Engineering
Simplicity

Junos® OS

Overview for Junos OS

Published
2025-06-23

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

YEAR 2000 NOTICE

END USER LICENSE AGREEMENT

Table of Contents

About This Guide

Use this guide to get familiar with the various functions of Junos OS devices, and learn how to configure, monitor, and manage them.

1

PART

Understanding Junos OS

CHAPTER 1

Junos OS Software Overview

IN THIS CHAPTER

About the Overview for Junos OS

The Overview for Junos OS is intended to provide a technical and detailed exploration of Junos OS, explaining both concepts and operational principles, as well as how to configure and use Juniper Networks devices.

In this guide, we cover:

- Understanding Junos OS
- Security management
- Device configuration
- Device monitoring
- Managing network devices

- Using configuration statements and operational commands

For a basic introduction to Junos OS, see the Getting Started Guide for Junos OS. It provides a high-level description of Junos OS, describes how to access devices, and provides simple step-by-step instructions for initial device configuration.

For introductory and overview information specific to Junos OS Evolved, see Introducing Junos OS Evolved. This guide will acquaint you with Junos OS Evolved, the next generation Junos OS, and explain its strengths, similarities to, and differences from Junos OS.

To learn how to use the Junos OS command-line interface (CLI) and understand more advanced Junos OS topics, see the CLI User Guide. This guide explains how to use the CLI, enter configuration statements, manage configurations, and enter operational commands for monitoring Junos OS networking devices.

RELATED DOCUMENTATION

CLI User Guide

Getting Started Guide for Junos OS

Introducing Junos OS Evolved

Junos OS Overview

Juniper Networks provides high-performance network devices that create a responsive and trusted environment for accelerating the deployment of services and applications over a single network. The Junos operating system (Junos OS) is the foundation of these high-performance networks. Unlike other complex, monolithic software architectures, Junos OS incorporates key design and developmental differences to deliver increased network availability, operational efficiency, and flexibility. These key advantages are:

- Consistent operating system
- Concurrent software releases
- Modular software architecture

Consistent Operating System

Unlike other network operating systems that share a common name but splinter into many different programs, Juniper takes a cohesive approach. Junos OS and Junos OS Evolved are supported across Juniper routers, switches, and firewalls. They are closely aligned with similar code and releases are made at the same cadence for seamless management continuity. Feature consistency across product lines and operating systems not only reduces the training required for network operators to learn different tools and methods but also enables simultaneous updates across product lines.

Concurrent Software Releases

Each new mainline version of Junos OS is released concurrently for all product lines. Each new Junos OS release includes working features released in previous versions of the software and must achieve zero critical regression errors. Any deprecated features or functions are not only announced, but any needed workarounds or solutions are provided. This discipline ensures reliable operations for the entire release.

Modular Software Architecture

Although individual architecture modules of Junos OS communicate through well-defined interfaces, each module runs in its own protected memory space, preventing one module from disrupting another. It also enables the independent restart of each module as necessary. This is in contrast to monolithic operating systems for which a malfunction in one module can ripple to other modules, possibly causing a full system crash or restart. This modular Junos OS architecture provides a high level of performance, high availability, security, and device scalability not found in other operating systems.

Generally, Junos OS is preinstalled on your Juniper Networks device when you receive it from the factory. When you first power on the device, all software starts automatically. You then configure the software so that the device can participate in your network. However, if needed, you can order Juniper Networks devices without any software installed, for additional flexibility.

You can upgrade the device software as new features are added or software problems are fixed. You obtain new software by downloading images from the Juniper Networks Support website onto your device or another system on your local network, then install the software upgrade on the device.

Juniper Networks devices run only binaries supplied by Juniper Networks. Each Junos OS image includes a digitally signed manifest of executables, which are registered with the system only if the signature can be validated. Junos OS will not execute any binary without a registered fingerprint. This feature protects the system against unauthorized software and activity that might compromise the integrity of your network devices.

