
  - [Understanding Configuration Files on page 11](#)



---

## CLI Operational Mode Overview

---

In the operational mode, you use the CLI to monitor and troubleshoot the device. The **monitor**, **ping**, **show**, **test**, and **traceroute** commands let you display information and test network connectivity for the device.

- [Overview of Junos OS CLI Operational Mode Commands on page 23](#)
- [Junos OS Operational Mode Commands That Combine Other Commands on page 26](#)
- [Understanding the brief, detail, extensive, and terse Options of Junos OS Operational Commands on page 26](#)
- [Interface Naming Conventions Used in the Junos OS Operational Commands on page 27](#)
- [Using Wildcard Characters in Interface Names on page 29](#)

## Overview of Junos OS CLI Operational Mode Commands

This topic provides an overview of Junos OS CLI operational mode commands.

- [CLI Command Categories on page 23](#)
- [Commonly Used Operational Mode Commands on page 24](#)

---

### CLI Command Categories

---

When you log in to a device running Junos OS and the CLI starts, there are several broad groups of CLI commands:

- Commands for controlling the CLI environment—Some set commands in the **set** hierarchy configure the CLI display screen. For information about these commands, see [“Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies” on page 103](#). For information about CLI commands, including usage and syntax, see the [CLI Explorer](#)
- Commands for monitoring and troubleshooting—The following commands display information and statistics about the software and test network connectivity.
  - **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..
- Commands for copying files—The **copy** command copies files from one location on the router or switch to another, from the router or switch to a remote system, or from a remote system to the router or switch..
- Commands for restarting software processes—The commands in the **restart** hierarchy restart the various Junos OS processes, including the routing protocol, interface, and SNMP. .
- A command—**request**—for performing system-level operations, including stopping and rebooting the router or switch and loading Junos OS images..
- A command—**start**—to exit the CLI and start a UNIX shell. For information about this command, see the [CLI Explorer](#).
- A command—**configure**—for entering configuration mode, which provides a series of commands that configure Junos OS, including the routing protocols, interfaces, network management, and user access. .
- A command—**quit**—to exit the CLI. .
- For more information about the CLI operational mode commands, see the [CLI Explorer](#).

**See Also**

- [CLI Explorer](#)
- [Understanding Junos OS CLI Configuration Mode on page 53](#)
- [Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies on page 103](#)

### Commonly Used Operational Mode Commands

The following table lists some operational commands you may find useful for monitoring router or switch operation. For a complete description of operational commands, see the Junos OS command references.



**NOTE:** The QFX3500 switch does not support the IS-IS, OSPF, BGP, MPLS, and RSVP protocols.

**Table 4: Commonly Used Operational Mode Commands**

Items to Check	Description	Command
Software version	Versions of software running on the router or switch	<b>show version</b>
Log files	Contents of the log files	<b>monitor</b>
	Log files and their contents and recent user logins	<b>show log</b>

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

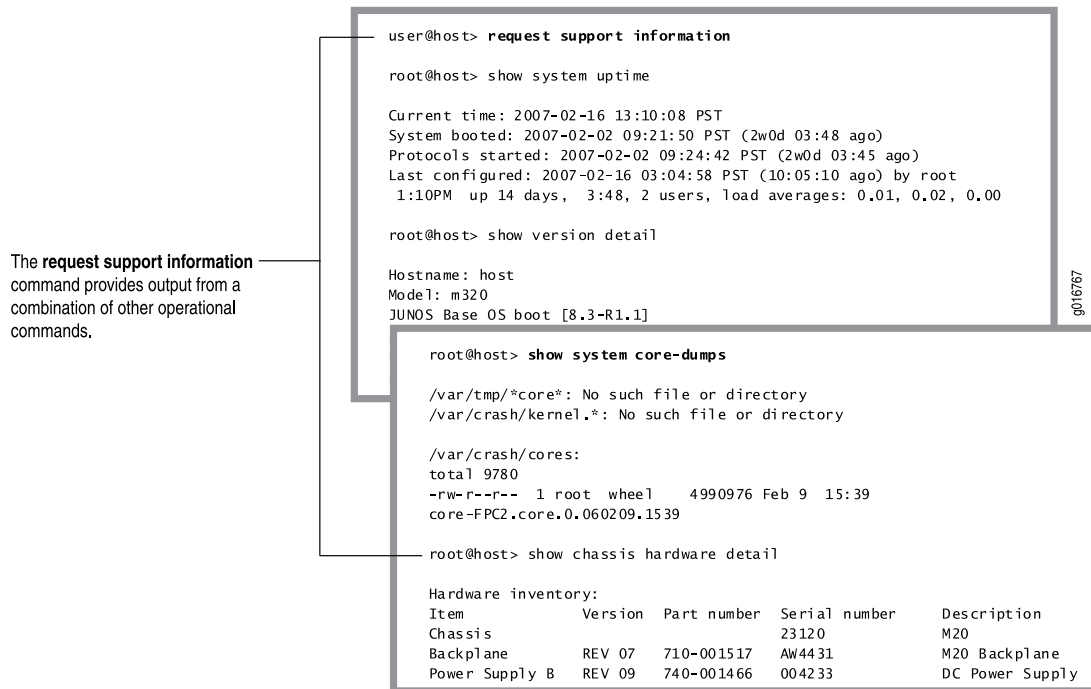
Items to Check	Description	Command
Remote systems	Host reachability and network connectivity	<b>ping</b>
	Route to a network system	<b>tracert</b>
Configuration	Current system configuration	<b>show configuration</b>
Manipulate files	List of files and directories on the router or switch	<b>file list</b>
	Contents of a file	<b>file show</b>
Interface information	Detailed information about interfaces	<b>show interfaces</b>
Chassis	Chassis alarm status	<b>show chassis alarms</b>
	Information currently on craft display	<b>show chassis craft-interface</b>
	Router or switch environment information	<b>show chassis environment</b>
	Hardware inventory	<b>show chassis hardware</b>
Routing table information	Information about entries in the routing tables	<b>show route</b>
Forwarding table information	Information about data in the kernel's forwarding table	<b>show route forwarding-table</b>
IS-IS	Adjacent routers or switches	<b>show isis adjacency</b>
OSPF	Display standard information about OSPF neighbors	<b>show ospf neighbor</b>
BGP	Display information about BGP neighbors	<b>show bgp neighbor</b>
MPLS	Status of interfaces on which MPLS is running	<b>show mpls interface</b>
	Configured LSPs on the router or switch, as well as all ingress, transit, and egress LSPs	<b>show mpls lsp</b>
	Routes that form a label-switched path	<b>show route label-switched-path</b>
RSVP	Status of interfaces on which RSVP is running	<b>show rsvp interface</b>
	Currently active RSVP sessions	<b>show rsvp session</b>
	RSVP packet and error counters	<b>show rsvp statistics</b>

**See Also** • *Configuring the Bandwidth Subscription Percentage for LSPs*

## Junos OS Operational Mode Commands That Combine Other Commands

In some cases, some Junos OS operational commands are created from a combination of other operational commands. These commands can be useful shortcuts for collecting information about the device, as shown in the following illustration.

Figure 2: Commands That Combine Other Commands



## Understanding the brief, detail, extensive, and terse Options of Junos OS Operational Commands

The Junos OS 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.

Figure 3: Command Output Options

Command output with the <b>brief</b> option.	<pre> user@host&gt; 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 </pre>
Command output with the <b>terse</b> option.	<pre> user@host&gt; show interfaces fe-1/1/1 terse Interface      Admin Link Proto  Local      Remote fe-1/1/1      up      down </pre>
Command output with the <b>extensive</b> option.	<pre> user@host&gt; 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)-- </pre>

**See Also** • [Controlling the Scope of an Operational Mode Command on page 31](#)

## Interface Naming Conventions Used in the Junos OS Operational Commands

This topic explains the interface naming conventions used in the Junos OS operational commands.

- [Physical Part of an Interface Name on page 27](#)
- [Logical Part of an Interface Name on page 28](#)
- [Channel Identifier Part of an Interface Name on page 28](#)

### Physical Part of an Interface Name

The physical interface naming conventions for Junos OS platforms is as follows:

- On SRX devices, the unique name of each network interface has the following format to identify the physical device that corresponds to a single physical network connector:

***type-slot/pim-or-ioc/port***

- On other platforms, 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.

---

### 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:

- SRX devices:

*type-slot/pim-or-ioc/port:channel.unit*

- Other platforms:

*type-fpc/pic/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.

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

- SRX devices:

*type-slot/pim-or-ioc/port:channel*  
*type-slot/pim-or-ioc/port:channel:channel*  
*type-slot/pim-or-ioc/port:channel:channel:channel*

- Other platforms:

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

- See Also**
- [Example: Configuring Interfaces Using Configuration Groups on page 142](#)
  - [Junos OS Network Interfaces Library for Routing Devices](#)

## Using Wildcard Characters in Interface Names

You can use wildcard characters in the Junos OS operational commands to specify groups of interface names without having to type each name individually. The following table lists the available wildcard characters. You must enclose all wildcard characters except the asterisk (\*) in quotation marks (" ").

*Table 5: Wildcard Characters for Specifying Interface Names*

Wildcard Character	Description
* (asterisk)	Match any string of characters in that position in the interface name. For example, <code>so*</code> matches all SONET/SDH interfaces.
"[character<character...>]"	Match one or more individual characters in that position in the interface name. For example, <code>so-"[03]"</code> matches all SONET/SDH interfaces in slots 0 and 3.
"[!character<character...>]"	Match all characters except the ones included in the brackets. For example, <code>so-"[!03]"</code> matches all SONET/SDH interfaces except those in slots 0 and 3.
"[character1-character2]"	Match a range of characters. For example, <code>so-"[0-3]"</code> 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, <code>so-"[!0-3]"</code> matches all SONET/SDH interfaces in slots 4, 5, 6, and 7.

- See Also**
- [Using Keyboard Sequences to Move Around and Edit the Junos OS CLI on page 111](#)
  - [Using Global Replace in the Junos OS Configuration on page 87](#)

## Using Operational Commands to Monitor a Device

Operational mode CLI commands enable you to monitor and control the operation of a device running the Junos OS. The operational mode commands exist in a hierarchical structure. For more information, see the following topics:

- [Using the Junos OS CLI Command Completion on page 30](#)
- [Controlling the Scope of an Operational Mode Command on page 31](#)
- [Monitoring Who Uses the Junos OS CLI on page 34](#)
- [Viewing Files and Directories on a Device Running Junos OS on page 34](#)
- [Displaying Junos OS Information on page 39](#)

- [Managing Programs and Processes Using Junos OS Operational Mode Commands on page 41](#)
- [Using the Junos OS CLI Comment Character # for Operational Mode Commands on page 45](#)
- [Using Comments in Junos OS Operational Mode Commands on page 46](#)
- [Displaying the Junos OS CLI Command and Word History on page 47](#)

## Using the Junos OS CLI Command Completion

The following examples show how you can use the command completion feature in Junos OS.

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
...
```



## Controlling the Scope of an Operational Mode Command

The Junos OS CLI operational commands include options that you can use to identify specific components on a device running Junos OS. 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>
```

### Operational Mode Commands on a TX Matrix Router or TX Matrix Plus 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 entire routing matrix.

These are the options shown in the CLI:

- **scc**—The TX Matrix router (or switch-card chassis)
- **sfc**—The TX Matrix Plus router (also referred to as or switch-fabric chassis)
- **lcc *number***—A specific router in a routing matrix based on a TX Matrix router or a TX Matrix Plus router.
- **all-lcc**—All T640 routers (in a routing matrix based on a TX Matrix router) or all T1600 routers or T4000 routers (in a routing matrix based on a TX Matrix Plus router).

If you specify none of these options, then the command applies by default to the whole routing matrix.

### 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-lcc Option)

```
user@host> show version all-lcc
```

```
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]
```

**See Also** • [Interface Naming Conventions Used in the Junos OS Operational Commands on page 27](#)

## Monitoring Who Uses the Junos OS CLI

Depending upon how you configure Junos OS, 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#
```

**See Also** • [Entering and Exiting the Junos OS CLI Configuration Mode on page 59](#)  
• [Controlling the Junos OS CLI Environment on page 126](#)

## Viewing Files and Directories on a Device Running Junos OS

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

Sections include:

- [Directories on the Device on page 35](#)
- [Listing Files and Directories on page 35](#)
- [Specifying Filenames and URLs on page 38](#)

## Directories on the Device

The following table lists some standard directories on a device running Junos OS.

*Table 6: Directories on the Device*

Directory	Description
<code>/config</code>	This directory is located on the device's router's internal flash drive. It contains the active configuration ( <b>juniper.conf</b> ) and rollback files 1, 2, and 3.
<code>/var/db/config</code>	This directory is located on the router's device's hard drive and contains rollback files 4 through 49.
<code>/var/tmp</code>	This directory is located on the device'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.
<code>/var/log</code>	This directory is located on the device's hard drive. It contains files generated by both the device's logging function as well as the <b>traceoptions</b> command.
<code>/var/home</code>	This directory is located on the device's hard drive. It contains a subdirectory for each configured user on the device. These individual user directories are the default file location for many Junos OS commands.
<code>/altroot</code>	This directory is located on the device'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.
<code>/altconfig</code>	This directory is located on the device's hard drive and contains a copy of the <b>/config</b> 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
|              Pipe through a command
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 device. For example, to show the files located in your home directory on the device:

```

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 device. In fact, the user's home directory is the default directory for most of Junos OS 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 device'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 2017
/altconfig/    Last changed: Aug 07 2017
/altroot/      Last changed: Aug 07 2017
/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 2017
/dev/          Last changed: Apr 09 22:36:21
/etc/          Last changed: Apr 11 03:14:22
/kernel        Size: 27823246, Last changed: Aug 07 2017
/mfs/          Last changed: Apr 09 22:36:49
/mnt/          Last changed: Jan 11 2017
/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 2017
/var/at/        Last changed: Jul 09 2017
/var/backups/   Last changed: Jul 09 2017
/var/bin/       Last changed: Jul 09 2017
/var/crash/     Last changed: Apr 09 22:31:08
/var/cron/      Last changed: Jul 09 2017
/var/db/        Last changed: May 07 20:28:40
/var/empty/     Last changed: Jul 09 2017
/var/etc/       Last changed: Apr 16 22:35:36
/var/heimdal/   Last changed: Jul 10 2017
/var/home/      Last changed: Apr 09 22:59:18
/var/jail/      Last changed: Oct 31 2017
/var/log/       Last changed: Apr 17 02:00:10
/var/mail/      Last changed: Jul 09 2017
/var/messages/  Last changed: Jul 09 2017
/var/named/     Last changed: Jul 10 2017
/var/packages/  Last changed: Jan 18 02:38:59
/var/pdb/       Last changed: Oct 31 2017
/var/preserve/  Last changed: Jul 09 2017
/var/run/       Last changed: Apr 17 02:00:01
/var/rundb/     Last changed: Apr 17 00:46:00
/var/rwho/      Last changed: Jul 09 2017
/var/sdb/       Last changed: Apr 09 22:37:31
/var/spool/     Last changed: Jul 09 2017
/var/sw/        Last changed: Jul 09 2017
/var/tmp/       Last changed: Apr 09 23:28:41
/var/transfer/  Last changed: Jul 09 2017
/var/yp/        Last changed: Jul 09 2017
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 2017-06-12 07:58:27
UTC
Jul  9 23:18:05 CHASSISD release 8.4I0 built by builder on 2017-06-12 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
Jul  9 23:19:03 SSRAM bank 0 - part number 710-001385, serial number 243071
Jul  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 OS. 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 OS Information

You can display Junos OS version information and other status to determine if the version of Junos OS that you are running supports specific features or hardware.

To display Junos OS 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 OS 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 OS. If the Junos Crypto Software Suite is not listed, the router is running worldwide nonencrypted Junos OS.

3. To display detailed version information, enter the **show version detail** command. This command display shows the hostname and version information for Junos OS 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 2017-08-08 00:33:41 UTC
```

```
MGD release 8.4R1.13 built by builder on 2017-08-08 00:34:00 UTC
CLI release 8.4R1.13 built by builder on 2017-08-08 00:34:47 UTC
RPD release 8.4R1.13 built by builder on 2017-08-08 00:45:21 UTC
CHASSISD release 8.4R1.13 built by builder on 2017-08-08 00:36:59 UTC
DFWD release 8.4R1.13 built by builder on 2017-08-08 00:39:32 UTC
DCD release 8.4R1.13 built by builder on 2017-08-08 00:34:24 UTC
SNMPD release 8.4R1.13 built by builder on 2017-08-08 00:42:24 UTC
MIB2D release 8.4R1.13 built by builder on 2017-08-08 00:46:47 UTC
APSD release 8.4R1.13 built by builder on 2017-08-08 00:36:39 UTC
VRRPD release 8.4R1.13 built by builder on 2017-08-08 00:45:44 UTC
ALARMD release 8.4R1.13 built by builder on 2017-08-08 00:34:30 UTC
PFED release 8.4R1.13 built by builder on 2017-08-08 00:41:54 UTC
CRAFTD release 8.4R1.13 built by builder on 2017-08-08 00:39:03 UTC
SAMPLED release 8.4R1.13 built by builder on 2017-08-08 00:36:05 UTC
ILMID release 8.4R1.13 built by builder on 2017-08-08 00:36:51 UTC
RMOPD release 8.4R1.13 built by builder on 2017-08-08 00:42:04 UTC
COSD release 8.4R1.13 built by builder on 2017-08-08 00:38:39 UTC
FSAD release 8.4R1.13 built by builder on 2017-08-08 00:43:01 UTC
IRSD release 8.4R1.13 built by builder on 2017-08-08 00:35:37 UTC
FUD release 8.4R1.13 built by builder on 2017-08-08 00:44:36 UTC
RTSPD release 8.4R1.13 built by builder on 2017-08-08 00:29:14 UTC
SMARTD release 8.4R1.13 built by builder on 2017-08-08 00:13:32 UTC
KSYNCD release 8.4R1.13 built by builder on 2017-08-08 00:33:17 UTC
SPD release 8.4R1.13 built by builder on 2017-08-08 00:43:50 UTC
L2TPD release 8.4R1.13 built by builder on 2017-08-08 00:43:12 UTC
HTTPD release 8.4R1.13 built by builder on 2017-08-08 00:36:27 UTC
PPPOED release 8.4R1.13 built by builder on 2017-08-08 00:36:04 UTC
RDD release 8.4R1.13 built by builder on 2017-08-08 00:33:49 UTC
PPPD release 8.4R1.13 built by builder on 2017-08-08 00:45:13 UTC
DFCD release 8.4R1.13 built by builder on 2017-08-08 00:39:11 UTC
DLSWD release 8.4R1.13 built by builder on 2017-08-08 00:42:37 UTC
LACPD release 8.4R1.13 built by builder on 2017-08-08 00:35:41 UTC
USBD release 8.4R1.13 built by builder on 2017-08-08 00:30:01 UTC
LFMD release 8.4R1.13 built by builder on 2017-08-08 00:35:52 UTC
CFMD release 8.4R1.13 built by builder on 2017-08-08 00:34:45 UTC
JDHCPD release 8.4R1.13 built by builder on 2017-08-08 00:35:40 UTC
PGCPD release 8.4R1.13 built by builder on 2017-08-08 00:46:31 UTC
SSD release 8.4R1.13 built by builder on 2017-08-08 00:36:17 UTC
MSPD release 8.4R1.13 built by builder on 2017-08-08 00:33:42 UTC
KMD release 8.4R1.13 built by builder on 2017-08-08 00:44:02 UTC
PPMD release 8.4R1.13 built by builder on 2017-08-08 00:36:03 UTC
LMPD release 8.4R1.13 built by builder on 2017-08-08 00:33:49 UTC
LRMUXD release 8.4R1.13 built by builder on 2017-08-08 00:33:55 UTC
PGMD release 8.4R1.13 built by builder on 2017-08-08 00:36:01 UTC
BFDD release 8.4R1.13 built by builder on 2017-08-08 00:44:22 UTC
SDXD release 8.4R1.13 built by builder on 2017-08-08 00:36:18 UTC
AUDITD release 8.4R1.13 built by builder on 2017-08-08 00:34:40 UTC
L2ALD release 8.4R1.13 built by builder on 2017-08-08 00:40:05 UTC
EVENTD release 8.4R1.13 built by builder on 2017-08-08 00:39:55 UTC
L2CPD release 8.4R1.13 built by builder on 2017-08-08 00:41:04 UTC
MPLSOAMD release 8.4R1.13 built by builder on 2017-08-08 00:45:11 UTC
jroute-dd release 8.4R1.13 built by builder on 2017-08-08 00:31:01 UTC
jkernel-dd release 8.4R1.13 built by builder on 2017-08-08 00:30:30 UTC
jcrypto-dd release 8.4R1.13 built by builder on 2017-08-08 00:30:12 UTC
jdocs-dd release 8.4R1.13 built by builder on 2017-08-08 00:02:52 UTC
```

```
user@host>
```

## Managing Programs and Processes Using Junos OS Operational Mode Commands

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

- [Showing Software Processes on page 41](#)
- [Restarting the Junos OS Process on page 43](#)
- [Stopping Junos OS on page 44](#)
- [Rebooting Junos OS on page 45](#)

### 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 device.

To show software processes:

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

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

```
last 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%	l2ald
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%	rpdp
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%	ppmd
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%	lfmd
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%	pfed
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	v1ruwt	0:26	0.00%	0.00%	vn1ru
3371	root	2	0	1024K	216K	sbwait	0:25	0.00%	0.00%	tnp.snmpd

```

    13 root      -18  0    OK    OK 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    OK    OK 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)---

```

The following table lists and describes the output fields included in this example. The fields are listed in alphabetical order.

*Table 7: show system process extensive Command Output Fields*

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

---

## Restarting the Junos OS Process

---

To correct an error condition, you might need to restart a software process running on the device. 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 device 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 OS to see a list of software processes that you can restart using this command.
- **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.

Figure 4: Restarting a Process

PID	USERNAME	PRI	NICE	SIZE	RES	STATE	TIME	WCPU	CPU	COMMAND
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%	rpdp
553	root	2	0	8792K	1544K	select	0:01	0.00%	0.00%	mib2d

PID before 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%	rpdp
558	root	2	20	8708K	1880K	select	0:00	0.00%	0.00%	samp1ed
555	root	2	0	1856K	932K	select	0:00	0.00%	0.00%	vrpdp
686	root	2	0	7808K	940K	select	0:00	0.00%	0.00%	apsd

PID after restart

## Stopping Junos OS



**CAUTION:** To avoid possible damage to the file system and to prevent loss of data, you must always gracefully shut down Junos OS before powering off the device.



**NOTE:** SRX Series Services Gateway devices for the branch and EX Series Ethernet Switches support resilient dual-root partitioning.

If you are unable to shut down a device gracefully because of unexpected circumstances such as a power outage or a device failure, resilient dual-root partitioning prevents file corruption and enables a device to remain operational. In addition, it enables a device to boot transparently from the second root partition if the system fails to boot from the primary root partition.

Resilient dual-root partitioning serves as a backup mechanism for providing additional resiliency to a device when there is an abnormal shutdown. However, it is not an alternative to performing a graceful shutdown under normal circumstances.

To stop Junos OS:

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 Junos OS

After a software upgrade or to recover (occasionally) from an error condition, you must reboot Junos OS.

To reboot Junos OS:

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...

```

**See Also** • [Checking the Status of a Device Running Junos OS on page 116](#)

## Using the Junos OS CLI Comment Character # for Operational Mode Commands

The comment character in Junos OS 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 OS.

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 #.

## Using Comments in Junos OS Operational Mode Commands

Following are some examples showing how to Junos OS operational mode comments.

The following example shows how to use comments in a file:

```
#Command 1: Show the router version  
show version  
#Command 2: Show all router interfaces  
show interfaces terse
```

The following example shows how to copy and paste contents of a file into the CLI:

```
user@host> #Command 1: Show the router version  
user@host> show version  
Hostname: myhost  
Model: m5  
Junos Base OS boot [16.4-20040511.0]  
Junos Base OS Software Suite [16.4-20040511.0]  
Junos Kernel Software Suite [16.4-20040511.0]  
Junos Packet Forwarding Engine Support (M5/M10) [16.4-20040511.0] Junos Routing  
  Software Suite [16.4-20040511.0] Junos Online Documentation [16.4-20040511.0] Junos  
  Crypto Software Suite [16.4-20040511.0]  
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  
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
```

## Displaying the Junos OS CLI Command and Word History

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

```
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
```

You can press 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, pressing Esc+. or Alt+. again restarts scrolling from the last word entered.

**See Also** • [Junos OS CLI Online Help Features on page 50](#)

## Online Help in the CLI

The Junos OS CLI includes comprehensive system of online help, which includes several options for getting help any time.

- [Getting Online Help from the Junos OS Command-Line Interface on page 47](#)
- [Junos OS CLI Online Help Features on page 50](#)
- [CLI Explorer Overview on page 52](#)

## Getting Online Help from the Junos OS Command-Line Interface

The Junos OS command-line interface (CLI) has a context-sensitive online help feature that enables you to access information about commands and statements from the Junos OS CLI.

- [Getting Help About Commands on page 48](#)
- [Getting Help About a String in a Statement or Command on page 49](#)
- [Getting Help About Configuration Statements on page 50](#)
- [Getting Help About System Log Messages on page 50](#)

## 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 context-relevant 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, this is the result:

```
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 re-displays the command names and options 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 re-displays 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.

For statements or commands which need input data type as STRING, the supported characters set is as follows:

- Any printable ASCII characters
- For characters with space, it should be enclosed in double-quotes
- To have double-quote as the input, it should be escaped with '\'



**NOTE:** No escape characters are supported in a string other than to escape from double quotes.

- The range of supported characters for attributes is 0 through 65499 characters.
- The range of supported characters for string type identifiers is 1 through 255 characters.

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 OS configuration guides. The **help reference** command displays summary information about the statement based on the summary descriptions that appear in the Junos OS 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.

**See Also** • [Getting Started with the Junos OS Command-Line Interface on page 107](#)

## Junos OS CLI Online Help Features

The Junos OS CLI online help provides the following features for ease of use and error prevention:

- [Help for Omitted Statements on page 50](#)
- [Using CLI Command Completion on page 51](#)
- [Using Command Completion in Configuration Mode on page 51](#)
- [Displaying Tips About CLI Commands on page 51](#)

---

### Help for Omitted Statements

If you have omitted a required statement at a specific 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 OS CLI provides you a command completion option that enables Junos OS 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 Space. 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 Space.

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 Space 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 later. 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>
```

**See Also** • [Examples: Using the Junos OS CLI Command Completion on page 29](#)

## CLI Explorer Overview

CLI Explorer is a Web application that helps you to explore Junos OS configuration statements and commands. It lists all the configuration statements and commands supported in the Junos OS across different platforms on several products.

To view the available configuration statements and commands, you can use any of the following filtering options:

- Filter by product family—To find the CLI reference information by product family, you can either select “All products” or select any of the specific product.

For example: ACX Series, EX Series.

- Filter by number or letter—To find the CLI reference information by number or letter, you can either select “All” or filter by numbers “3” or “8” or any of the letters (“A”, “B”, “C”...).

For example, if you select the letter “A”, commands such as **aaa**, **aaa clients (TDF)**, **aaa-access-profile (L2TP LNS)** appear.

- Filter by the normal search option—To use this option to filter the commands and statements, you enter your search criteria.

For example, if you enter the number “3”, all the commands and statements containing the number “3” appear in the search results.

When you click on the link in the search results, you are directed to a page describing the command or statement that is referenced in a feature guide.

To explore the Junos OS configuration statements and commands, see the [CLI Explorer](#).

**See Also** • [Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies on page 103](#)

## CLI Configuration Mode Overview

---

The configuration mode of the Junos OS CLI enables you to configure a device, using configuration statements to set, manage, and monitor device properties.

- [Understanding Junos OS CLI Configuration Mode on page 53](#)
- [Entering and Exiting the Junos OS CLI Configuration Mode on page 59](#)
- [Issuing Relative Junos OS Configuration Mode Commands on page 61](#)
- [Using Command Completion in Configuration Mode on page 62](#)
- [Notational Conventions Used in Junos OS Configuration Hierarchies on page 64](#)

## Understanding Junos OS CLI Configuration Mode

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

As described in “[Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies](#)” on page 103, a device configuration is stored as a hierarchy of statements. In configuration mode, you create the specific hierarchy of configuration statements to use. When you have finished entering the configuration statements and you are certain they are complete and correct, you commit them, which activates the configuration on the device.

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

This topic covers:

- [Configuration Mode Commands on page 54](#)
- [Configuration Statements and Identifiers on page 55](#)
- [Configuration Statement Hierarchy on page 57](#)

## Configuration Mode Commands

The following table summarizes each CLI configuration mode command. The commands are organized alphabetically.

*Table 8: Summary of Configuration Mode Commands*

Command	Description
<b>activate</b>	Remove the <b>inactive:</b> 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 <b>commit</b> command.
<b>annotate</b>	Add comments to a configuration. You can add comments only at the current hierarchy level.
<b>commit</b>	Commit the set of changes to the database and cause the changes to take operational effect.
<b>copy</b>	Make a copy of an existing statement in the configuration.
<b>deactivate</b>	Add the <b>inactive:</b> 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 <b>commit</b> command.
<b>delete</b>	Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it.
<b>edit</b>	Move inside the specified statement hierarchy. If the statement does not exist, it is created.
<b>exit</b>	Exit the current level of the statement hierarchy, returning to the level prior to the last edit command or exit from configuration mode. The <b>quit</b> and <b>exit</b> commands are synonyms.
<b>extension</b>	Manage configurations that are contributed by SDK application packages. Either display or delete user-defined configuration contributed by the named SDK application package. A configuration defined in any native Junos OS package is never deleted by the extension command.
<b>help</b>	Display help about available configuration statements.
<b>insert</b>	Insert an identifier into an existing hierarchy.
<b>load</b>	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.



*Table 8: Summary of Configuration Mode Commands (continued)*

Command	Description
<b>quit</b>	Exit the current level of the statement hierarchy, returning to the level prior to the last edit command, or exit from configuration mode. The <b>quit</b> and <b>exit</b> commands are synonyms.
<b>rename</b>	Rename an existing configuration statement or identifier.
<b>replace</b>	Replace identifiers or values in a configuration.
<b>rollback</b>	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 <b>commit configuration</b> command.
<b>run</b>	Run a top-level CLI command without exiting from configuration mode.
<b>save</b>	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.
<b>set</b>	Create a statement hierarchy and set identifier values. This is similar to <b>edit</b> except that your current level in the hierarchy does not change.
<b>show</b>	Display the current configuration.
<b>status</b>	Display the users currently editing the configuration.
<b>top</b>	Return to the top level of configuration command mode, which is indicated by the <b>[edit]</b> banner.
<b>up</b>	Move up one level in the statement hierarchy.
<b>update</b>	Update a private database.
<b>wildcard</b>	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 a pattern. Based on this pattern, you search for items that contain these patterns and delete them.

### Configuration Statements and Identifiers

You can configure device properties by including the corresponding statements in the configuration. Typically, a statement consists of a keyword, which is fixed text, and an optional identifier. An identifier is an identifying name that you can define, such as the

name of an interface or a username, which enables you and the CLI to differentiate among a collection of statements.



**NOTE:** The QFX3500 switch does not support the IS-IS, OSPF, BGP, LDP, MPLS, and RSVP protocols.

**Table 9: Configuration Mode Top-Level Statements**

Statement	Description
<b>access</b>	Configure the Challenge Handshake Authentication Protocol (CHAP). For information about the statements in this hierarchy, see the <i>Junos OS Administration Library</i> .
<b>accounting-options</b>	Configure accounting statistics data collection for interfaces and firewall filters. For information about the statements in this hierarchy, see the <i>Network Management and Monitoring Guide</i> .
<b>chassis</b>	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 OS Administration Library</i> .
<b>class-of-service</b>	Configure class-of-service parameters. For information about the statements in this hierarchy, see the <a href="#">Junos OS Class of Service Feature Guide for Routing Devices</a> .
<b>firewall</b>	Define filters that select packets based on their contents. For information about the statements in this hierarchy, see the <i>Routing Policies, Firewall Filters, and Traffic Policers Feature Guide</i> .
<b>forwarding-options</b>	Define forwarding options, including traffic sampling options. For information about the statements in this hierarchy, see the <i>Junos OS Network Interfaces Library for Routing Devices</i> .
<b>groups</b>	Configure configuration groups. For information about statements in this hierarchy, see the <i>Junos OS Administration Library</i> .
<b>interfaces</b>	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 OS Network Interfaces Library for Routing Devices</i> .
<b>policy-options</b>	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>Routing Policies, Firewall Filters, and Traffic Policers Feature Guide</i> .
<b>protocols</b>	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 OS Routing Protocols Library</i> and the <a href="#">MPLS Applications Feature Guide for Routing Devices</a> .

Table 9: Configuration Mode Top-Level Statements (continued)

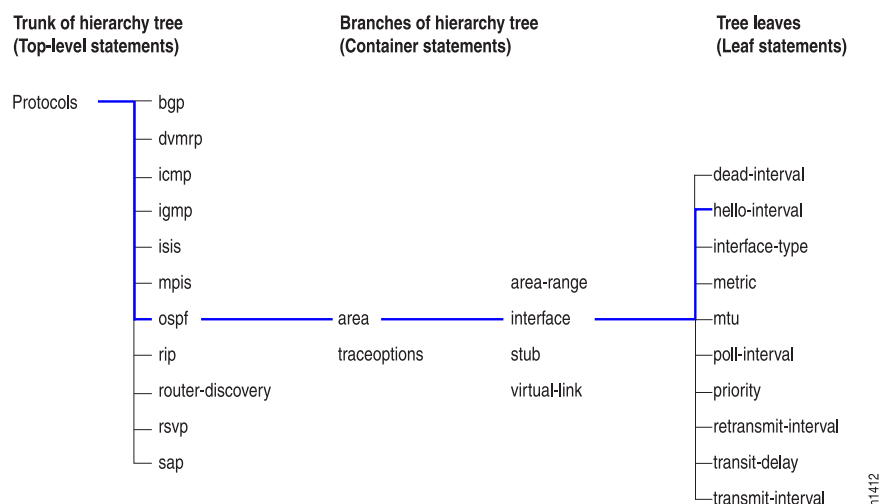
Statement	Description
<b>routing-instances</b>	Configure multiple routing instances. For information about the statements in this hierarchy, see the <i>Junos OS Routing Protocols Library</i> .
<b>routing-options</b>	Configure protocol-independent routing options, such as static routes, autonomous system numbers, confederation members, and global tracing (debugging) operations to log. For information about the statements in this hierarchy, see the <i>Junos OS Routing Protocols Library</i> .
<b>security</b>	Configure IP Security (IPsec) services. For information about the statements in this hierarchy see the <i>Junos OS Administration Library</i> .
<b>snmp</b>	Configure SNMP community strings, interfaces, traps, and notifications. For information about the statements in this hierarchy, see the <i>Network Management and Monitoring Guide</i> .
<b>system</b>	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 OS Administration Library</i> .

For specific information on configuration statements, see the Junos OS configuration guides.

### Configuration Statement Hierarchy

The Junos OS 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. All the container and leaf statements together form the configuration hierarchy.

Figure 5: 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. The previous illustration shows 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). 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 5 on page 57](#):

**[edit protocols ospf area *area-number* interface *interface-name*]**

The command 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.

The configuration hierarchy can also contain "oneliners" at the last level in the hierarchy. Oneliners remove one level of braces in the syntax and display the container statement, its identifiers, the child or leaf statement and its attributes all on one line. For example, in the following sample configuration hierarchy, the line **level 1 metric 10** is a oneliner because the **level** container statement with identifier **1**, its child statement **metric**, and its corresponding attribute **10** all appear on a single line in the hierarchy:

**[edit protocols]**

```
isis {
  interface ge-0/0/0.0 {
    level 1 metric 10;
  }
}
```

Likewise, in the following example, **dynamic-profile** *dynamic-profile-name* **aggregate-clients**; is a oneliner because the **dynamic-profile** statement, its identifier *dynamic-profile-name*, and leaf statement **aggregate-clients** all appear on one line when you run the **show** command in the configuration mode:

```
[edit forwarding-options]
user@host# show
dhcp-relay {
  dynamic-profile dynamic-profile-name aggregate-clients;
}
```

## Entering and Exiting the Junos OS CLI Configuration Mode

You configure Junos OS 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, you can view and modify only those statements for which you have access privileges set.

- 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.

- 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
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>
```

- See Also**
- [Junos OS Administration Library](#)
  - [Switching Between Junos OS CLI Operational and Configuration Modes on page 109](#)
  - [Using the configure exclusive Command on page 140](#)
  - [Updating the configure private Configuration on page 141](#)
  - [Modifying the Junos OS Configuration on page 68](#)
  - [Displaying set Commands from the Junos OS Configuration on page 8](#)
  - [Commit Operation When Multiple Users Configure the Software on page 174](#)
  - [Managing Programs and Processes Using Junos OS Operational Mode Commands on page 41](#)

## Issuing Relative Junos OS Configuration Mode 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
```

- See Also**
- [Displaying the Current Junos OS Configuration on page 3](#)

## Using Command Completion in Configuration Mode

This topic demonstrates using basic command completion in the Junos OS CLI 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 ?