RELATED DOCUMENTATION

Junos OS Architecture Overview

IN THIS SECTION

This topic provides an overview of the Junos OS routing process architecture:

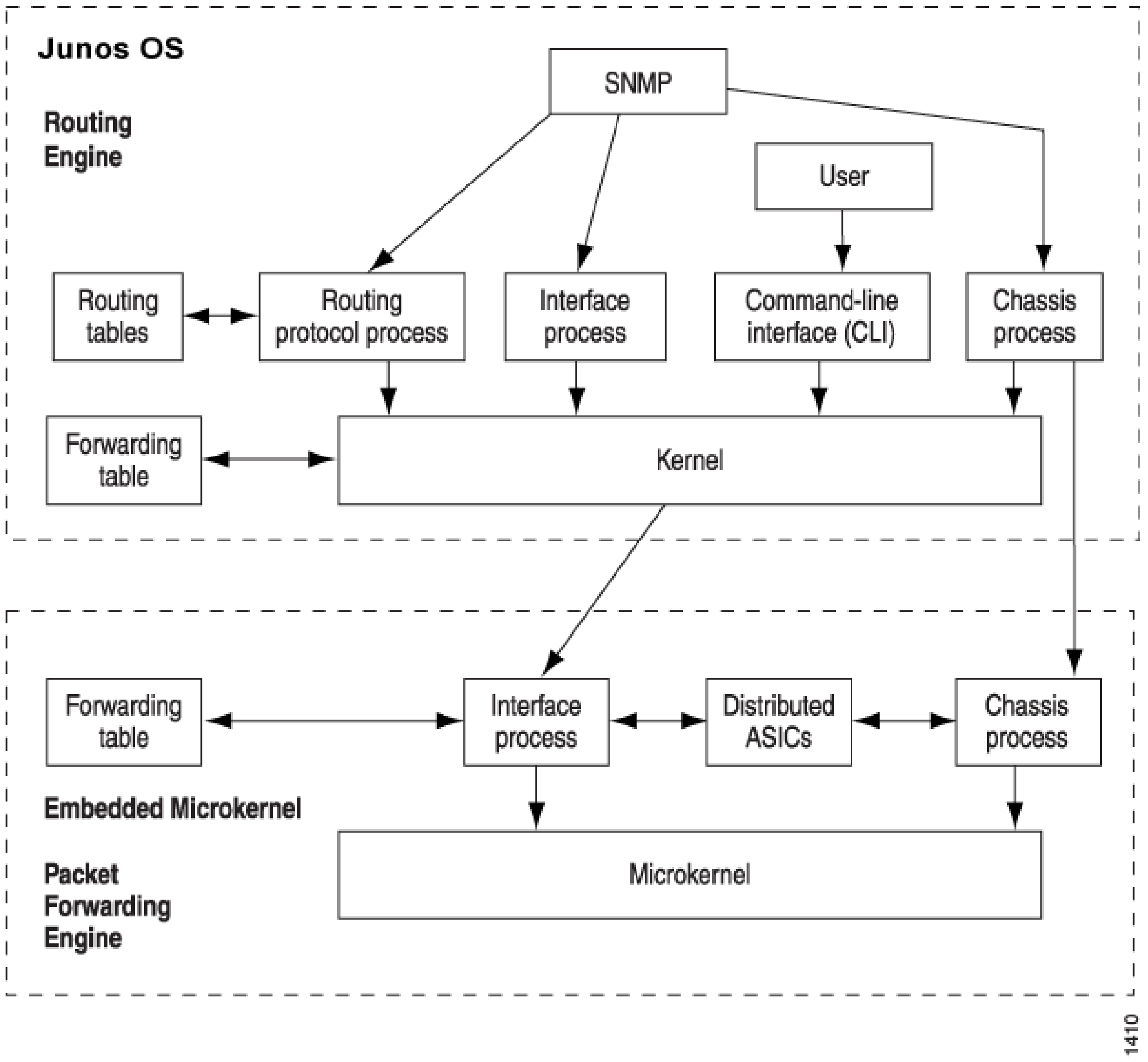
Routing Process Architecture

The routing process is handled by the following two components (see Figure 1 on page 6):

- *Routing Engine*
- *Packet Forwarding Engine*

Because this architecture separates control operations such as routing updates and system management from packet forwarding, the router can deliver superior performance and highly reliable Internet operation.

Figure 1: Product Architecture



Packet Forwarding Engine

The Packet Forwarding Engine uses application-specific integrated circuits (*ASIC*s) to perform Layer 2 and Layer 3 packet switching, route lookups, and packet forwarding. The Packet Forwarding Engine forwards packets between input and output interfaces.

Routing Engine

The Routing Engine controls the routing updates and the system management. The Routing Engine consists of routing protocol software processes running inside a protected memory environment on a general-purpose computer platform. The Routing Engine handles all of the routing protocol processes and other software processes that control the routers' interfaces, some of the chassis components, system management, and user access to the router. These routers and software processes run on top of a kernel that interacts with the Packet Forwarding Engine.

The Routing Engine has these features:

- Routing protocol packets processing—All routing protocol packets from the network are directed to the Routing Engine, and therefore do not unnecessarily delay the Packet Forwarding Engine.
- Software modularity—Software functions are in separate processes, so a failure of one process has little or no effect on other software processes.
- In-depth IP functionality—Each routing protocol is implemented with a complete set of IP features and provides full flexibility for advertising, filtering, and modifying routes. Routing policies are set according to route parameters, such as *prefix*, prefix lengths, and Border Gateway Protocol (*BGP*) attributes.
- Scalability—Junos OS routing tables are designed to hold all the routes used in current and near-future networks. Additionally, Junos OS can efficiently support large numbers of *interfaces* and *virtual circuit*s.
- Storage and change management—Configuration files, system images, and microcode are held and maintained in one primary and two secondary storage systems, permitting local or remote upgrades.
- Monitoring efficiency and flexibility—Alarms are generated and packets are counted without adversely affecting packet forwarding performance.

The Routing Engine constructs and maintains one or more routing tables. From the routing tables, the Routing Engine derives a table of active routes, called the *forwarding table*, which is then copied into the Packet Forwarding Engine. The forwarding table in the Packet Forwarding Engine can be updated without interrupting the router's forwarding.

RELATED DOCUMENTATION

Junos OS Overview | 3

Router Hardware Components

Junos OS runs on all Juniper Networks devices, including both routers and switches. This section focuses specifically on router hardware components.

Table 1 on page 8 lists the major hardware components in each router series.

NOTE: The ACX Series router is a single-board router with a built-in Routing Engine and one Packet Forwarding Engine. The “pseudo” FPCs and PICs are described in *ACX2000 and ACX2100 Routers Hardware and CLI Terminology Mapping*.

Table 1: Major Router Hardware Components

	M Series	MX Series	T Series	PTX Series	J Series
Routing Engines	X	X	X	X	X
Control Board	X		X	X	
Switch Interface Board (SIB)	X		X	X	
Forwarding Engine Board (FEB)	X				
Power Supply	X	X	X	X	X
Cooling System	X	X	X	X	X
Dense Port Concentrators (DPC)		X			
Switch Control Board (SCB)		X			
Flexible PIC Concentrators (FPC)	X	X	X	X	
Physical Interface Module (PIM)					X
Physical Interface Card (PIC)	X	X	X	X	

Flexible PIC Concentrators (*FPCs*) are each populated by *PICs* for various interface types. On some routers, the PICs are installed directly in the chassis.

For information about specific components in your router, refer to its hardware guide.

RELATED DOCUMENTATION

Junos OS Architecture Overview | 5

Junos OS Routing Engine Components and Processes

IN THIS SECTION

Junos OS also runs on the *Routing Engine*. Junos OS consists of software processes that support Internet routing *protocols*, control router interfaces and the router chassis, enable router system management, and much more. Junos OS processes run on top of a *kernel*, which enables communication between processes and provides a direct link to the Packet Forwarding Engine software. Junos OS can be used to configure routing protocols and router interface properties, as well as to monitor and troubleshoot protocol and network connectivity problems.

The Routing Engine software consists of several software processes that control router functionality and a kernel that provides the communication among the processes. Following is a listing of the major Routing Engine-related processes.

Routing Engine Kernel

The Routing Engine kernel provides the underlying infrastructure for all Junos OS processes, including providing the link between the routing tables and the Routing Engine's forwarding table. The kernel is also responsible for all communication with the *Packet Forwarding Engine*, which includes keeping the Packet Forwarding Engine's copy of the forwarding table synchronized with the master copy in the Routing Engine.

Initialization Process

When the device boots, an initialization process (init) starts and monitors all the other software processes.

If a software process terminates or fails to start when called, the init process attempts to restart it a limited number of times and logs any failure information for further investigation.

Management Process

The management process (mgd) manages the configuration of the router and all user commands. The management process is responsible for notifying other processes when a new configuration is committed. A dedicated management process handles Junos XML protocol XML requests from its client, which might be the CLI or any Junos XML protocol client.

Process Limits

There are limits to the total number of Junos OS processes that can run simultaneously on a device. There are also limits set for the maximum number of iterations of any single process. The limit for iterations of any single process can only be reached if the limit of overall system processes is not exceeded.