```
<policy_name>      Name to identify a policy filter
user@host# show policy-options policy-statement
<policy_name>      Name to identify a policy filter
lo0only-v4          Name to identify a policy filter
lo0only-v6          Name to identify a policy filter
lo2bgp              Name to identify a policy filter
```

**See Also** • [Adding Junos OS Configuration Statements and Identifiers on page 69](#)

- [Examples: Using the Junos OS CLI Command Completion on page 29](#)
- [Displaying the Junos OS CLI Command and Word History on page 47](#)
- [Adding Junos OS Configuration Statements and Identifiers on page 69](#)

## Notational Conventions Used in Junos OS Configuration Hierarchies

When you are working in Junos OS command-line interface (CLI) configuration mode, the banner on the line preceding the prompt indicates the current hierarchy level. In the following example, the level is **[edit protocols ospf]**:

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

(The Junos OS documentation uses **user@host#** as the standard configuration mode prompt. In an actual CLI session, the prompt shows your user ID and the configured name of the Juniper Networks device you are working on.)

Use the **set ?** command to display the statements that you can include in the configuration at the current level. The **help apropos** command is also context-sensitive, displaying matching statements only at the current level and below.



**NOTE:** In this topic, statements are listed alphabetically within each hierarchy and subhierarchy. If a subhierarchy is sufficiently long that it might be difficult to determine where it ends and its next peer statement begins, the subhierarchy appears at the end of its parent hierarchy instead of in alphabetical order. In this case, a placeholder appears in its actual alphabetical position.

For example, at the **[edit interfaces *interface-name* unit *logical-unit-number*]** hierarchy level, the family *family-name* subhierarchy has more than 20 child statements, including several subhierarchies with child statements of their own. The full family *family-name* hierarchy appears at the end of its parent hierarchy (**[edit interfaces *interface-name* unit *logical-unit-number*]**), and the following placeholder appears at its actual alphabetical position:

```
family family-name {
  ... the family subhierarchy appears after the main [edit interfaces interface-name
    unit logical-unit-number] hierarchy ...
}
```

Another exception to alphabetical order is that the **disable** statement always appears first in any hierarchy that includes it.

- 
- See Also**
- [Configuration Features in the Junos OS](#)
  - [Configuration Mode Commands in the Junos OS](#)

## Backing Up Configurations to an Archive Site

You can configure a device to transfer its configuration to an archive file periodically. The following tasks describe how to transfer the configuration to an archive site:

1. [Configuring the Transfer of the Currently Active Configuration to an Archive Site on page 65](#)
2. [Configuring the Periodic Transfer of the Active Configuration to an Archive Site on page 65](#)
3. [Configuring the Transfer of the Currently Active Configuration When a Configuration Is Committed on page 66](#)
4. [Configuring Archive Sites for the Transfer of Active Configuration Files on page 66](#)

### Configuring the Transfer of the Currently Active Configuration to an Archive Site

If you want to back up your device's current configuration to an archive site, you can configure the device to transfer its currently active configuration by FTP, HTTP, or secure copy (SCP) periodically or after each commit.

To configure the device to transfer its currently active configuration to an archive site, include statements at the **[edit system archival configuration]** hierarchy level:

```
[edit system archival configuration]
archive-sites {
  ftp://username<:password>@host-address<:port>/url-path;
  scp://username<:password>@host-address<:port>/url-path;
  http://username @host-address :url-path <password>;
}
transfer-interval interval;
transfer-on-commit;
```



**NOTE:** When specifying a URL in a Junos OS statement using an IPv6 host address, you must enclose the entire URL in quotation marks (") and enclose the IPv6 host address in brackets ([ ]). For example, "ftp://username<:password>@[ipv6-host-address]<:port>/url-path"

### Configuring the Periodic Transfer of the Active Configuration to an Archive Site

To configure the device to periodically transfer its currently active configuration to an archive site, include the **transfer-interval** statement at the **[edit system archival configuration]** hierarchy level:

```
[edit system archival configuration]
transfer-interval interval;
```

The **interval** is a period of time ranging from 15 through 2880 minutes.

## Configuring the Transfer of the Currently Active Configuration When a Configuration Is Committed

To configure the device to transfer its currently active configuration to an archive site each time you commit a candidate configuration, include the **transfer-on-commit** statement at the **[edit system archival configuration]** hierarchy level:

```
[edit system archival configuration]
transfer-on-commit;
```



**NOTE:** When specifying a URL in a Junos OS statement using an IPv6 host address, you must enclose the entire URL in quotation marks (") and enclose the IPv6 host address in brackets ([ ]). For example,  
 "scp://username<:password>@[ipv6-host-address]<:port>/url-path"

## Configuring Archive Sites for the Transfer of Active Configuration Files

When you configure the device to transfer its configuration files, you specify an archive site to which the files are transferred. If you specify more than one archive site, the device attempts to transfer files to the first archive site in the list, moving to the next site only if the transfer fails.

When you use the **archive-sites** statement, you can specify a destination as an FTP URL, HTTP URL, or SCP-style remote file specification. The URL type **file://** is also supported.

To configure the archive site, include the **archive-sites** statement at the **[edit system archival configuration]** hierarchy level:

```
[edit system archival configuration]
archive-sites {
  ftp://username@host:<port>url-path password password;
  scp://username@host:<port>url-path password password;
  file://<path>/<filename>;
  http://username@host: url-path password password;
}
```



**NOTE:** When specifying a URL in a Junos OS statement using an IPv6 host address, you must enclose the entire URL in quotation marks (") and enclose the IPv6 host address in brackets ([ ]). For example,  
 "scp://username<:password>@[ipv6-host-address]<:port>/url-path"

When you specify the archive site, do not add a forward slash (/) to the end of the URL.

The destination filename is saved in the following format, where *n* corresponds to the number of the compressed configuration rollback file that has been archived:

```
<router-name>_YYYYMMDD_HHMMSS_juniper.conf.n.gz
```



**NOTE:** Whenever configurations are made, the time included in the destination filename is either in Coordinated Universal Time (UTC) or Japan Standard Time (JST) . The default time zone on the device is UTC.

**See Also** • [Junos OS Commit Model for Configurations on page 171](#)

## Modifying the Configuration for a Device

Junos CLI enables you to modify an existing Junos OS configuration. This section also explains the specifics of adding a statement, deleting a statement, copying a statement, and inserting a new identifier, including examples.

- [Displaying Users Currently Editing the Junos OS Configuration on page 68](#)
- [Modifying the Junos OS Configuration on page 68](#)
- [Adding Junos OS Configuration Statements and Identifiers on page 69](#)
- [Deleting a Statement from a Junos OS Configuration on page 70](#)
- [Example: Deleting a Statement from the Junos OS Configuration on page 71](#)
- [Copying a Junos OS Statement in the Configuration on page 73](#)
- [Example: Copying a Statement in the Junos Configuration on page 73](#)
- [Example: Re-Using Configuration on page 75](#)
- [Inserting a New Identifier in a Junos OS Configuration on page 81](#)
- [Example: Inserting a New Identifier in a Junos Configuration on page 81](#)
- [Renaming an Identifier in a Junos OS Configuration on page 85](#)
- [Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration on page 85](#)
- [Example: Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration on page 85](#)
- [Using Global Replace in the Junos OS Configuration on page 87](#)
- [Common Regular Expressions to Use with the replace Command on page 88](#)
- [Example: Using Global Replace in a Junos OS Configuration—Using the \n Back Reference on page 89](#)
- [Example: Using Global Replace in a Junos OS Configuration—Replacing an Interface Name on page 91](#)
- [Example: Using Global Replace in a Junos OS Configuration—Using the upto Option on page 93](#)
- [Using Regular Expressions to Delete Related Items from a Junos OS Configuration on page 95](#)
- [Adding Comments in a Junos OS Configuration on page 96](#)
- [Example: Including Comments in a Junos OS Configuration by Using the CLI on page 98](#)

## Displaying Users Currently Editing the Junos OS 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 2018-03-01 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 (**2018-03-01 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 (**2018-10-31 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 2018-10-31 14:55:15 PST, idle 00:03:09
commit at
```

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 (**2018-11-01 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 2018-11-01 13:05:11 PST
exclusive [edit]
```

- See Also**
- [Forms of the configure Command on page 138](#)
  - [Scheduling a Junos OS Commit Operation on page 180](#)
  - [Using the configure Command on page 139](#)

## Modifying the Junos OS Configuration

To configure a device running Junos OS or to modify an existing Junos OS 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 specified 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.



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

- See Also**
- [Issuing Relative Junos OS Configuration Mode Commands on page 61](#)
  - [Using the configure exclusive Command on page 140](#)
  - [Updating the configure private Configuration on page 141](#)
  - [Displaying the Current Junos OS Configuration on page 3](#)

## Adding Junos OS Configuration Statements and Identifiers

All properties of a device running Junos OS 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
          ^
syntax error, expecting <identifier>
```

- See Also**
- [Using the configure exclusive Command on page 140](#)
  - [Displaying the Current Junos OS Configuration on page 3](#)
- Additional Details About Specifying Junos OS Statements and Identifiers*

## Deleting a Statement from a Junos OS Configuration

To delete a statement or identifier from a Junos OS 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)
```



**NOTE:** You cannot delete multiple statements or identifiers within a hierarchy using a single **delete** command. You must delete each statement or identifier individually using multiple **delete** commands. For example, consider the following configuration at the **[edit system]** hierarchy level:

```
system {
  host-name host-211;
  domain-name domain-122;
  backup-router 192.168.71.254;
  arp;
  authentication-order [ radius password tacplus ];
}
```

To delete the **domain-name**, **host-name**, and **backup-router** from the configuration, you cannot issue a single **delete** command. For example, the following command would not work:

```
user@host> delete system hostname host-211 domain-name domain-122 backup-router
192.168.71.254
```

Instead, you must delete each statement individually:

```
user@host delete system host-name host-211
user@host delete system domain-name domain-122
user@host delete system backup-router 192.168.71.254
```

## Example: Deleting a Statement from the Junos OS 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] yes
[edit protocols ospf area 0.0.0.0]
user@host# show
[edit]
user@host#
```

Unconfigure a specific property, in this case, removing the interface speed setting:

```
[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;
}
```

## Copying a Junos OS Statement in the Configuration

When you have many similar statements in a Junos configuration, 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

This example shows how you can create one virtual connection (VC) on an interface by copying an existing VC.

- [Requirements on page 73](#)
- [Overview on page 74](#)
- [Configuration on page 74](#)

### Requirements

No special configuration beyond device initialization is required before configuring this example.

Before you begin this example, configure the following initial configuration.

```
[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;
    }
  }
}
```

To quickly configure the initial configuration for this example, copy the following commands, paste it into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste this command into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces at-1/0/0 description "PAIX to MAE West"
set interfaces at-1/0/0 encapsulation atm-pvc
set interfaces at-1/0/0 unit 61 point-to-point
set interfaces at-1/0/0 unit 61 vci 0.61
set interfaces at-1/0/0 unit 61 family inet address 10.0.1.1/24
```

---

## Overview

Copying statements is useful when you are configuring many physical or logical interfaces of the same type. 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. In the case of this example, we are adding a virtual connection that is very similar to a virtual connection already configured.

---

## Configuration

### CLI Quick Configuration

Start at the **[edit interfaces at-1/0/0]** hierarchy level.

```
copy unit 61 to unit 62
set unit 62 vci 0.62
edit unit 62
replace pattern 10.0.1.1 with 10.0.2.1
```

### Configuring by Copying

### Step-by-Step Procedure

To configure by copying a configuration:

1. Go to the **[edit interfaces at-1/0/0]** hierarchy level and copy unit 61.

```
[edit interfaces at-1/0/0]
user@host# copy unit 61 to unit 62
```

2. Take a look at the new configuration and see what you need to change to make the configuration valid..

```
user@host# show interfaces 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;
  }
}
unit 62 {
  point-to-point;
  vci 0.61;
  family inet {
    address 10.0.1.1/24;
  }
}
```

```
}
```

3. Change the configuration to make it valid.

In this example you want to reconfigure the virtual circuit identifier (VCI) and virtual path identifier (VPI).

```
[edit interfaces at-1/0/0]
user@host# set unit 62 vci 0.62
```

You also want to replace the IP address of the new interface with its own IP address.

```
[edit interfaces at-1/0/0]
user@host# edit unit 62
user@host# replace pattern 10.0.1.1 with 10.0.2.1
```

### Results

```
[edit]
show interfaces
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;
    }
  }
  unit 62 {
    point-to-point;
    vci 0.62;
    family inet {
      address 10.0.2.1/24;
    }
  }
}
```

### Example: Re-Using Configuration

If you need to make changes to the configuration of a device, you can always remove the original configuration settings using the **delete** command and add your new configuration settings using the **set** command. However, there are other ways of modifying a configuration that are more efficient and easier to use.

This example shows how to use the following configuration mode commands to update an existing configuration:

- **rename**—Rename an existing configuration setting, such as an interface name. This can be useful when you are adding new interfaces to a device.
- **copy**—Copy a configuration setting and the entire hierarchy of statements configured under that setting. Copying configuration statements is useful when you are configuring many physical or logical interfaces of the same type.
- **replace**—Make global changes to text patterns in the configuration. For example, if you consistently misspell a word common to the description statement for all of the interfaces on your device, you can fix this mistake with a single command.

- [Requirements on page 76](#)
- [Overview on page 76](#)
- [Configuration on page 76](#)

---

### Requirements

No special configuration beyond device initialization is required before configuring this example.

---

### Overview

During the first example in this topic, you will make the following configuration changes:

- Create a new interface with a description that contains a typing error.
- Copy the configuration from the interface that you created to create a new interface.
- Rename one of the interfaces that you created.
- Fix the typing error in the description for the interfaces that you created.

In the second, shorter example, you will experiment with some of the same commands under slightly different circumstances.

---

### Configuration

#### CLI Quick Configuration

This example does not use commands that are suitable for this section.

#### *Using the Copy, Rename, and Replace Commands to Modify a Loopback Interface Configuration*

#### Step-by-Step Procedure



---

**CAUTION:** If your existing configuration uses any of the loopback interface unit numbers used in this example, you must substitute different unused loopback interface unit numbers. Otherwise, following these steps could damage the existing operational status of your device.

---

To create and modify a configuration of a loopback interface using the **copy**, **rename**, and **replace** commands:

1. Create a new loopback interface unit number and include a description.

The mistakes in the spelling of loopback in the description are intentional.

```
[edit]
user@host# set interfaces lo0 unit 100 description "this is a lopbck interface"
```

2. Display the configuration for the loopback interface you have just added.

```
[edit]
user@host# show interfaces lo0 unit 100
description "this is a lopbck interface";
```

3. Duplicate the loopback interface you have just created, warts and all, from unit 100 to unit 101.

```
[edit]
user@host# copy interfaces lo0 unit 100 to unit 101
```

4. Display the configurations for loopback interfaces lo0 unit 100 and lo0 unit 101.

```
[edit]
user@host# show interfaces lo0 unit 100
description "this is a lopbck interface";
[edit]
user@host# show interfaces lo0 unit 101
description "this is a lopbck interface";
```

The **copy** command duplicates an interface including any child statements such as **description**.

5. Rename the loopback interface lo0 unit 100 to loopback interface lo0 unit 102.

```
[edit]
user@host# rename interfaces lo0 unit 100 to unit 102
```

6. Display the configuration for loopback interface lo0 unit 100.

[Warning: element unresolved in stylesheets: <\_nopagebreak> (in <step>). This is probably a new element that is not yet supported in the stylesheets.]

```
[edit]
user@host# show interfaces lo0 unit 100
[edit]
user@host#
```

You should not see any results from this command. The loopback interface lo0 unit 100 is now gone. The **rename** command replaces the configuration statement indicated with the new configuration.

7. Fix the misspelling of the word *loopback* in the descriptions for loopback interfaces lo0 unit 101 and lo0 unit 102.

```
[edit]
user@host# replace pattern lopbck with loopback
```

8. Display the configuration for loopback interfaces lo0 unit 101 and lo0 102 to verify that the word *loopback* is spelled correctly now.

```
[edit]
user@host# show interfaces lo0 unit 101
description "this is a loopback interface";
[edit]
user@host# show interfaces lo0 unit 102
description "this is a loopback interface";
```

The **replace** command replaces all instances of the pattern specified in the command, unless limited in some way. The next example in this topic shows one way to limit the effect of the **replace** command.

9. From configuration mode, use the **rollback** command to put the device's configuration back to the state it was in before you executed the previous steps.

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

**Results** From configuration mode, use the **show interfaces lo0 unit 101** and **show interfaces lo0 unit 102** commands to ensure that the device's configuration is back to the state it was in before you executed the steps in this example.

```
[edit]
user@host: show interfaces lo0 unit 101
[edit]
user@host#
```

You should not see any results from this command.

```
[edit]
user@host# show interfaces lo0 unit 102
[edit]
user@host#
```

You should not see any results from this command.



### Compare the Copy Command at the Top-Level Configuration Hierarchy Level

**Step-by-Step Procedure** The previous example shows the **copy**, **rename**, and **replace** commands at the **[edit interfaces interface-name unit logical-interface-number]** hierarchy level. This example shows how some of these commands work at the top level of the CLI configuration mode hierarchy.

The following example requires you to navigate to various levels in the configuration hierarchy. For information about navigating the CLI, see [“Using the CLI Editor in Configuration Mode” on page 114](#).

1. Create an Ethernet interface.

```
[edit]
user@host# set interfaces et-2/0/0 unit 0 family inet address 192.0.2.2
```

2. Copy the interface you just created to another interface.

```
[edit]
user@host# copy interfaces et-2/0/0 to et-2/1/0
```

Compare this **copy** command to the one in the previous example, where the **copy** command takes the keyword **unit** before the value to be copied:

```
[edit]
user@host# copy interfaces lo0 unit 100 to unit 101
```

Notice that the keyword **interfaces** is not repeated after the preposition **to** and before the value to be copied. This happens in some top-level statements with the **copy** command.



**TIP:** Similarly, in the **rename** command, you do not repeat the keyword part of the statement before the new identifier in some top-level statements.

3. Show your configuration so far.

```
[edit]
user@host# show interfaces
et-2/0/0 {
  unit 0 {
    family inet {
      address 192.0.2.2/32;
    }
  }
}
et-2/1/0 {
  unit 0 {
```

```
        family inet {
            address 192.0.2.2/32;
        }
    }
}
```

4. Replace the address for et-2/1/0 with another IP address.

```
[edit interfaces et-2/1/0 unit 0 family inet]
user@host# replace pattern 192.0.2.2 with 192.0.2.40
```

Notice that if you want to change only a specific occurrence of a pattern instead of all of them, you need to navigate down to that specific hierarchy level before using the **replace** command.

5. Show the interfaces again.

```
[edit]
user@host# show interfaces
et-2/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.2/32;
        }
    }
}
et-2/1/0 {
    unit 0 {
        family inet {
            address 192.0.2.40/32;
        }
    }
}
```

6. From configuration mode, use the **rollback** command to put the device's configuration back to the state it was in before you executed the previous steps.

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

**Results** From configuration mode, use the **show interfaces et-2/0/0** and **show interfaces et-2/1/0** commands to ensure that the device's configuration is back to the state it was in before you executed the steps in this example.

```
[edit]
user@host# show interfaces et-2/0/0
[edit]
user@host#
```

You should not see any results from this command.

```
[edit]
user@R1# show interfaces et-2/1/0
[edit]
user@host#
```

You should not see any results from this command.

## Inserting a New Identifier in a Junos OS Configuration

When configuring a device running Junos OS, 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 in a Junos Configuration

This example shows the use of the **insert** command.

Whereas a term added using the **set** command is placed at the end of the existing list of terms, you use the **insert** command to add a term in the order you specify. Specifying the order of statement is important in the cases in which the order of the statements matters because the configuration statements create a sequence that is analyzed in order.

As this example shows, you must create the term (or it must already exist), before you can place it using the **insert** command. The reference point for placing the term must also exist, for example, to place the term T1 before the term T2, both T1 and T2 must already exist, and be populated (Junos automatically removes empty terms).

- [Requirements on page 82](#)
- [Overview on page 82](#)
- [Configuration on page 83](#)

## Requirements

---

Before you can insert a term, you must configure an initial policy. To quickly configure the initial policy for this example, copy the following commands, paste them into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit policy-options]** hierarchy level, and then enter **commit** from configuration mode.

```
set policy-statement statics term term1 from route-filter 192.168.0.0/16 orlonger
set policy-statement statics term term1 from route-filter 224.0.0.0/3 orlonger
set policy-statement statics term term1 then reject
set policy-statement statics term term2 from protocol direct
set policy-statement statics term term2 then reject
set policy-statement statics term term3 from protocol static
set policy-statement statics term term3 then reject
set policy-statement statics term term4 then accept
```

Now check to verify you have the hierarchy correctly configured.

```
[edit policy-options]
user@host# show
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;
  }
}
```

## Overview

---

When configuring a device running Junos OS, you can enter most statements and identifiers in any order. However, there are a few cases, such as in routing policies or firewall filters, in which the order of the statements matters because the configuration statements create a sequence that is analyzed in order.

To modify a portion of the configuration in which the statement order matters, you must use the **insert** configuration mode command. If you use the **set** command instead, the added statement or identifier will be in the wrong place sequentially. The only other way

to get the terms of the command in the correct order is to dismantle the configuration and start over.

### Configuration

**CLI Quick Configuration** To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit policy-options] hierarchy level, and then enter commit from configuration mode.

```
[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
```

### Configuring to Insert Terms

**Step-by-Step Procedure** 1. Determine in what order the terms in your configuration need to go, both the original terms and the new terms you plan to add.

In the original configuration, the policy is named **statics** and there are four terms. Each of the first three terms matches on a different match criteria and the resulting matches are rejected. The last term accepts all the rest of the traffic.

In this example, you need to add two terms that eliminate additional types of traffic. Both these terms need to go before the last term in the original configuration.

2. Rename original term4 to term6.

```
[edit]
user@host# rename policy-options policy-statement statics term term4 to term
    term6
```

This step preserves the original last term, now renamed term6, as the last term.

3. Create a new term4.

```
[edit]
user@host# set policy-options policy-statement statics term term4 from protocol
    local
user@host# set policy-options policy-statement statics term term4 then reject
```

A new term is added that matches traffic from local system addresses and rejects it.

4. Create new term5.

```
[edit]
user@host# set policy-options policy-statement statics term term5 from protocol
aggregate
user@host# set policy-options policy-statement statics term term5 then reject
```

A new term is added that matches traffic from aggregate routes and rejects it.

5. Insert term4 after term3.

```
[edit]
user@host# insert policy-options policy-statement statics term term4 after term
term3
```

6. Insert term5 after term4.

```
[edit]
user@host# insert policy-options policy-statement statics term term5 after term
term4
```

## **Results**

```
[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;
}

```

## Renaming an Identifier in a Junos OS Configuration

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
```

## Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration

In a Junos OS configuration, you can deactivate statements and identifiers so 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** and **identifier** you specify must be at the current hierarchy level. When you deactivate a statement, that specific statement is ignored and is not applied at all when you issue a **commit** command.

To disable a statement, use the **disable** configuration mode command:

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 disable a functionality, it is activated when you issue a **commit** command but is treated as though it is down or administratively disabled.

## Example: Deactivating and Reactivating Statements and Identifiers in a Junos OS Configuration

This example shows a common use case in which the **deactivate** and **activate** configuration mode commands are used. It involves dual Routing Engines, master and backup, that have graceful Routing Engine switchover (GRES) configured. The software on both Routing Engines needs to be upgraded. This can easily be accomplished by deactivating GRES, updating the Routing Engines, and then reactivating GRES.



**NOTE:** You can also perform a similar upgrade using the same setup except that nonstop active routing (NSR) is configured instead of GRES. You would need to deactivate NSR and then upgrade the Routing Engines before reactivating NSR.

- [Requirements on page 86](#)
- [Overview on page 86](#)
- [Configuration on page 86](#)

---

## Requirements

This example requires the use of a device with dual Routing Engines that can be upgraded.

Before you begin this example, make sure that you have GRES configured.

---

## Overview

In this example, there are two Routing Engines. GRES is configured, and the Routing Engines need to be upgraded. To accomplish the upgrading, you need to deactivate the GRES feature, upgrade each of the Routing Engines, and then activate GRES again.

---

## Configuration

### *Configuring the Deactivation and Reactivation of GRES*

#### Step-by-Step Procedure

To deactivate and reactivate GRES for Routing Engine upgrade:

1. Show that GRES is enabled for the router.

```
[edit]
user@host# show chassis
chassis {
  redundancy {
    graceful-switchover;
  }
  fpc 2 {
    pic 0 {
      tunnel-services {
        bandwidth 1g;
      }
    }
  }
}
```

2. Deactivate GRES.

```
[edit]
user@host# deactivate chassis redundancy graceful-switchover
user@host# commit
```

3. Show that GRES is deactivated.



```
[edit]
user@host# show chassis
redundancy {
    inactive: graceful-switchover;
}
fpc 2 {
    pic 0 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}
```

4. Upgrade the Routing Engines one by one.

For instructions on upgrading Junos OS on dual Routing Engines, see *Installing the Software Package on a Device with Redundant Routing Engines*.

5. Reactivate GRES.

```
[edit]
user@host# activate chassis redundancy graceful-switchover
user@host# commit
```

**Results** Verify that GRES feature is activated again.

```
[edit]
user@host# show chassis
redundancy {
    graceful-switchover;
}
fpc 2 {
    pic 0 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}
```

## Using Global Replace in the Junos OS Configuration

You can make global changes to variables and identifiers in the Junos OS 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

*Table 10: 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 ( - ).
( )	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 ( " ").

*Table 10: Common Regular Expressions to Use with the replace Command (continued)*

Operator	Function
<code>\</code>	A backslash escapes special characters to suppress their special meaning. For example, <code>\.</code> matches <code>.</code> (period symbol).
<code>\n</code>	Back reference. Matches the <i>n</i> th group.
<code>&amp;</code>	Back reference. Matches the entire match.

The following table lists some replacement examples.

*Table 11: Replacement Examples*

Command	Result
<code>replace pattern myrouter with router1</code>	Match: <b>myrouter</b> Result: <b>router1</b>
<code>replace pattern "192.168\.(*)/24" with "10.2.\1/28"</code>	Match: <b>192.168.3.4/24</b> Result: <b>10.2.3.4/28</b>
<code>replace pattern "1.\1" with "abc&amp;def"</code>	Match: <b>1.1</b> Result: <b>abc1.1def</b>
<code>replace pattern 1.1 with "abc&amp;def"</code>	Match: <b>1#1</b> Result: <b>abc&amp;def</b>

### Example: Using Global Replace in a Junos OS Configuration—Using the `\n` Back Reference

This example shows how you can use a backreference to replace a pattern.

- [Requirements on page 89](#)
- [Overview on page 90](#)
- [Configuration on page 90](#)

#### Requirements

No special configuration beyond device initiation is required before configuring this example.

Before you begin, configure the following:

```
[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;  
    }  
  }  
}
```

To quickly configure this initial configuration, copy the following commands and paste them in a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level:

```
set interfaces xe-0/0/0 unit 0  
set interfaces fe-3/0/1 vlan-tagging  
set interfaces fe-3/0/1 unit 0 description "inet6 configuration IP: 2000::c0a8::1bf5"  
set interfaces fe-3/0/1 unit 0 vlan-id 100  
set interfaces fe-3/0/1 unit 0 family inet address 17.10.1.1/24  
set interfaces fe-3/0/1 unit 0 family inet6 address 2000::c0a8:1bf5/3
```

---

## Overview

One of the most useful features of regular expressions is the backreference. Backreferences provide a convenient way to identify a repeated character or substring within a string. Once you find the pattern, you can repeat it without writing it again. You refer to the previously captured pattern with just `\#` (where `#` is a numeral that indicates the number of times you want the pattern matched).

You can use backreferences to recall, or find, data and replace it with something else. In this way you can reformat large sets of data with a single replace command, thus saving you the time it would take to look for and replace the pattern manually.

---

## Configuration

### *Configuring a Replacement Using a Backreference in the Command*

#### Step-by-Step Procedure

To replace a pattern in a Junos OS configuration using a backreference:

- Use the **replace** command.

```
[edit]  
user@host# replace pattern pattern1 with pattern2
```

In this case, we want to replace `:1bf5` with `1bf5`.

```
[edit]  
user@host# replace pattern "(.*)1bf5" with "\11bf5"
```

Notice the backreference (`\1`), which indicates the pattern should be searched for and replaced only once.

### Results

Here is the resulting configuration:

```
[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;
    }
  }
}
```

In this example, the pattern `2000::c0a8:1bf5` is replaced with `2000::c0a8:1bf5` once.

## Example: Using Global Replace in a Junos OS Configuration—Replacing an Interface Name

This example shows how to replace an interface name globally in a configuration by using the **replace** command.

Using the **replace** command can be a faster and better way to change a configuration. For example, a PIC might be moved to another slot in a router, which changes the interface name. With one command you can update the whole configuration. Or you might want to quickly extend the configuration with other similar configurations, for example, similar interfaces.

By using a combination of the **copy** and **replace** commands, you can add to a configuration and then replace certain aspects of the newly copied configurations. The **replace** command works with regular expressions. Regular expressions are quick, flexible, and ubiquitous. You can fashion just about any pattern you might need to search for, and most programming languages support regular expressions.

- [Requirements on page 92](#)
- [Overview on page 92](#)
- [Configuration on page 92](#)

## Requirements

---

No special configuration beyond device initialization is required before configuring this example.

Before you begin, configure the following hierarchy on the router. To quickly configure this hierarchy, see [“CLI Quick Configuration” on page 92](#).

```
user@host# show interfaces
so-0/0/0 {
  dce;
}
user@host# show protocols
ospf {
  area 0.0.0.0 {
    interface so-0/0/0.0 {
      hello-interval 5;
    }
  }
}
```

## Overview

---

This example shows how to replace an interface name globally in a configuration by using the **replace** command. It is a simple example.

The previous configuration is the starting point for this configuration update. In the course of this example, you change the name of the initial interface throughout the configuration with one command.

## Configuration

---

### CLI Quick Configuration

To quickly configure the initial configuration for this example, copy the following commands, paste them into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste these commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.:

```
set interfaces so-0/0/0 dce
set protocols ospf area 0.0.0.0 interface so-0/0/0.0 hello-interval 5
```

### Configuring an Interface Name Change

#### Step-by-Step Procedure

To change an interface name:

1. Make sure that you are at the top of the configuration mode hierarchy.

```
user@host# top
```

2. Replace **so-0/0/0** with **so-1/1/0** using the **replace** command, which uses the **pattern** keyword.

```
user@host# replace pattern so-0/0/0 with so-1/1/0
```

### Results

After making the required changes, verify the configuration by using the **show interfaces** and **show protocols** configuration mode commands.

```
[edit]
user@host# show interfaces
so-1/1/0 {
    dce;
}
user@host# show protocols
ospf {
    area 0.0.0.0 {
        interface so-1/1/0.0 {
            hello-interval 5;
        }
    }
}
```

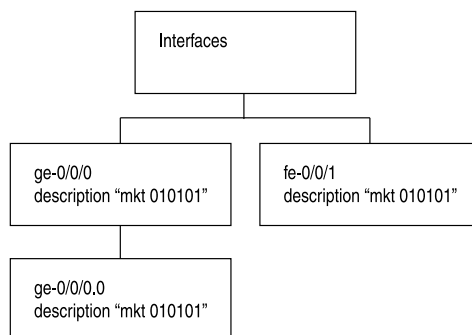
After you have confirmed that the configuration is correct, enter the **commit** command.

### Example: Using Global Replace in a Junos OS Configuration—Using the upto Option

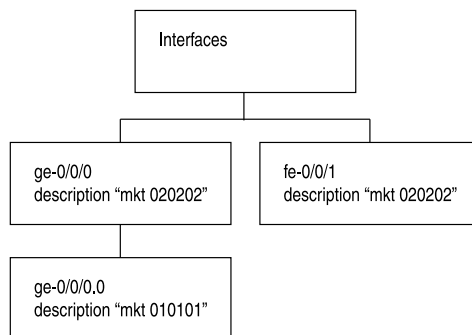
Consider the hierarchy shown in [Figure 6 on page 94](#). The text string **010101** appears in three places: the description sections of **ge-0/0/0**, **ge-0/0/0.0**, and **fe-0/0/1**. These three instances are three objects. The following example shows how you can use the **upto** option to perform replacements in a JUNOS configuration:

**Figure 6: Replacement by Object**

Current Configuration:

user@host > **replace pattern 01 with pattern 02 upto 2**

Resulting Configuration:



g017228

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).

```

user@host# show interfaces
ge-0/0/0 {
  description "mkt 010101"; #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 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

[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 OS Configuration

The Junos OS 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 the Junos OS 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)
```

**See Also** • [Disabling Inheritance of a Junos OS Configuration Group on page 149](#)

## Adding Comments in a Junos OS 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 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 OS Network Interfaces Library for Routing Devices*.

- [Adding Comments in the CLI on page 97](#)
- [Adding Comments in a File on page 98](#)

### Adding Comments in the CLI

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 ""
```

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.



**NOTE:** The Junos OS supports annotation up to the last level in the configuration hierarchy, including oneliners. However, annotation of parts (the child statements or identifiers within the oneliner) of the oneliner is not supported. For example, in the following sample configuration hierarchy, annotation is supported up to the level 1 parent hierarchy, but not supported for the **metric** child statement:

```
[edit protocols]
  isis {
    interface ge-0/0/0.0 {
      level 1 metric 10;
    }
  }
}
```

### Adding Comments in a File

---

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.

The following excerpt from a configuration example illustrates how to place and how not to place 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 */
    }
    dvmp-rib {
      import-rib inet.2;
      export-rib inet.2;
      /* A comment here is dropped */
    }
    /* A comment here is dropped */
  }
  /* A comment here is dropped */
}
```

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).

### Example: Including Comments in a Junos OS Configuration by Using the CLI

Adding comments to a Junos OS configuration makes the configuration file readable and more readily understood by users. Using the Junos OS CLI, you can include comments

as you configure by using the **annotate** statement. In this example, comments are added by using the CLI for an already existing configuration:

- [Requirements on page 99](#)
- [Overview on page 99](#)
- [Configuration on page 99](#)

## Requirements

No special configuration beyond device initialization is required before configuring this example.

Before you add a comment, you must configure the following hierarchy on the router.

To quickly configure the initial configuration for this example, copy the following command, paste it into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste this command into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
set protocols ospf area 0.0.0.0 interface so-0/0/0.0 hello-interval 5
```

Now verify that you have this hierarchy configured.

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

## Overview

When you add comments by using the CLI, you do so in configuration mode using the **annotate** statement. Each comment you add is associated with a statement at the current level. Each statement can have one single-line comment associated with it.

To configure the **annotate** statement, move to the level of the statement with which you want to associate a comment. To view the comments, go to the top of the configuration hierarchy and use the **show** command.

## Configuration

### CLI Quick Configuration

To quickly configure the comments for this example, copy the following commands, paste them into a text file, remove any line breaks and change any details necessary to match your network configuration, copy and paste the commands into the CLI, starting at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
edit protocols ospf
annotate area 0.0.0.0 "Backbone area configuration added June 15, 2018"
```

```
edit area 0.0.0.0
annotate interface so-0/0/0.0 "Interface from router sj1 to router sj2"
```

Notice that the commands are moving you down the hierarchy as you annotate different sections of the hierarchy.

### *Including Comments in the CLI Configuration Mode*

#### **Step-by-Step Procedure**

This procedure assumes that you have already configured the initial configuration.

To add comments to a configuration:

1. Move to the first hierarchy level to which you need to add a comment.

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

2. Add a comment to the **area** configuration statement by using the **annotate** statement.

```
[edit protocols ospf]
user@host# annotate area 0.0.0.0 "Backbone area configuration added June 15,
1998"
```

3. Move down a level to the **interface** configuration statement.

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

4. Add a comment to interface **so-0/0/0.0** by using the **annotate** statement.

```
[edit protocols ospf area 0.0.0.0]
user@host# annotate interface so-0/0/0.0 "Interface from router sj1 to router sj2"
```

### **Results**

Move to the top of the hierarchy and use the **show** command to see the comments you added. The comments precede the statement they are associated with.

```
[edit]
user@host# show protocols
ospf {
  /* Backbone area configuration added June 15, 2018 */
  area 0.0.0.0 {
    /* Interface from router sj1 to router sj2 */
    interface so-0/0/0.0 {
      hello-interval 5;
    }
  }
}
```

```
}  
}
```

After you have confirmed that the configuration is correct, enter the **commit** command.

---

## CLI Overview

---

The Junos command-line interface (CLI) is the software interface used to access your device. From here you configure the device, monitor its operations, and adjust the configuration as needed.

- [Introducing the Junos OS Command-Line Interface on page 101](#)
- [Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies on page 103](#)
- [Other Tools to Configure and Monitor Devices Running Junos OS on page 106](#)
- [Configuring Junos OS in a FIPS Environment on page 106](#)

### Introducing the Junos OS Command-Line Interface

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

The Junos OS CLI is a Juniper Networks-specific command shell that runs on top of a FreeBSD UNIX-based operating system kernel. Through the use of industry-standard tools and utilities, the CLI provides a powerful set of commands that you can use to monitor and configure devices running Junos OS.

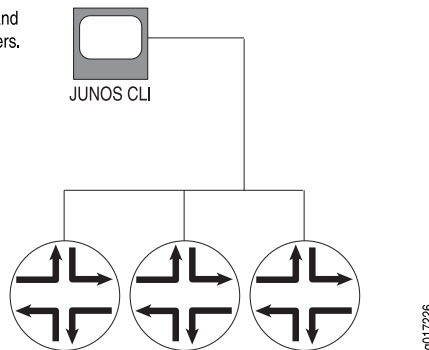
The Junos OS CLI has two modes:

- **Operational mode**—This mode displays the current status of the device. In operational mode, you enter commands to monitor and troubleshoot the Junos OS, devices, and network connectivity.
- **Configuration mode**—This mode enables you to configure the device. A configuration is stored as a hierarchy of configuration statements. In this mode, you enter statements to configure all properties of the device, 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 file enables you to make configuration changes without causing operational changes to the current operating configuration, called the *active configuration*. The router or switch does not implement the changes you added to the candidate configuration file until you commit them, which activates the configuration on the device. Candidate configurations enable you to alter your configuration without causing potential damage to your current network operations.

**Figure 7: Monitoring and Configuring Routers**

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



### Key Features of the CLI

The Junos OS CLI commands and statements follow a hierarchal organization and have a regular syntax. The Junos OS 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 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. This means that if you already are familiar with the Junos OS 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.

If you have typed the mandatory arguments for executing a command in the operational or configuration mode the CLI displays **<[Enter]>** as one of the choices when you type a question mark (?). This indicates that you have entered the mandatory arguments and can execute the command at that level without specifying any further options. Likewise, the CLI also displays **<[Enter]>** when you have reached a specific hierarchy level in the configuration mode and do not have to enter any more mandatory arguments or statements.

- Industry-standard technology—With FreeBSD UNIX as the kernel, a variety of UNIX utilities are available on the Junos OS 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 OS 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.