Access methods such as *telnet* and *SSH* spawn multiple system processes for each session created. For this reason, it might not be possible to simultaneously support the maximum number of access sessions for multiple services.

Routing Protocol Process

Within Junos OS, the routing protocol process (rpd) controls the routing protocols that run on the device. The rpd process starts all configured routing protocols and handles all routing messages. It maintains one or more routing tables, which consolidate the routing information learned from all routing protocols. From this routing information, the routing protocol process determines the active routes to network destinations and installs these routes into the Routing Engine’s forwarding table. Finally, rpd implements routing policy, which enables you to control the routing information that is transferred between the routing protocols and the routing table. Using routing policy, you can filter and limit the transfer of information as well as set properties associated with specific routes.

Interface Process

The Junos OS interface process enables you to configure and control the physical interface devices and logical interfaces present in a network device. You can configure interface properties such as the interface location, for example, in which slot the Flexible PIC Concentrator (FPC) is installed and in which location on the FPC the *Physical Interface Card* (PIC) is installed, as well as the interface encapsulation and interface-specific properties. You can configure the interfaces currently present in the device, as well as interfaces that are not present but that you might add later.

The Junos OS interface process communicates through the Junos OS kernel with the interface process in the Packet Forwarding Engine, enabling Junos OS to track the status and condition of the network device's interfaces.

Chassis Process

The Junos OS chassis process (chassisd) enables you to configure and control the properties of the device, including conditions that trigger alarms. The chassisd on the Routing Engine communicates directly with its peer processes running on the Packet Forwarding Engine.

SNMP and MIB II Processes

Junos OS supports the Simple Network Management Protocol (*SNMP*), which helps administrators monitor the state of a device. The software supports SNMP version 1 (SNMPv1), version 2 (SNMPv2, also known as version 2c, or v2c), and version 3 (SNMPv3). The Junos OS implementation of SNMP does not include any of the security features that were originally included in the *IETF* SNMP drafts but were later dropped. The SNMP software is controlled by the Junos OS SNMP and Management Information Base II (MIB II) processes, which consist of an SNMP master agent and various subagents.

RELATED DOCUMENTATION

Junos OS Architecture Overview | 5

Junos OS Routing Processes

Junos OS consists of multiple processes that run on different platforms and have unique functions. The separation of functions provides operational stability, because each process accesses its own protected memory space. This section provides a brief overview of Junos OS routing-specific processes.

As an example, Table 2 on page 12 describes the processes that run on MX Series 5G Universal Routing Platforms.

Table 2: Junos OS Processes on MX Series Platform

Process	Name	Description
Clksync process (RE)	clksyncd	Defines the operation of synchronous Ethernet and Precision Time Protocol (*PTP*) on a Juniper Networks MX Series router. The operation includes communication with the Packet Forwarding Engine (clock-sync module) to program and process clock events from the *EEC* clock. Operates the PTP stack, exchanges packets, and handles the configuration changes for the modular MX Series (MX80). Controls the configuration and monitoring of the overall operation of the PTP functionality for chassis-based MX Series platforms (MX240, MX480, and so on).
Clock-sync process (PFE)	clock-sync	Programs and monitors the modular interface card (MIC), the CPLD, and the EEC clock. Peer of the clksyncd process module. Captures all PTP and Synchronous Ethernet statistics on the Packet Forwarding Engine and provides them to the Routing Engine.
Interchassis communication process	iccpd	Exchanges proprietary Junos OS messages between two Juniper Networks MX Series routers that take part in a multichassis link aggregation group (*LAG*).
Statistics agent process	stats-agentd	Acts as a relay process to collect interface statistics for all software development kit (*SDK*) applications. Interacts with the pfed process to collect the logical interface statistics for SDK applications.

Table 3 on page 13 lists other processes that are common across Junos OS routing platforms.