**See Also** • [Getting Started with the Junos OS Command-Line Interface on page 107](#)

## Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies

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

- [Junos OS CLI Command Modes on page 103](#)
- [CLI Command Hierarchy on page 104](#)
- [Configuration Statement Hierarchy on page 104](#)
- [Moving Among Hierarchy Levels on page 105](#)

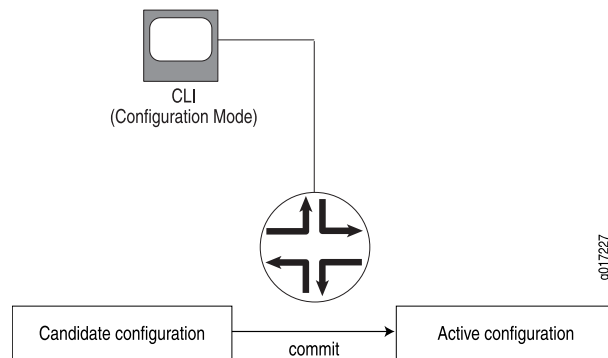
### Junos OS CLI Command Modes

The Junos OS 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 the Junos OS, devices, and network connectivity. To enter the operational mode, type the **CLI** command. The character “>” identifies operational mode. For example, `user@router>`
- **Configuration mode**—A configuration for a device running on Junos OS is stored as a hierarchy of statements. In configuration mode, you enter these statements to define all properties of the Junos OS, including interfaces, general routing information, routing protocols, user access, and several system and hardware properties. You enter the configuration mode by issuing the **configure** command from the operational mode. The character “#” identifies configuration mode. For example, `user@router#`

When you enter configuration mode, you are viewing and changing a file called the *candidate configuration*. The candidate configuration file enables you to make configuration changes without causing operational changes to the current operating configuration, called the *active configuration*. The router or switch does not implement the changes you added to the candidate configuration file until you commit them, which activates the configuration on the device. Candidate configurations enable you to alter your configuration without causing potential interruptions in your current network operations.

Figure 8: 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.

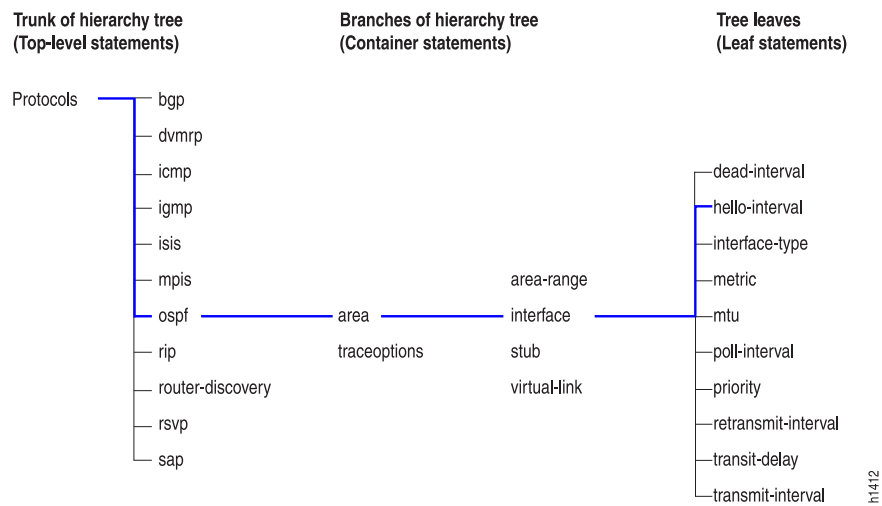
To execute a command, enter the full command name, starting at the top level of the hierarchy. For example, to display a brief view of the routes in the routing table, use the command **show route brief**.

### 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 the container and leaf statements together form the *configuration hierarchy*.

The following illustration shows 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 9: Configuration Statement Hierarchy Example



### Moving Among Hierarchy Levels

The following table shows the CLI commands used to navigate the levels of the configuration statement hierarchy.

Table 12: CLI Configuration Mode Navigation Commands

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

**See Also** • [Getting Started with the Junos OS Command-Line Interface on page 107](#)

## Other Tools to Configure and Monitor Devices Running Junos OS

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

- J-Web graphical user interface (GUI)—Available on select Juniper Networks devices, the J-Web 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*.
- Junos XML management protocol—The Junos XML management protocol allows you to monitor and configure Juniper Networks devices. 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 *Junos XML Management Protocol Developer Guide*.
- NETCONF Application Programming Interface (API)—You can also use the NETCONF XML management protocol to monitor and configure Juniper Networks routers. For more information, see the *NETCONF XML Management Protocol Developer Guide*.
- Junos OS 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 *Automation Scripting Feature 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 device. For more information about MIBs, see the *Network Management and Monitoring Guide*.

**See Also** • [Getting Started with the Junos OS Command-Line Interface on page 107](#)

## Configuring Junos OS in a FIPS Environment

Junos-FIPS enables you to configure a network of Juniper Networks devices 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 OS commands and statements have restrictions and some additional configuration statements are available. For more information, see the following resources:

- *Common Criteria and FIPS Certifications*—Provides links to guidelines for configuring devices running Junos OS so the secure environment complies with the requirements of public sector certifications such as Common Criteria (CC) and FIPS certification.
- [Compliance Advisor](#)—A Web application that provides regulatory compliance information about Common Criteria, FIPS, Homologation, ROHS2, and USGv6 for Juniper Networks products.

- See Also**
- [IPsec Requirements for Junos-FIPS](#)
  - [Configuring IPsec for Enabling Internal Communications Between Routing Engines for Junos OS in FIPS Mode](#)

- Related Documentation**
- [Day One: Exploring the Junos CLI](#)

## Getting Started: A Quick Tour of the CLI

---

The following topics can help you get started with the Junos OS CLI after installing Junos OS on the device, perform configuration changes, switch between operational mode and configuration mode, create a user account, execute some of the basic commands.



**NOTE:** If you need a more basic introduction to Junos OS, see the *Getting Started Guide*. For more in-depth information, as well as to learn how to use Junos OS with Juniper Networks devices, see the *Overview for Junos OS*. This Junos OS CLI Guide generally assumes you are at least familiar with the content in the other two guides, as well as Junos OS concepts and operation principles in general.

- [Getting Started with the Junos OS Command-Line Interface on page 107](#)
- [Switching Between Junos OS CLI Operational and Configuration Modes on page 109](#)
- [Using Keyboard Sequences to Move Around and Edit the Junos OS CLI on page 111](#)
- [Configuring a User Account on a Device Running Junos OS on page 112](#)
- [Using the CLI Editor in Configuration Mode on page 114](#)
- [Checking the Status of a Device Running Junos OS on page 116](#)
- [Rolling Back Junos OS Configuration Changes on page 119](#)
- [Configuring a Routing Protocol on page 120](#)

### Getting Started with the Junos OS Command-Line Interface

As an introduction to the Junos OS command-line interface (CLI), this topic describes what to do after installing Junos OS 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 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 any Juniper Networks devices running Junos OS.
- Before you begin, make sure your device hardware is set up and Junos OS 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.

To log in to a device 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 OS Release 7.6 or later, 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.

The CLI includes several ways to get help about commands. This section demonstrates some examples showing 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
tracert	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
```

- See Also**
- [Getting Online Help from the Junos OS Command-Line Interface on page 47](#)
  - [Examples: Using the Junos OS CLI Command Completion on page 29](#)

## Switching Between Junos OS CLI Operational and Configuration Modes

When you monitor and configure a device running Junos OS, 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 or hash sign (#).

To switch between operational mode and configuration mode:

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

```
--- JUNOS 17.2B1.8 built 2018-05-09 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 device while you are modifying the VRRP configuration for a backup device:



```
[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 ...
```

- See Also**
- [Understanding the Junos OS CLI Modes, Commands, and Statement Hierarchies on page 103](#)
  - [Getting Online Help from the Junos OS Command-Line Interface on page 47](#)

## Using Keyboard Sequences to Move Around and Edit the Junos OS CLI

You can use keyboard sequences in the Junos OS command-line interface (CLI) to move around and edit the command line. You can also use keyboard sequences to scroll through a list of recently executed commands. The following table lists some of the CLI keyboard sequences. They are the same as those used in Emacs.

*Table 13: CLI Keyboard Shortcuts*

Keyboard sequence	Action
Ctrl+b	Move the cursor back one character.
Esc+b or Alt+b	Move the cursor back one word.
Ctrl+f	Move the cursor forward one character.
Esc+f or Alt+f	Move the cursor forward one word.
Ctrl+a	Move the cursor to the beginning of the command line.
Ctrl+e	Move the cursor to the end of the command line.
Ctrl+h, Delete, or Backspace	Delete the character before the cursor.
Ctrl+d	Delete the character at the cursor.
Ctrl+k	Delete the all characters from the cursor to the end of the command line.
Ctrl+u or Ctrl+x	Delete the all characters from the command line.
Ctrl+w, Esc+Backspace, or Alt+Backspace	Delete the word before the cursor.

Table 13: CLI Keyboard Shortcuts (continued)

Keyboard sequence	Action
Esc+d or Alt+d	Delete the word after the cursor.
Ctrl+y	Insert the most recently deleted text at the cursor.
Ctrl+l	Redraw the current line.
Ctrl+p	Scroll backward through the list of recently executed commands.
Ctrl+n	Scroll forward through the list of recently executed commands.
Ctrl+r	Search the CLI history incrementally in reverse order for lines matching the search string.
Esc+/ or Alt+/	Search the CLI history for words for which the current word is a prefix.
Esc+. or Alt+	Scroll backward through the list of recently entered words in a command line.
Esc+ <i>number sequence</i> or Alt+ <i>number sequence</i>	Specify the number of times to execute a keyboard sequence.

- See Also**
- [Using Wildcard Characters in Interface Names on page 29](#)
  - [Using Global Replace in the Junos OS Configuration on page 87](#)

## Configuring a User Account on a Device Running Junos OS

This topic describes how to log on to a device running Junos OS 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 new 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 that 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).



**NOTE:** In Junos OS Release 12.2 and later, user account names can contain a period (.) in the name. For example, you can have a user account named **nathan.chen**. However, the username cannot begin or end with a period.

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 can 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.

You have successfully used the CLI to view the device status and perform a simple configuration change. See the related topics listed in this section for more information about the Junos OS CLI features.



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

- See Also**
- [Getting Online Help from the Junos OS Command-Line Interface on page 47](#)
  - [Displaying the Junos OS CLI Command and Word History on page 47](#)

## Using the CLI Editor in Configuration Mode

This topic describes some of the basic commands that you can use to enter configuration mode in the command-line interface (CLI) editor, navigate through the configuration hierarchy, get help, and commit or revert the changes that you make during the configuration session.

Task	Command/Statement	Example
Edit Your Configuration		

Task	Command/Statement	Example
<p>Enter configuration mode.</p> <p>When you first log in to the device, the device is in operational mode. You must explicitly enter configuration mode. When you do, the CLI prompt changes from <b>user@host&gt;</b> to <b>user@host#</b> and the hierarchy level appears in square brackets.</p>	<b>configure</b>	<pre>user@host&gt; configure [edit] user@host#</pre>
<p>Create a statement hierarchy.</p> <p>You can use the <b>edit</b> command to simultaneously create a hierarchy and move to that new level in the hierarchy. You cannot use the <b>edit</b> command to change the value of identifiers.</p>	<b>edit <i>hierarchy-level value</i></b>	<pre>[edit] user@host# edit security zones security-zone myzone [edit security zones security-zone myzone] user@host#</pre>
<p>Create a statement hierarchy and set identifier values.</p> <p>The <b>set</b> command is like <b>edit</b> except that your current level in the hierarchy does not change.</p>	<b>set <i>hierarchy-level value</i></b>	<pre>[edit] user@host# set security zones security-zone myzone [edit] user@host#</pre>
<b>Navigate the Hierarchy</b>		
Navigate down to an existing hierarchy level.	<b>edit <i>hierarchy-level</i></b>	<pre>[edit] user@host# edit security zones [edit security zones] user@host#</pre>
Navigate up one level in the hierarchy.	<b>up</b>	<pre>[edit security zones] user@host# up [edit security] user@host#</pre>
Navigate to the top of the hierarchy.	<b>top</b>	<pre>[edit security zones] user@host# top [edit] user@host#</pre>
<b>Commit or Revert Changes</b>		
Commit your configuration.	<b>commit</b>	<pre>[edit] user@host# commit commit complete</pre>

Task	Command/Statement	Example
<p>Roll back changes from the current session.</p> <p>Use the <b>rollback</b> command to revert all changes from the current configuration session. When you run the <b>rollback</b> command before exiting your session or committing changes, the software loads the most recently committed configuration onto the device. You must enter the <b>rollback</b> statement at the <b>edit</b> level in the hierarchy.</p>	<b>rollback</b>	<pre>[edit] user@host# rollback  load complete</pre>
<b>Exit Configuration Mode</b>		
Commit the configuration and exit configuration mode.	<b>commit and-quit</b>	<pre>[edit] user@host# commit and-quit  user@host&gt;</pre>
<p>Exit configuration mode without committing your configuration.</p> <p>You must navigate to the top of the hierarchy using the <b>up</b> or <b>top</b> commands before you can exit configuration mode.</p>	<b>exit</b>	<pre>[edit] user@host# exit  The configuration has been changed but not committed Exit with uncommitted changes? [yes,no] (yes)</pre>
<b>Get Help</b>		
Display a list of valid options for the current hierarchy level.	<b>?</b>	<pre>[edit ] user@host# edit security zones ?  Possible completions: &lt;[Enter]&gt; Execute this command &gt; functional-zone Functional zone &gt; security-zone Security zones   Pipe through a command [edit]</pre>

- See Also**
- [Understanding Junos OS CLI Configuration Mode on page 53](#)
  - [Entering and Exiting the Junos OS CLI Configuration Mode on page 59](#)
  - [Displaying the Current Junos OS Configuration on page 3](#)

## Checking the Status of a Device Running Junos OS

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

vpls	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

**See Also** • [Displaying the Junos OS CLI Command and Word History on page 47](#)



- [Managing Programs and Processes Using Junos OS Operational Mode Commands on page 41](#)
- [Viewing Files and Directories on a Device Running Junos OS on page 34](#)

## Rolling Back Junos OS Configuration Changes

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

The following procedure shows how to configure an SNMP health monitor on a device running Junos OS and then return to the most recently committed configuration that does not include the health monitor. When configured, the SNMP health monitor provides the 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 because SNMP has not been configured on the device.

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 device.

5. Enter 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. Enter 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.

**See Also** • [Returning to the Most Recently Committed Junos OS Configuration](#)

## Configuring a Routing Protocol

This topic provides a sample configuration that describes how to configure an OSPF backbone area that has two SONET interfaces.

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;
      }
    }
  }
}
```

- [Shortcut on page 120](#)
- [Longer Configuration on page 121](#)
- [Making Changes to a Routing Protocol Configuration on page 123](#)

---

### Shortcut

You can create a shortcut for this entire configuration with the following 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 previous 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 or hash 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 return up a level and view what you have done 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. Move up to the top level and review 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 device:

```

[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 recreate 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#
```

- See Also**
- [Displaying the Junos OS CLI Command and Word History on page 47](#)
  - [Interface Naming Conventions Used in the Junos OS Operational Commands on page 27](#)

---

## CLI Environment Settings

In operational mode, you can control the Junos OS command-line interface (CLI) environment and change the default CLI environment according to your specific requirements. For more information, see the following topics:

- [Controlling the Junos OS CLI Environment on page 126](#)
- [Setting the Junos OS CLI Screen Length and Width on page 129](#)
- [Example: Controlling the CLI Environment on page 129](#)
- [Example: Enabling Configuration Breadcrumbs on page 136](#)

### Controlling the Junos OS CLI Environment

In operational mode, you can control the Junos OS command-line interface (CLI) environment. For example, you can specify the number of 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



<code>prompt</code>	Set CLI command prompt string
<code>restart-on-upgrade</code>	Set whether CLI prompts to restart after software upgrade
<code>screen-length</code>	Set number of lines on screen
<code>screen-width</code>	Set number of characters on a line
<code>terminal</code>	Set terminal type
<code>timestamp</code>	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, your terminal type, screen length, and screen width are already set.

This chapter discusses the following topics:

- [Setting the Terminal Type on page 127](#)
- [Setting the CLI Prompt on page 127](#)
- [Setting the CLI Directory on page 127](#)
- [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 128](#)
- [Setting Command Completion on page 128](#)
- [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
```

The *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 Space 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
```

## Setting the Junos OS CLI Screen Length and Width

You can set the Junos OS 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 129](#)
- [Setting the Screen Width on page 129](#)

---

### 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 value of CLI screen width can be 0 or in the range of 40 through 1024. 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
```



**NOTE:** In Junos OS Release 13.2 and earlier, *width* can be 0 through 1024.

---

## Example: Controlling the CLI Environment

The following example shows you how to change the default CLI environment.

Changing the CLI environment is all about customizing the CLI window to fit your personal preferences. Use the settings discussed in this topic to make the CLI window look and behave according to what you find most convenient and efficient.

- [Requirements on page 130](#)
- [Overview on page 130](#)
- [Configuration on page 131](#)

## Requirements

---

No special configuration beyond device initialization is required before configuring this example.

Before starting this example, check what the default settings are. Use the **show cli** operational mode command.

```
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 66
CLI screen width set to 80
CLI terminal is 'xterm'
```

Is the prompt set to your *username@routername*? If not, exit the CLI and enter the operational mode again.

Is the CLI screen length set to 66 and the CLI screen width set to 80? If so, you can start the example. Otherwise, make these changes to the CLI settings:

```
user@host> set cli screen-length 66
Screen length set to 66 lines long
user@host> set cli screen-width 80
Screen width set to 80 columns wide
```

## Overview

---

To see a list of CLI environmental settings that you can change, use the **set cli ?** command.

```
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
```

This example focuses on three of these commands: **set cli screen-length**, **set cli screen-width**, and **set cli prompt**.

## Configuration

This configuration example has the following sections:

- [Configuring the CLI Prompt on page 131](#)
- [Configuring CLI Width on page 131](#)
- [Configuring CLI Length on page 132](#)
- [Return to the Default CLI Prompt on page 135](#)

### CLI Quick Configuration

To quickly configure this example, copy the following commands and paste them in a text file, remove any line breaks, change the values used to match your network configuration, and then copy and paste the commands into the CLI at the operational command prompt.

```
set cli prompt "router1-san-jose> "
set cli screen-width 110
set cli screen-length 45
```

### Configuring the CLI Prompt

#### Step-by-Step Procedure

The default CLI prompt is your *username@hostname*. But you can have any prompt you find useful.

To configure a different CLI prompt:

- Use the following operational mode command where *string* is the exact text you want to see at the command line.

```
set cli prompt "string"
```

For example, if "*string*" is "router1-san-jose> ", the command is as follows:

```
set cli prompt "router1-san-jose> "
router1-san-jose>
```

### Configuring CLI Width

#### Step-by-Step Procedure

How do you know what width works best for you? This example discusses how CLI width can affect what you see.

To configure a new default CLI width:

1. See what the current defaults are for the CLI environment.

```
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 80
CLI terminal is 'xterm'
router1-san-jose>
```

2. Examine the following output for the operational command **show class-of-service forwarding-class**.

The output from this command is wider than some and so illustrates a common problem with viewing output. If, for example, you have a relatively narrow window, command output might be displayed in overrun lines.

```
router1-san-jose> show class-of-service forwarding-class
```

Forwarding class	ID	Queue	Restricted queue	Fabric
priority	Policing priority	SPU priority	priority	
premium-rate	0	0	0	low
	normal		low	
medium-rate	1	1	1	low
	normal		low	
low-rate	2	2	2	low
	normal		low	
NC	3	3	3	low
	normal		low	
tunnel-rate	4	4	0	low
	normal		low	

The lines look to be intermingled and it is hard to read across to find the information you might be seeking.

3. Change the window width to 110 columns. Notice how the output of this command is much easier to read in the wider format.

```
router1-san-jose> set cli screen-width 110
```

```
router1-san-jose> show class-of-service forwarding-class
```

Forwarding class	ID	Queue	Restricted queue	Fabric priority	Policing priority	SPU priority
premium-rate	0	0	0	low	normal	low
medium-rate	1	1	1	low	normal	low
low-rate	2	2	2	low	normal	low
NC	3	3	3	low	normal	low
tunnel-rate	4	4	0	low	normal	low

### Configuring CLI Length

**Step-by-Step Procedure** You can set the length of the CLI screen the same way you set the width. To configure a new default CLI length:

1. See what the current defaults are for the CLI environment.

```
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 80
CLI terminal is 'xterm'
router1-san-jose>

```

2. Examine the output for the operational command **show version**.

```

Makefile          sync-dpm-sb.manifest
build             sync-equilibrium-sb.manifest
etc              sync-equilibrium2-sb.manifest
include          sync-hellopics-sb.manifest
jexample         sync-ipprobe-mt-sb.manifest
jnx-cc-routeservice-sb.manifest sync-ipprobe-sb.manifest
jnx-example-sb.manifest sync-ipsnooper-sb.manifest
jnx-flow-sb.manifest sync-monitube-sb.manifest
jnx-gateway-sb.manifest sync-monitube2-plugin-sb.manifest
jnx-ifinfo-sb.manifest sync-packetproc-sb.manifest
jnx-mspexamp1ed-sb.manifest sync-passthru-sb.manifest
jnx-msprsm-sb.manifest sync-policy-manager-sb.manifest
jnx-routeservice-sb.manifest sync-reassembler-sb.manifest
lib              sync-route-manager-sb.manifest
JnprFirewall-Proto.html Makefile.depend.octeon dfw_filter.proto
JnprFirewall.html      Makefile.depend.powerpc dfw_ifattach.proto
Makefile               Makefile.depend.xlr   dfw_policer.proto
Makefile.depend.arm    dfw.jsdl               dfw_stats.proto
Makefile.depend.host   dfw_bulk.proto
Makefile.depend.i386   dfw_common.proto

Trying 192.168.184.75...
Connected to spot-fxp0.englab.juniper.net.
Escape character is '^]'.
Unauthorized use is prohibited.

router1-san-jose> show version
Hostname: spot
Model: mx240
Junos: 14.2-20140710_ib_14_2_psd.1
JUNOS Base OS boot [14.2-20140710_ib_14_2_psd.1]
JUNOS Base OS Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Kernel Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Crypto Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Packet Forwarding Engine Support (M/T/EX Common)
[14.2-20140710_ib_14_2_psd.1]
JUNOS Packet Forwarding Engine Support (MX Common)
[14.2-20140710_ib_14_2_psd.1]
JUNOS Online Documentation [14.2-20140710_ib_14_2_psd.1]
JUNOS Services AACL Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Application Level Gateways [14.2-20140710_ib_14_2_psd.1]
JUNOS AppId Services [14.2-20140710_ib_14_2_psd.1]
JUNOS Border Gateway Function package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Captive Portal and Content Delivery Container package
[14.2-20140710_ib_14_2_psd.1]
JUNOS Services HTTP Content Management package [14.2-20140710_ib_14_2_psd.1]
JUNOS IDP Services [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Jflow Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services LL-PDF Container package [14.2-20140710_ib_14_2_psd.1]

```

```

JUNOS Services MobileNext Software package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Mobile Subscriber Service Container package
[14.2-20140710_ib_14_2_psd.1]
JUNOS Services NAT [14.2-20140710_ib_14_2_psd.1]
JUNOS Services PTSP Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services RPM [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Stateful Firewall [14.2-20140710_ib_14_2_psd.1]
JUNOS Voice Services Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Crypto [14.2-20140710_ib_14_2_psd.1]
JUNOS Services SSL [14.2-20140710_ib_14_2_psd.1]
JUNOS Services IPSec [14.2-20140710_ib_14_2_psd.1]
JUNOS platform Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Routing Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Runtime Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Web Management [14.2-20140710_ib_14_2_psd.1]
JUNOS py-base-i386 [14.2-20140710_ib_14_2_psd.1]

```

```
router1-san-jose>
```

The current length is 66 lines, which is close to the length of a typical monitor. But even though the output is long, it does not need all that space to be clearly seen in its entirety. In fact, it is harder to pick out just where the output starts in a screen this long.

3. Change the window width to 45 lines.

```
router1-san-jose> set cli screen-length 45
```

4. Now examine the output again.

```
router1-san-jose> show version
```

```

Hostname: spot
Model: mx240
Junos: 14.2-20140710_ib_14_2_psd.1
JUNOS Base OS boot [14.2-20140710_ib_14_2_psd.1]
JUNOS Base OS Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Kernel Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Crypto Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Packet Forwarding Engine Support (M/T/EX Common)
[14.2-20140710_ib_14_2_psd.1]
JUNOS Packet Forwarding Engine Support (MX Common)
[14.2-20140710_ib_14_2_psd.1]
JUNOS Online Documentation [14.2-20140710_ib_14_2_psd.1]
JUNOS Services AACL Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Application Level Gateways [14.2-20140710_ib_14_2_psd.1]
JUNOS AppId Services [14.2-20140710_ib_14_2_psd.1]
JUNOS Border Gateway Function package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Captive Portal and Content Delivery Container package
[14.2-20140710_ib_14_2_psd.1]
JUNOS Services HTTP Content Management package [14.2-20140710_ib_14_2_psd.1]
JUNOS IDP Services [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Jflow Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services LL-PDF Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services MobileNext Software package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Mobile Subscriber Service Container package
[14.2-20140710_ib_14_2_psd.1]

```



```
JUNOS Services NAT [14.2-20140710_ib_14_2_psd.1]
JUNOS Services PTSP Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services RPM [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Stateful Firewall [14.2-20140710_ib_14_2_psd.1]
JUNOS Voice Services Container package [14.2-20140710_ib_14_2_psd.1]
JUNOS Services Crypto [14.2-20140710_ib_14_2_psd.1]
JUNOS Services SSL [14.2-20140710_ib_14_2_psd.1]
JUNOS Services IPSec [14.2-20140710_ib_14_2_psd.1]
JUNOS platform Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Routing Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Runtime Software Suite [14.2-20140710_ib_14_2_psd.1]
JUNOS Web Management [14.2-20140710_ib_14_2_psd.1]
JUNOS py-base-i386 [14.2-20140710_ib_14_2_psd.1]

router1-san-jose>
```

With a shorter screen, you can easily see where the current output begins and ends.

### *Return to the Default CLI Prompt*

**Step-by-Step Procedure** To go back to the default prompt:

1. Exit the CLI.

```
router1-san-jose> exit
%
```

2. Enter the CLI operational mode again.

```
% cli
user@host>
```

## Example: Enabling Configuration Breadcrumbs

The output of **show configuration** operational mode command and **show configuration** mode commands can be configured to display configuration breadcrumbs that indicate the exact location in the hierarchy of the output being viewed.

Before enabling the configuration breadcrumbs feature, check the output of the **show configuration** command.

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

```
...
    }
  }
}
fe-4/1/2 {
  description "FA4/1/2: mxj1-mr6 (64.12.137.160/27) (T=bb1an, bbmail,
bbowmtc)";
  unit 0 {
    family inet {
      filter {
        output 151mj;
      }
      address 64.12.137.187/27 {
        vrrp-group 1 {
          virtual-address 64.12.137.189;
        }
      }
    }
  }
}
---(more 18%)-----
```

In the output, there is no clear indication about the section of the configuration being viewed.

To enable the configuration breadcrumbs feature:

1. Define a class at the **[edit system login]** hierarchy level.

```
[edit system login]
user@host# set class breadclass idle-timeout 10
```

2. Add a user to the defined login class to enable the breadcrumbs output view when this user enters the **show configuration** operational mode command.

```
[edit system login user user1]
user@host# set class breadclass
```

3. Configure the **configuration-breadcrumbs** statement at the **[edit system login class <class name>]** hierarchy level.

```
[edit system login class breadclass]
user@host# set configuration-breadcrumbs
```

4. Confirm the configuration.

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

On enabling configuration breadcrumbs in the CLI, User1 (the user added to the login class) can verify the feature in the output by entering the **show configuration** command.

```
user1@host> show configuration
```

```
...
    }
  }
}
fe-4/1/2 {
  description "FA4/1/2: mxxj1-mr6 (64.12.137.160/27) (T=bb1an, bbmail,
bbowmtc)";
  unit 0 {
    family inet {
      filter {
        output 151mj;
      }
      address 64.12.137.187/27 {
        vrrp-group 1 {
          virtual-address 64.12.137.189;
        }
      }
    }
  }
}
---(more 18%)---[groups main interfaces fe-4/1/2 unit 0 family inet address
64.12.137.187/27 vrrp-group 1]---
```

The new output indicates the exact location of the configuration hierarchy being viewed. User1 is currently viewing the interface configuration of a group.



**NOTE:** If you are enabling configuration breadcrumbs for your own user account, you should log out and log in again to see the changes.

See Also • [class](#)

## Configure Command Overview

The **configure** command is used to enter the CLI configuration mode. It can also be used to gather other information, such as other users currently in configuration mode.

- [Forms of the configure Command on page 138](#)
- [Using the configure Command on page 139](#)
- [Using the configure exclusive Command on page 140](#)
- [Updating the configure private Configuration on page 141](#)

## Forms of the configure Command

The Junos OS 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 are managing the network and device configuration.

*Table 14: Forms of the configure Command*

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

Table 14: Forms of the configure Command (continued)

Command	Edit Access	Commit Access
<b>configure private</b>	<ul style="list-style-type: none"> <li>Multiple users can edit the configuration at the same time.</li> <li>Each user has a private candidate configuration to edit independently of other users.</li> <li>When multiple users enter conflicting configurations, the first commit operation takes precedence over subsequent commit operations.</li> </ul>	<ul style="list-style-type: none"> <li>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.</li> <li>If the configuration has been modified by another user, you can merge the modifications into your private candidate configuration and attempt to commit again.</li> </ul>

- See Also**
- [Committing a Junos OS Configuration on page 172](#)
  - [Displaying Users Currently Editing the Junos OS Configuration on page 68](#)
  - [Displaying set Commands from the Junos OS Configuration on page 8](#)

## Using the configure Command

You can use the **configure** command not only to enter the CLI configuration mode but also to gather other information, such as whether other users are currently in configuration mode.

Up to 32 users can be in configuration mode simultaneously, and they all can make changes to the configuration at the same time. When you commit changes to the configuration, you may be committing a combination of changes you and other users have made. For this reason, you will want to keep track of who is in configuration mode with you.

To see other users currently logged onto the same device in configuration mode:

- Use the **configure** command to enter the CLI configuration mode.

If there are other users, the message displayed indicates who the users are and what portion of the configuration each person is viewing or editing.

```

user@host> configure
Entering configuration mode
Current configuration users:
root terminal p3 (pid 1088) on since 2018-05-13 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#

```

Notice also that If, when you enter configuration mode, the configuration contains changes that have not been committed, another message is displayed:

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

This tells you that another user has already made changes to the configuration.

## 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 commit 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 identifies the user and the portion of the configuration that the user is viewing or editing:

```
user@host> configure
Entering configuration mode
Users currently editing the configuration:
root terminal p3 (pid 1088) on since 2018-10-30 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 OS 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 OS will roll back any uncommitted changes.

If you enter configuration mode with the **configure exclusive** command, and issue **commit confirmed**, but do not confirm the commit, automatic rollback is triggered. Once automatic rollback occurs, the management daemon (MGD) removes the exclusive lock from your session and as a result, the error message “access has been revoked” is displayed. This is because the session is no longer an exclusive session.

```
user@host>configure exclusive
warning: uncommitted changes will be discarded on exit
Entering configuration mode
[edit]
user@host# commit confirmed 1
commit confirmed will be automatically rolled back in 1 minutes unless confirmed
commit
# commit confirmed will be rolled back in 1 minute
Commit was not confirmed; automatic rollback complete.
[edit]
user@host# commit
error: access has been revoked.
user@host# commit check
error: access has been revoked.
```

If you initiate a configure exclusive session, issue **commit confirmed** and confirm the commit, the exclusive lock is retained in your session

```
user@host> configure exclusive
warning: uncommitted changes will be discarded on exit
Entering configuration mode
[edit]
user@host# commit confirmed 1
commit confirmed will be automatically rolled back in 1 minutes unless confirmed
commit complete
# commit confirmed will be rolled back in 1 minute
[edit]
user@host# commit
commit complete
[edit]
user@host# commit
commit complete
```

**See Also** • [Adding Junos OS Configuration Statements and Identifiers on page 69](#)

## 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#
```



**NOTE:** Junos OS does not support using **configure private** mode to configure statements corresponding to third-party YANG data models, for example, OpenConfig or custom YANG data models.

---

## Using Configuration Groups to Quickly Configure Devices

---

Configuration groups are used to set up and apply common elements that are reused within the same configuration.

- [Understanding Junos OS Configuration Groups on page 143](#)
- [Creating a Junos OS Configuration Group on page 144](#)
- [Applying a Junos OS Configuration Group on page 146](#)
- [Example: Creating and Applying Junos OS Configuration Groups on page 147](#)
- [Example: Creating and Applying Configuration Groups on a TX Matrix Router on page 148](#)
- [Disabling Inheritance of a Junos OS Configuration Group on page 149](#)
- [Using the junos-defaults Configuration Group on page 151](#)
- [Using Wildcards with Configuration Groups on page 152](#)
- [Improving Commit Time When Using Configuration Groups on page 155](#)
- [Example: Configuring Sets of Statements with Configuration Groups on page 155](#)
- [Example: Configuring Interfaces Using Configuration Groups on page 156](#)
- [Example: Configuring a Consistent IP Address for the Management Interface Using Configuration Groups on page 159](#)
- [Example: Configuring Peer Entities Using Configuration Groups on page 160](#)
- [Example: Establishing Regional Configurations Using Configuration Groups on page 162](#)
- [Example: Configuring Wildcard Configuration Group Names on page 163](#)



- [Example: Referencing the Preset Statement from the Junos OS defaults Group on page 164](#)
- [Example: Viewing Default Statements That Have Been Applied to the Configuration on page 165](#)
- [Setting Up Routing Engine Configuration Groups on page 166](#)
- [Using Conditions to Apply Configuration Groups on page 168](#)
- [Example: Configuring Conditions for Applying Configuration Groups on page 168](#)

## Understanding Junos OS Configuration Groups

This topic provides an overview of the configuration groups feature and the inheritance model in Junos OS.

- [Configuration Groups Overview on page 143](#)
- [Inheritance Model on page 143](#)
- [Configuring Configuration Groups on page 144](#)

---

### Configuration Groups Overview

The configuration groups feature in Junos OS 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 OS. 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 the Junos OS 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 does not need to 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 where the configuration statements contained in a configuration group are needed.

### Creating a Junos OS 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 OS. 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.

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.



**NOTE:** The configuration statements **re0** and **re1** are case sensitive.

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. .

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.



**NOTE:** The management Ethernet interface used for the TX Matrix Plus router, T1600 or T4000 routers in a routing matrix, and PTX Series Packet Transport Routers is **em0**. Junos OS automatically creates the router's management Ethernet interface, **em0**.

- See Also**
- [Using the junos-defaults Configuration Group on page 151](#)
  - [Applying a Junos OS Configuration Group on page 146](#)
  - *User Access and Authentication Feature Guide*

## Applying a Junos OS Configuration Group

To have the Junos OS 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.



**NOTE:** The management Ethernet interface used for the TX Matrix Plus router, T1600 routers in a routing matrix, and PTX Series Packet Transport Switches, is **em0**.

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 ];
      }
    }
  }
}
```

```
}
```

When you configure a group defined for the root level—that is, in the default logical system—you cannot successfully apply that group to a nondefault logical system under the `[edit logical-systems logical-system-name]` hierarchy level. Although the router accepts the commit if you apply the group, the configuration group does not take effect for the nondefault logical system. You can instead create an additional configuration group at the root level and apply it within the logical system. Alternatively, you can modify the original group so that it includes configuration for both the default and nondefault logical system hierarchy levels.

### Example: Creating and Applying Junos OS Configuration Groups

In this example, the SNMP configuration is divided between the group **basic** and the normal configuration hierarchy.

There are several advantages to placing the system-specific configuration (SNMP contact) into a configuration group and thus separating it from the normal configuration hierarchy—you 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;
  }
}
```

**See Also** • [Disabling Inheritance of a Junos OS Configuration Group on page 149](#)

## Example: Creating and Applying Configuration Groups on a TX Matrix Router

The following example shows how to configure and apply configuration groups on a TX Matrix Router.

```
[edit]
groups {
  re0 { # Routing Engine 0 on TX Matrix router
    system {
      host-name hostname;
      backup-router ip-address;
    }
    interfaces {
      fxp0 {
        unit 0 {
          family inet {
            address ip-address;
          }
        }
      }
    }
  }
  re1 { # Routing Engine 1 on TX Matrix router
    system {
      host-name hostname;
      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 hostname;
      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 hostname;
        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 OS 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 for the **hold-time** and **link-mode** statements.

```

[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"
  }
}

```

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;
  }
}
```





**NOTE:** Using the `display inheritance | except ##` option removes all the lines with `##`. Therefore, you might also not be able to view information about passwords and other important data where `##` is used. To view the complete configuration details with all the information without just the comments marked with `##`, use the `no-comments` option with the `display inheritance` command:

```
[edit]
user@host# show | display inheritance no-comments
snmp {
  location "West of Nowhere";
  contact "My Engineering Group";
  community BasicAccess {
    authorization read-only;
  }
}
```

**See Also** • [Applying a Junos OS Configuration Group on page 146](#)

## Using the junos-defaults Configuration Group

Junos OS 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.

## 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. Within a configuration group, an interface name that includes a character class 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 close bracket

[1] 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.



**NOTE:** If used inside the **groups** hierarchy, an identifier name cannot start with < unless you are defining a wildcard statement, in which case the wildcard statement must have a closing >.

Wildcarding in configuration groups follows the same rules, but < and > have a special meaning when used under the **groups** hierarchy. In the **groups** hierarchy, any term using a wildcard pattern must be enclosed in angle brackets <pattern> to differentiate it from other wildcarding in the configuration file.

```
[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 previous example, the expression <so-\*> passes its **sonet-options** statement to any interface that matches the expression **so-\***.

The following example shows how to specify a range of interfaces:

```
[edit]
groups {
  gigabit-ethernet-interfaces {
    interfaces {
      "<ge-1/2/[5-8]>" {
        description "These interfaces reserved for Customer ABC";
      }
    }
  }
}
```

Angle brackets allow you to pass normal wildcarding through without modification. In any 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;
    }
  }
}
```

```
    }
}
```

## Improving Commit Time When Using Configuration Groups

Configuration groups are used for applying configurations across other hierarchies without re-entering configuration data. Some configuration groups specify every configuration detail. Other configuration groups make use of wildcards to configure ranges of data, without detailing each configuration line. Some configurations have an inheritance path that includes a long string of configurations to be applied.

When a configuration that uses configuration groups is committed, the commit process expands and reads all the configuration data of the group into memory to apply the configurations as intended. The commit performance can be negatively impacted if many configuration groups are being applied, especially if the configuration groups use wildcards extensively.

If your system uses many configuration groups that use wildcards, you can configure the **persist-groups-inheritance** statement at the **[edit system commit]** hierarchy level to improve commit time performance.

Using this option allows the system to build the inheritance path for each configuration group inside the database, rather than in the process memory. This can improve commit time performance. However, it can also increase the database size by up to 22 percent.

## Example: 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;
  }
}
```

## Example: Configuring Interfaces Using 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;
      }
    }
  }
}
```

---

Copyright © 2019, Juniper Networks, Inc. 157

```
##
## "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;
    }
  }
}
```

**See Also** • [Interface Naming Conventions Used in the Junos OS Operational Commands on page 27](#)



## Example: Configuring a Consistent IP Address for the Management Interface Using Configuration Groups

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 a routing matrix composed of the TX Matrix router, this feature is applicable to the switch-card chassis (SCC) only. Likewise, on a routing matrix composed of a TX Matrix Plus router, this feature is applicable to the switch-fabric chassis (SFC) only.

**NOTE:**

- If you configure the same IP address for a management interface or internal interface such as fxp0 and an external physical interface such as ge-0/0/1, when graceful Routing Engine switchover (GRES) is enabled, the CLI displays an appropriate commit error message that identical addresses have been found on the private and public interfaces. In such cases, you must assign unique IP addresses for the two interfaces that have duplicate addresses.
- The management Ethernet interface used for the TX Matrix Plus router, T1600 routers in a routing matrix, and PTX Series Packet Transport Routers, is em0. Junos OS automatically creates the router's management Ethernet interface, em0.

---

## Example: Configuring Peer Entities Using Configuration Groups

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 {
        ##
        ## "igether-options" was inherited from group "some-isp"
        ##
        igether-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;
  }
}
```

## Example: Establishing Regional Configurations Using Configuration Groups

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;
  }
}
}

```

### Example: Configuring Wildcard Configuration Group Names

Wildcards are configuration group names that use special characters to create a pattern that can be applied to multiple statements. Wildcards are useful for copying one set of configuration options to many of different configuration groups. It is important to set up your wildcard name properly to ensure that the wildcard configuration options get copied to the appropriate configuration groups.

In this example, you configure different values for the `<*-major>` and `<*-minor>` wildcard groups under the `label-switched-path` statement. The asterisk (\*) character represents a section of the wildcard name that can match any string of characters. For example, the configuration options under `label-switched-path <*-major>` are passed onto `label-switched-path metro-major` and any other `label-switched-path` configuration group containing `-major` in its name.

```

[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;
        }
    }
}
```

### Example: Referencing the Preset Statement from the Junos OS defaults Group

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 my-rule term my-term 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.

To view the complete configuration information without the comments marked with **##**, use the **no-comments** option with the **display inheritance** command.

## Setting Up Routing Engine Configuration Groups

In a router with two Routing Engines, one configuration should be shared between both Routing Engines. This ensures that both Routing Engine configurations are identical. Within this configuration, create two Routing Engine groups, one for each Routing Engine. Within these groups, you specify the Routing Engine-specific parameters.

For more information about the initial configuration for redundant Routing Engine systems and the re0 group, see *High Availability Feature Guide*.

1. Create the configuration group **re0**. The **re0** group is a special group designator that is only used by **RE0** in a redundant routing platform.

```
[edit]
root# set groups re0
```

2. Navigate to the **groups re0** level of the configuration hierarchy.

```
[edit]
root# edit groups re0
```

3. Specify the device hostname.

```
[edit groups re0]
root# set system host-name host-name
```



**NOTE:** The hostname specified in the device configuration is not used by the DNS server to resolve to the correct IP address. This hostname is used to display the name of the Routing Engine in the CLI. For example, the hostname appears at the command-line prompt when you are logged in to the CLI:

```
user-name@host-name>
```

4. Configure the IP address and prefix length for the device Ethernet interface.
  - For all devices *except* the TX Matrix Plus router, T1600 or T4000 routers in a routing matrix, and PTX Series Packet Transport Routers:

```
[edit]
root@# set interfaces fxp0 unit 0 family inet address address/prefix-length
```

- For the TX Matrix Plus router, T1600 or T4000 routers in a routing matrix only, and PTX Series Packet Transport Routers:

```
[edit]
root@# set interfaces em0 unit 0 family inet address address/prefix-length
```



To use **em0** as an out-of-band management Ethernet interface, you must configure its logical port, **em0.0**, with a valid IP address.

- For a T1600 standalone router (not connected to a TX Matrix Plus router and not in a routing matrix):

```
[edit]
root@# set interfaces fxp0 unit 0 family inet address address/prefix-length
```

5. Return to the top level of the hierarchy.

```
[edit groups re0]
root# top
```

6. Create the configuration group **re1**.

```
[edit]
root# set groups re1
```

7. Navigate to the **groups re1** level of the configuration hierarchy.

```
[edit]
root# edit groups re1
```

8. Specify the device hostname.

```
[edit groups re1]
root# set system host-name host-name
```

9. Configure the IP address and prefix length for the device Ethernet interface.

- For all devices *except* the TX Matrix Plus router, T1600 or T4000 routers in a routing matrix, and PTX Series Packet Transport Routers:

```
[edit]
root@# set interfaces fxp0 unit 0 family inet address address/prefix-length
```

- For the TX Matrix Plus router and T1600 or T4000 routers in a routing matrix only:

```
[edit]
root@# set interfaces em0 unit 0 family inet address address/prefix-length
```

To use **em0** as an out-of-band management Ethernet interface, you must configure its logical port, **em0.0**, with a valid IP address.

- For a T1600 standalone router (not connected to a TX Matrix Plus router, and not in a routing matrix):

```
[edit]
root@# set interfaces fxp0 unit 0 family inet address address/prefix-length
```

10. Return to the top level of the hierarchy.

```
[edit groups re0]  
root# top
```

11. Specify the group application order.

```
[edit]  
root# set apply-groups [ re0 re1 ]
```

## Using Conditions to Apply Configuration Groups

You can use the **when** statement at the **[edit groups group-name]** hierarchy level to define conditions under which a configuration group should be applied.

You can configure a group to be applied based on the type of chassis, model, or Routing Engine, virtual chassis member, cluster node, and start and optional end time of day or date.

For example, you could use the **when** statement to create a generic configuration group for each type of node and then apply the configuration based on certain node properties, such as chassis or model.

## Example: Configuring Conditions for Applying Configuration Groups

This example shows how to configure conditions under which a specified configuration group is to be applied.

- [Requirements on page 168](#)
- [Overview on page 168](#)
- [Configuration on page 169](#)

### Requirements

---

No special configuration beyond device initialization is required before you configure this example.

### Overview

---

You can configure your group configuration data at the **[edit groups group-name]** hierarchy level, then use the **when** statement to have the group applied based on conditions including: Type of chassis, model, routing-engine, virtual chassis member, cluster node, and start and optional end time of day or date.

If you specify multiple conditions in a single configuration group, all conditions must be met before the configuration group is applied.

You can specify the start time or the time duration for the configuration group to be applied. If only the start time is specified, the configuration group is applied at the specified time and it remains in effect until the time is changed. If the end time is specified, then

on each day, the applied configuration group is started and stopped at the specified times.

This example sets conditions in a configuration group, **test1**, such that this group is applied only when all of the following conditions are met: the router is a model MX240 router with chassis type LCC0, with a Routing Engine operating as RE0, is member0 of the virtual chassis on node0, and the configuration group will only be in effect from 9:00 a.m. until 5:00 p.m. each day.

### Configuration

#### CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set groups test1 when model mx240
set groups test1 when chassis lcc0
set groups test1 when routing-engine re0
set groups test1 when member member0
set groups test1 when node node0
set groups test1 when time 9 to 5
```

#### Step-by-Step Procedure

To configure conditions for configuration group **test1**:

1. Set the condition that identifies the model MX240 router.

```
[edit groups test1 when]
user@host# set model mx240
```

2. Set the condition that identifies the chassis type as LCC0.

```
[edit groups test1 when]
user@host# set chassis lcc0
```

3. Set the condition that identifies the Routing Engine operating as RE0.

```
[edit groups test1 when]
user@host# set routing-engine re0
```

4. Set the condition that identifies the virtual chassis **member0**.

```
[edit groups test1 when]
user@host# set member member0
```

5. Set the condition that identifies the cluster **node0**.

```
[edit groups test1 when]
```

```
user@host# set node node0
```

6. Set the condition that applies the group only between the hours of 9:00 a.m. and 5:00 p.m. daily.

```
[edit groups test1 when]
user@host# set time 9 to 5
```



**NOTE:** The syntax for specifying the time is: `time <start-time> [to <end-time>]` using the time format `yyyy-mm-dd.hh:mm`, `hh:mm`, or `hh`.

7. Commit the configuration.

```
user@host# commit
```

**Results** From configuration mode, confirm your configuration by entering the **show groups test1** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show groups test1
when {
  time 9 to 5;
  chassis lcc0;
  model mx240;
  routing-engine re0;
  member member0;
  node node0;
}
```

### *Verification*

#### *Checking Group Inheritance with Conditional Data*

**Purpose** Verify that conditional data from a configuration group is inherited when applied.

**Action** The **show | display inheritance** operational command can be issued with the **when** data to display the conditional inheritance. Using this example, you could issue one of these commands to determine that the conditional data was inherited:

```
user@host> show | display inheritance when model mx240
user@host> show | display inheritance when chassis lcc0
user@host> show | display inheritance when routing-engine re0
user@host> show | display inheritance when member member0
user@host> show | display inheritance when node node0
```

```
user@host> show | display inheritance when time 9 to 5
```

---

## Committing a Configuration

---

To commit a configuration, the **commit** configuration mode command enables you to save the Junos OS configuration changes to the configuration database and to activate the configuration on the device.

- [Junos OS Commit Model for Configurations on page 171](#)
- [Committing a Junos OS Configuration on page 172](#)
- [Commit Operation When Multiple Users Configure the Software on page 174](#)
- [Commit Preparation and Activation Overview on page 175](#)
- [Committing Junos OS Configurations in Two Steps: Preparation and Activation on page 177](#)
- [Activating a Junos OS Configuration but Requiring Confirmation on page 178](#)
- [Scheduling a Junos OS Commit Operation on page 180](#)
- [Monitoring the Junos OS Commit Process on page 181](#)
- [Adding a Comment to Describe the Committed Configuration on page 182](#)
- [Junos OS Batch Commits Overview on page 183](#)
- [Example: Configuring Batch Commit Server Properties on page 183](#)
- [Backing Up the Committed Configuration on the Alternate Boot Drive on page 192](#)

## Junos OS Commit Model for Configurations

The device configuration is saved using a commit model—a candidate configuration is modified as desired and then committed to the system. When a configuration is committed, the device checks the configuration for syntax errors, and if no errors are found, the configuration is saved as **juniper.conf.gz** and activated. The formerly active configuration file is saved as the first rollback configuration file (**juniper.conf.1.gz**), and any other rollback configuration files are incremented by 1. For example, **juniper.conf.1.gz** is incremented to **juniper.conf.2.gz**, making it the second rollback configuration file. The device can have a maximum of 49 rollback configurations (numbered 1 through 49) saved on the system.

On the device, the active configuration file and the first three rollback files (**juniper.conf.gz.1**, **juniper.conf.gz.2**, **juniper.conf.gz.3**) are located in the **/config** directory. If the file **rescue.conf.gz** is saved on the system, this file should also be saved in the **/config** directory. The factory default files are located in the **/etc/config** directory.

There are two mechanisms used to propagate the configurations between Routing Engines within a device:

- **Synchronization:** Propagates a configuration from one Routing Engine to a second Routing Engine within the same device chassis.



**NOTE:** The QFX3500 switch has only one Routing Engine.

To synchronize configurations, use the **commit synchronize** CLI command. If one of the Routing Engines is locked, the synchronization fails. If synchronization fails because of a locked configuration file, you can use the **commit synchronize force** command. This command overrides the lock and synchronizes the configuration files.

- **Distribution:** Propagates a configuration across the routing plane on a multichassis device. Distribution occurs automatically. There is no user command available to control the distribution process. If a configuration is locked during a distribution of a configuration, the locked configuration does not receive the distributed configuration file, so the synchronization fails. You need to clear the lock before the configuration and resynchronize the routing planes.



**NOTE:** When you use the **commit synchronize force** CLI command on a multichassis platform, the forced synchronization of the configuration files does not affect the distribution of the configuration file across the routing plane. If a configuration file is locked on a device remote from the device where the command was issued, the synchronization fails on the remote device. You need to clear the lock and reissue the **synchronize** command.

---

### Committing a Junos OS Configuration and Exiting Configuration Mode

To save Junos OS configuration changes, activate the configuration on the device and exit configuration mode, using 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>
```



**NOTE:** We do not recommend performing a commit operation on the backup Routing Engine when graceful Routing Engine switchover is enabled on the router.

**See Also** • *Configuring Junos OS for the First Time on a Device with a Single Routing Engine*

### Committing a Junos OS Configuration

To save Junos OS configuration changes to the configuration database and to activate the configuration on the device, use the **commit** configuration mode command. You can issue the **commit** command from any hierarchy level:

```
[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 device configuration.

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

A configuration commit can fail for any of the following reasons:

- The configuration includes incorrect syntax, which causes the commit check to fail.
- The candidate configuration that you are trying to commit is larger than 700 MB.
- The configuration is locked by a user who entered the **configure exclusive** command.

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.

The uncommitted, candidate configuration file is **/var/run/db/juniper.db**. It is limited to 700 MB. If the commit fails with a message **configuration database size limit exceeded**, view the file size from configuration mode by entering the command **run file list /var/run/db detail**. You can simplify the configuration and reduce the file size by creating configuration groups with wildcards or defining less specific match policies in your firewall filters.



**NOTE:** CLI commit-time warnings displayed for configuration changes at the [edit interfaces] hierarchy level are removed and are logged as system log messages.

This is also applicable to VRRP configuration at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* family (*inet* | *inet6*) address *address*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family (*inet* | *inet6*) address *address*]

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:**

- We do not recommend performing a commit operation on the backup Routing Engine when graceful Routing Engine switchover is enabled on the device.
- If you configure the same IP address for a management interface or internal interface such as fxp0 and an external physical interface such as ge-0/0/1, when graceful Routing Engine switchover (GRES) is enabled, the CLI displays an appropriate commit error message that identical addresses have been found on the private and public interfaces. In such cases, you must assign unique IP addresses for the two interfaces that have duplicate addresses.
- The management Ethernet interface used for the TX Matrix Plus router, T1600 or T4000 routers in a routing matrix, and PTX Series Packet Transport Routers, is em0. Junos OS automatically creates the router's management Ethernet interface, em0.

---

## Commit Operation 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.

If you enter configuration mode with the **configure private** command, each user has a private candidate configuration to edit somewhat independently of other users. When you commit the configuration, only your own changes are committed. To synchronize your copy of the configuration after other users have committed changes, you can run



the **update** command in configuration mode. A commit operation also updates all the private candidate configurations. For example, suppose user X and user Y are both in **configure private** mode, and user X commits a configuration change. When user Y performs a subsequent commit operation and then views the new configuration, the new configuration seen by user Y includes the changes made by user X.

If you enter configuration mode with the **configure exclusive** command, you 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 commit the configuration. This is true even if the other users entered configuration mode before you enter the **configure exclusive** command. For example, suppose user X is already in the **configure private** or **configure** mode. Then suppose user Y enters the **configure exclusive** mode. User X cannot commit any changes to the configuration, even if those changes were entered before user Y logged in. If user Y exits **configure exclusive** mode, user X can then commit the changes made in **configure private** or **configure** mode.

## Commit Preparation and Activation Overview

To save Junos configuration changes to the configuration database and to activate the configuration on the router, the configuration mode command **commit** is used.

Beginning with Junos OS Release 17.3R1, you can complete the commit process in two steps. This feature enables you to configure several devices and simultaneously activate the configurations. Prior to Junos OS Release 17.3R1, the commit process was completed in a single step. The purpose of decoupling these stages of commit is to provide a definitive time window for the commit to be effective on the system. You can enter commit mode after the commit is prepared, but you will receive a message informing that the commit is pending activation.

In the first step, known as the preparation stage, the commit is validated and a new database with the necessary files is generated. If the configuration contains any syntax errors, an appropriate error message is displayed, and the configuration is not prepared. In the event of failure during the preparation stage, the error message **commit check-out failed** is displayed.

In the second step, referred to as the activation stage, the previously prepared configuration is activated. Next, if you need to clear the prepared configuration, you can do so by using **clear system commit prepared** command. A log message is generated upon successful clearing of the pending commit.



**NOTE:** Commit operations cannot be performed in between preparation and activation stages.

The two-step commit process is superior to the single-step process for time-critical commits. In the single-step process, the preparation time can vary depending on the existing configuration on the device. In the two-step process, the complex preparation work is more efficiently handled.

Configuration commands are provided that allow you to prepare the configuration cache and activate the configuration. You can prepare the devices with new configurations and activate them at the exact times you want.

The **commit prepare** command validates the configurations, and the **commit activate** command activates the configurations. The commands have the following configuration options:

- **and-quit**
- **no-synchronize**
- **peers-synchronize**
- **synchronize**

The **commit prepare** and **commit activate** commands are available for private, exclusive and shared commits only. The commands are not applicable for dynamic and ephemeral modes. This feature is applicable for multichassis devices, but it is not applicable for batch commits.

To support this functionality using Network Configuration Protocol (NETCONF), the following new remote procedure calls (RPCs) are provided:

- `<commit-configuration>< prepare/></commit-configuration>`
- `<commit-configuration>< activate/></commit-configuration>`
- `<clear-system-commit>< prepared/></clear-system-commit>`



NOTE:

- In an MX Series Virtual Chassis setup the following applies: When **commit prepare** is issued on one Routing Engine followed by switchover, the Routing Engine where the switchover command is issued reboots. Therefore, the prepared cache is cleared in that Routing Engine.
  - In an MX Series Virtual Chassis setup, it is advisable to execute **clear system commit prepared** command only on VC master.
-

## Committing Junos OS Configurations in Two Steps: Preparation and Activation

To save Junos OS configuration changes to the configuration database and to activate the configuration on the router, the configuration mode command **commit** is used.

Beginning with Junos OS Release 17.3, you can complete the commit process in two steps. This enables you to configure several devices, and the configurations can be activated simultaneously. In the first step, known as the preparation stage, the commit is validated and a new database along with necessary files is generated. If the configuration contains any syntax errors, an appropriate error message is displayed, and the configuration is not prepared. In the second step, referred to as the activation stage, the previously prepared configuration is activated and becomes the current, operational device configuration.

To prepare the configuration:

1. At the **[edit]** hierarchy level in configuration mode, make the necessary changes to the configuration.

For example, to configure the scripts of the system, issue the following command:

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

For example:

```
[edit]
user@host# set system scripts language python
```

2. Issue the **commit prepare** command.

```
[edit]
user@host# commit prepare
```

The message **commit prepare successful** is displayed.

3. To verify the output of the **show system commit** command after **commit prepare** is issued, use the following command:

```
user@host> show system commit
commit prepared by user via cli is pending activation
```

If the preparation stage fails, the error message **commit check-out failed** is displayed.

```
[edit]
user@host# set interfaces ge-0/0/0 unit 0 family inet address 1.1.1.2/2
[edit]
user@host# set interfaces ge-0/0/1 unit 0 family inet address 1.1.1.2/24
[edit]
user@host# commit prepare
[edit interfaces ge-2/0/0 unit 0 family inet]
'address 1.1.1.2/24'
Cannot have the same local address on the same unit of an interface
error: configuration check-out failed
```

To activate the prepared configuration:

1. Use the **commit activate** command

```
[edit]
user@host# commit activate
```

The message **commit complete** is displayed.

2. To verify the activated system configuration, use the following command:

```
user@host> show configuration system scripts
language python;
```

To verify the output of the **show system commit** and **show system commit revision detail** commands after **commit activate** is issued, issue the following commands.

```
user@host> show system commit
0 2018-07-12 22:54:46 PDT by user via cli commit activate
```

```
user@host> show system commit revision detail
Revision: re0-1499925285-2214
User : user
Client : cli
Time : 2018-07-12 22:54:46 PDT
Comment : commit activate
```

## Activating a Junos OS 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 device.

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
configuration check succeeds
```

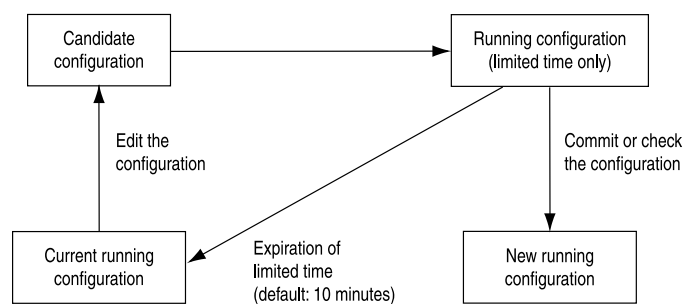
If the commit is not confirmed within a certain time (10 minutes by default), Junos OS automatically rolls back to the previous configuration and a broadcast message is sent to all logged-in users.

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 2018-01-05 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 temporarily (10 minutes by default) and begins running on the device.

**Figure 10: Confirm a Configuration**



1414

To change the amount of time before you must 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#
```

Beginning with Junos OS Release 11.4, you can also use the **commit confirmed** command in the **[edit private]** configuration mode.

## Scheduling a Junos OS Commit Operation

You can schedule when you want your candidate configuration to become active. To save Junos OS 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
```

Where *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 mode 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, **2018-08-21 12:30:00** is 12:30 PM on August 21, 2018. 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 "2018-03-10 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 Junos OS fails before the configuration changes become active, all configuration changes are lost.

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

You cannot enter 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 [CLI Explorer](#).



**NOTE:** We do not recommend performing a commit operation on the backup Routing Engine when graceful Routing Engine switchover is enabled on the device.

## Monitoring the Junos OS 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
2018-09-22 15:39:39 PDT: exporting juniper.conf
2018-09-22 15:39:39 PDT: setup foreign files
2018-09-22 15:39:39 PDT: propagating foreign files
2018-09-22 15:39:39 PDT: complete foreign files
2018-09-22 15:39:40 PDT: copying configuration to juniper.data+
2018-09-22 15:39:40 PDT: dropping unchanged foreign files
2018-09-22 15:39:40 PDT: daemons checking new configuration
2018-09-22 15:39:41 PDT: commit wrapup...
2018-09-22 15:39:42 PDT: activating '/var/etc/ntp.conf'
2018-09-22 15:39:42 PDT: activating '/var/etc/kmd.conf'
2018-09-22 15:39:42 PDT: activating '/var/db/juniper.data'
2018-09-22 15:39:42 PDT: notifying daemons of new configuration
2018-09-22 15:39:42 PDT: signaling 'Firewall daemon', pid 24567, signal 1,
status 0
2018-09-22 15:39:42 PDT: signaling 'Interface daemon', pid 24568, signal 1,
status 0
2018-09-22 15:39:43 PDT: signaling 'Routing protocol daemon', pid 25679,
signal 1, status 0
2018-09-22 15:39:43 PDT: signaling 'MIB2 daemon', pid 24549, signal 1,
status 0
2018-09-22 15:39:43 PDT: signaling 'NTP daemon', pid 37863, signal 1, status 0
```

```
2018-09-22 15:39:43 PDT: signaling 'Sonet APS daemon', pid 24551, signal 1,
status 0
2018-09-22 15:39:43 PDT: signaling 'VRRP daemon', pid 24552, signal 1,
status 0
2018-09-22 15:39:43 PDT: signaling 'PFE daemon', pid 2316, signal 1, status 0
2018-09-22 15:39:43 PDT: signaling 'Traffic sampling control daemon', pid 24553
signal 1, status 0
2018-09-22 15:39:43 PDT: signaling 'IPsec Key Management daemon', pid
24556, signal 1, status 0
2018-09-22 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.

Beginning with Junos OS Release 11.4, you can also use the **commit confirmed** command in the **[edit private]** configuration mode.



## Junos OS Batch Commits Overview

Junos OS provides a batch commit feature that aggregates or merges multiple configuration edits from different CLI sessions or users and adds them to a batch commit queue. A batch commit server running on the device takes one or more jobs from the batch commit queue, applies the configuration changes to the shared configuration database, and then commits the configuration changes in a single commit operation.

Batches are prioritized by the commit server based on priority of the batch specified by the user or the time when the batch job is added. When one batch commit is complete, the next set of configuration changes are aggregated and loaded into the batch queue for the next session of the batch commit operation. Batches are created until there are no commit entries left in the queue directory.

When compared to the regular commit operation where all commits are independently committed sequentially, batch commits save time and system resources by committing multiple small configuration edits in a single commit operation.

Batch commits are performed from the **[edit batch]** configuration mode. The commit server properties can be configured at the **[edit system commit server]** hierarchy level.

### Aggregation and Error Handling

When there is a load-time error in one of the aggregated jobs, the commit job that encounters the error is discarded and the remaining jobs are aggregated and committed.

For example, if there are five commit jobs (**commit-1**, **commit-2**, **commit-3**, **commit-4**, and **commit-5**) being aggregated, and **commit-3** encounters an error while loading, **commit-3** is discarded and **commit-1**, **commit-2**, **commit-4**, and **commit-5** are aggregated and committed.

If there is an error during the commit operation when two or more jobs are aggregated and committed, the aggregation is discarded and each of those jobs is committed individually like a regular commit operation.

For example, if there are five commit jobs (**commit-1**, **commit-2**, **commit-3**, **commit-4**, and **commit-5**) that are aggregated and if there is a commit error caused because of **commit-3**, the aggregation is discarded, **commit-1**, **commit-2**, **commit-3**, **commit-4**, and **commit-5** are committed individually, and the CLI reports a commit error for **commit-3**.

## Example: Configuring Batch Commit Server Properties

This example shows how to configure batch commit server properties to manage batch commit operations.

- [Requirements on page 184](#)
- [Overview on page 184](#)
- [Configuration on page 184](#)
- [Verification on page 186](#)

## Requirements

---

This example uses the following hardware and software components:

- MX Series 5G Universal Routing Platform
- Junos OS Release 12.1 or later running on the device

## Overview

---

You can control how the batch commit queue is handled by the commit server by configuring the server properties at the **[edit system commit server]** hierarchy level. This enables you to control how many commit jobs are aggregated or merged into a single batch commit, the maximum number of jobs that can be added to the queue, days to keep batch commit error logs, interval between two batch commits, and tracing operations for batch commit operations.

## Configuration

---

### CLI Quick Configuration

To quickly configure this section of the example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level. You can configure the commit server properties from either the regular **[edit]** mode or the **[edit batch]** mode.

### Device R0

```
set system commit server maximum-aggregate-pool 4
set system commit server maximum-entries 500
set system commit server commit-interval 5
set system commit server days-to-keep-error-logs 30
set system commit server traceoptions file commitd_nov
set system commit server traceoptions flag all
```

### Configuring the Commit Server Properties

### Step-by-Step Procedure

1. (Optional) Configure the number of commit transactions to aggregate or merge in a single commit operation.

The default value for **maximum-aggregate-pool** is 5.



**NOTE:** Setting **maximum-aggregate-pool** to 1 commits each of the jobs individually.

---

In this example, the number of commit transactions is set to 4 indicating that four different commit jobs are aggregated into a single commit before the commit operation is initiated.

```
[edit system commit server]
user@R0# set maximum-aggregate-pool 4
```

2. (Optional) Configure the maximum number of jobs allowed in a batch.

This limits the number of commits jobs that are added to the queue.

```
[edit system commit server]
user@R0# set maximum-entries 500
```



**NOTE:** If you set `maximum-entries` to 1, the commit server cannot add more than one job to the queue, and the CLI displays an appropriate message when you try to commit more than one job.

3. (Optional) Configure the time (in seconds) to wait before starting the next batch commit operation.

```
[edit system commit server]
user@R0# set commit-interval 5
```

4. (Optional) Configure the number of days to keep error logs.

The default value is **30** days.

```
[edit system commit server]
user@R0# set days-to-keep-error-logs 30
```

5. (Optional) Configure tracing operations to log batch commit events.

In this example, the filename for logging batch commit events is **commitd\_nov**, and all traceoption flags are set.

```
[edit system commit server]
user@R0# set traceoptions commitd_nov
user@R0# set traceoptions flag all
```

**Results** From configuration mode, confirm your configuration by entering the **show system commit server** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R0# show system commit server
maximum-aggregate-pool 4;
maximum-entries 500;
commit-interval 5;
days-to-keep-error-logs 30;
traceoptions {
  file commitd_nov;
  flag all;
}
```

### *Committing the Configuration from Batch Configuration Mode*

#### **Step-by-Step Procedure**

To commit the configuration from the **[edit batch]** mode, do one of the following:

- Log in to the device and enter **commit**.

```
[edit batch]
user@R0# commit
Added to commit queue request-id: 1000
```

- To assign a higher priority to a batch commit job, issue the **commit** command with the **priority** option.

```
[edit batch]
user@R0# commit priority
Added to commit queue request-id: 1001
```

- To commit a configuration without aggregating the configuration changes with other commit jobs in the queue, issue the **commit** command with the **atomic** option.

```
[edit batch]
user@R0# commit atomic
Added to commit queue request-id: 1002
```

- To commit a configuration without aggregating the configuration changes with other commit jobs in the queue, and issuing a higher priority to the commit job, issue the **commit** command with the **atomic priority** option.

```
[edit batch]
user@R0# commit atomic priority
Added to commit queue request-id: 1003
```

---

### **Verification**

Confirm that the configuration is working properly.

- [Checking the Batch Commit Server Status on page 186](#)
- [Checking the Batch Commit Status on page 187](#)
- [Viewing the Patch Files in a Batch Commit Job on page 188](#)
- [Viewing the Trace Files for Batch Commit Operations on page 190](#)

### *Checking the Batch Commit Server Status*

**Purpose** Check the status of the batch commit server.

**Action** user@R0> show system commit server  
Commit server status : Not running

By default, the status of the commit server is **Not running**. The commit server starts running only when a batch commit job is added to the queue.

When a batch commit job is added to the queue, the status of the commit server changes to **Running**.

user@R0> show system commit server  
Commit server status : Running  
Jobs in process:  
1003 1004 1005

**Meaning** The **Jobs in process** field lists the commit IDs of jobs that are in process.

#### *Checking the Batch Commit Status*

**Purpose** Check the commit server queue for the status of the batch commits.

**Action** user@R0> show system commit server queue

```
Pending commits:
  Id: 1005
  Last Modified: Tue Nov  1 23:56:43 2018

Completed commits:
  Id: 1000
  Last Modified: Tue Nov  1 22:46:43 2018
  Status: Successfully committed 1000

  Id: 1002
  Last Modified: Tue Nov  1 22:50:35 2018
  Status: Successfully committed 1002

  Id: 1004
  Last Modified: Tue Nov  1 22:51:48 2018
  Status: Successfully committed 1004

  Id: 1007
  Last Modified: Wed Nov  2 01:08:04 2018
  Status: Successfully committed 1007

  Id: 1009
  Last Modified: Wed Nov  2 01:16:45 2018
  Status: Successfully committed 1009

  Id: 1010
  Last Modified: Wed Nov  2 01:19:25 2018
  Status: Successfully committed 1010

  Id: 1011
  Last Modified: Wed Nov  2 01:28:16 2018
  Status: Successfully committed 1011

Error commits:
  Id: 1008
  Last Modified: Wed Nov  2 01:08:18 2018
  Status: Error while committing 1008
```

**Meaning** **Pending commits** displays commit jobs that are added to the commit queue but are not committed yet. **Completed commits** displays the list of commit jobs that are successful. **Error commits** are commits that failed because of an error.

#### *Viewing the Patch Files in a Batch Commit Job*

**Purpose** View the timestamps, patch files, and the status of each of the commit jobs. Patch files show the configuration changes that occur in each commit operation that is added to the batch commit queue.

**Action** 1. Use the **show system commit server queue patch** command to view the patches for all commit operations.

```
user@R0> show system commit server queue patch
```

```
Pending commits:
  none
```

```
Completed commits:
```

```
  Id: 1000
  Last Modified: Tue Nov  1 22:46:43 2018
  Status: Successfully committed 1000
```

```
Patch:
```

```
[edit groups]
  re1 { ... }
+ GRP-DHCP-POOL-NOACCESS {
+   access {
+     address-assignment {
+       pool <*> {
+         family inet {
+           dhcp-attributes {
+             maximum-lease-time 300;
+             grace-period 300;
+             domain-name verizon.net;
+             name-server {
+               4.4.4.1;
+               4.4.4.2;
+             }
+           }
+         }
+       }
+     }
+   }
+ }
```

```
  Id: 1002
  Last Modified: Tue Nov  1 22:50:35 2018
  Status: Successfully committed 1002
```

```
Patch:
```

```
[edit]
+ snmp {
+   community abc;
+ }
```

```
  Id: 1010
  Last Modified: Wed Nov  2 01:19:25 2018
  Status: Successfully committed 1010
```

```
Patch:
```

```
[edit system syslog]
  file test { ... }
+ file j {
+   any any;
+ }
```

```
Error commits:
```

```
  Id: 1008
  Last Modified: Wed Nov  2 01:08:18 2018
  Status: Error while committing 1008
```

```
Patch:
```

```
[edit system]
+ radius-server {
```

```
+    10.1.1.1 port 222;  
+ }
```

The output shows the changes in configuration for each commit job ID.

2. To view the patch for a specific commit job ID, issue the **show system commit server queue patch id <id-number>** command.

```
user@R0> show system commit server queue patch id 1000
```

```
Completed commits:
```

```
Id: 1000  
Last Modified: Tue Nov  1 22:46:43 2018  
Status: Successfully committed 1000
```

```
Patch:
```

```
[edit system]  
+ radius-server {  
+   192.168.69.162 secret teH.bTc/RVbPM;  
+   192.168.64.10 secret teH.bTc/RVbPM;  
+   192.168.60.52 secret teH.bTc/RVbPM;  
+   192.168.60.55 secret teH.bTc/RVbPM;  
+   192.168.4.240 secret teH.bTc/RVbPM;  
+ }
```

**Meaning** The output shows the patch created for a commit job. The + or - sign indicates the changes in the configuration for a specific commit job.

#### *Viewing the Trace Files for Batch Commit Operations*

**Purpose** View the trace files for batch commit operations. You can use the trace files for troubleshooting purposes.



- **Action** Use the `file show /var/log/<filename>` command to view all entries in the log file.

```
user@R0> file show /var/log/commitd_nov
```

The output shows commit server event logs and other logs for batch commits.

```
Nov  1 22:46:43 Successfully committed 1000
Nov  1 22:46:43 pausing after commit for 0 seconds
...
Nov  1 22:46:43 Done working on queue
...

Nov  1 22:47:17 maximum-aggregate-pool = 5
Nov  1 22:47:17 maximum-entries= 0
Nov  1 22:47:17 asynchronous-prompt = no
Nov  1 22:47:17 commit-interval = 0
Nov  1 22:47:17 days-to-keep-error-logs = -1
...
Nov  1 22:47:17 Added to commit queue request-id: 1001
Nov  1 22:47:17 Commit server status=running
Nov  1 22:47:17 No need to pause
...

Nov  1 22:47:18 Error while committing 1001
Nov  1 22:47:18 doing rollback
...
```

- To view log entries only for successful batch commit operations, issue the `file show /var/log/<filename>` command with the `| match committed` pipe option.

The output shows batch commit job IDs for successful commit operations.

```
user@R0> file show /var/log/commitd_nov | match committed
```

```
Nov  1 22:46:43 Successfully committed 1000
Nov  1 22:50:35 Successfully committed 1002
Nov  1 22:51:48 Successfully committed 1004
Nov  2 01:08:04 Successfully committed 1007
Nov  2 01:16:45 Successfully committed 1009
Nov  2 01:19:25 Successfully committed 1010
Nov  2 01:28:16 Successfully committed 1011
```

- To view log entries only for failed batch commit operations, issue the `file show /var/log/<filename>` command with the `| match "Error while"` pipe option.

The output shows commit job IDs for failed commit operations.

```
user@R0> file show /var/log/commitd_nov | match "Error while"
```

```
Nov  1 22:47:18 Error while committing 1001
Nov  1 22:51:10 Error while committing 1003
Nov  1 22:52:15 Error while committing 1005
...
```

- To view log entries only for commit server events, issue the `file show /var/log/<filename>` command with the `| match "commit server"` pipe option.

The output shows commit server event logs.

```
user@R0> file show/var/log/commitd_nov | match "commit server"
```

```
Nov  1 22:46:39 Commit server status=running
Nov  1 22:46:39 Commit server jobs=1000
Nov  1 22:46:43 Commit server status=not running
Nov  1 22:46:43 Commit server jobs=
Nov  1 22:47:17 Commit server status=running
Nov  1 22:47:18 Commit server jobs=1001
Nov  1 22:47:18 2 errors reported by commit server
Nov  1 22:47:18 Commit server status=not running
Nov  1 22:47:18 Commit server jobs=
Nov  1 22:50:31 Commit server status=running
Nov  1 22:50:31 Commit server jobs=1002
Nov  1 22:50:35 Commit server status=not running
Nov  1 22:50:35 Commit server jobs=
Nov  1 22:51:09 Commit server status=running
Nov  1 22:51:10 Commit server jobs=1003
Nov  1 22:51:10 2 errors reported by commit server
Nov  1 22:51:10 Commit server status=not running
...
```

## Backing Up the Committed Configuration on 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:** For more information about backing up the file system on an ACX Series Universal Metro Router, see *Understanding System Snapshot on an ACX Series Router*.

.....

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.

**Release History Table**

Release	Description
17.3R1	Beginning with Junos OS Release 17.3R1, you can complete the commit process in two steps. This feature enables you to configure several devices and simultaneously activate the configurations

**Related  
Documentation**

- *commit*
- [Configure Command Overview on page 137](#)