Table 3: Junos OS Routing-Specific Processes

Process	Name	Description
Adaptive services process	adaptive-services	Manages the configuration for *stateful firewall*, Network Address Translation (*NAT*), intrusion detection service (*IDS*), and IP Security (*IPsec*) services on the Adaptive Services *PIC*.
Alarm control process	alarm-control	Configures the system alarm.
Access Node Control Protocol (*ANCP*) process	ancpd-service	Works with a special Internet Group Management Protocol (*IGMP*) session to collect outgoing interface mapping events in a scalable manner.
Application identification process	application-identification	Identifies an application using intrusion detection and prevention (*IDP*) to allow or deny traffic based on applications running on standard or nonstandard ports.
RADIUS accounting process	audit-process	Gathers statistical data that can be used for general network monitoring, analyzing, and tracking usage patterns, for billing a user based upon the amount of time or type of services accessed.
Auto-configuration process	auto-configuration	Configures interfaces automatically.
Boot process	bootp	Enables a router, switch, or interface to act as a Dynamic Host Configuration Protocol (*DHCP*) or bootstrap protocol (*BOOTP*) relay agent. DHCP relaying is disabled.
Captive portal content delivery process	captive-portal-content-delivery	Specifies the location to which a subscriber's initial Internet browser session is redirected, enabling initial provisioning and service selection for the subscriber.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Universal Edge Layer 2 Tunneling Protocol process	ce-l2tp-service	(M10, M10i, M7i, and MX Series routers only) Establishes L2TP tunnels and Point-to-Point Protocol (*PPP*) sessions through L2TP tunnels.
Ethernet *OAM* connectivity fault management process	cfm	Monitors the physical link between two switches.
Chassis control process	chassis-control	Manages the chassis.
Class of service process	class-of-service	Controls the network device's *CoS* configuration.
Ethernet clock synchronization process	clksyncd-service	Uses Synchronous Ethernet (*SyncE*) for external clock synchronization .
Craft interface *I/O* control process	craft-control	Controls the I/O of the craft interface.
Database replication process	database-replication	(EX Series switches and MX Series routers only) Manages the replication of updates from the primary to the client in the database management system.
Datapath trace process	datapath-trace-service	Traces the path taken by the packet through the network.
Dynamic Host Configuration Protocol process	dhcp-service	(EX Series switches and MX Series routers only) Enables a DHCP server to allocate network IP addresses and deliver configuration settings to client hosts without user intervention.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Diameter process	diameter-service	Implements the Diameter protocol which uses the Transmission Control Protocol (*TCP*) and Stream Control Transmission Protocol (*SCTP*) instead of User Datagram Protocol (*UDP*), for monitoring the network.
Disk monitoring process	disk-monitoring	Checks the health of the hard drive on the Routing Engine.
Dynamic flow capture (*DFC*) process	dynamic-flow-capture	Controls the DFC configurations on Monitoring Services III PICs.
ECC parity errors logging process	ecc-error-logging	Logs the ECC parity errors into the memory on the Routing Engine.
Connectivity fault management (*CFM*) process	ethernet-connectivity-fault-management	Provides IEEE 802.1ag OAM CFM database information for CFM maintenance association end points *(MEPs*) in a CFM session.
Ethernet *OAM* Link-Fault-Management process	ethernet-link-fault-management	(EX Series switches and MX Series routers only) Provides the OAM link fault management (*LFM*) information for Ethernet interfaces.
Event processing process	event-processing or eventd	Configures the application to handle all generated events.
Firewall process	firewall	Manages the firewall configuration and enables accepting or rejecting packets that are transiting an interface on a device.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
General authentication process	general-authentication-service	(EX Series switches and MX Series routers only) Manages general authentication of a user.
Inter-Chassis Communication Protocol (*ICCP*) process	iccp-service	Synchronizes data within a set of two (or more) *PEs* that form a redundancy group (*RG*).
IDP policy process	idp-policy	Enables various attack detection and prevention techniques on traffic traversing the network.
Integrated Local Management Interface process	ilmi	Provides bidirectional exchange of management information between two Asynchronous Transfer Mode (*ATM*) interfaces across a physical connection.
Inet process	inet-process	Configures the IP multicast family.
Init process	init	Initializes the *USB* modem.
Interface control process	interface-control	Controls the router's or switch’s physical interface devices and logical interfaces.
Kernel replication process	kernel-replication	Replicates the state of the backup Routing Engine when graceful Routing Engine switchover (*GRES*) is configured.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Layer 2 address flooding and learning process	l2-learning	Enables a network device to: • Learn unicast media access control (*MAC*) addresses to avoid flooding the packets to all the ports in a bridge domain. • Create a source MAC entry in its source and destination MAC tables for each MAC address learned from packets received on ports that belong to the bridge domain.
Layer 2 Control Protocol process	l2cpd-service	Enables features such as Layer 2 protocol tunneling and nonstop bridging.
Link Aggregation Control Protocol process	lacp	The process: • Provides a standardized means for exchanging information between partner systems on a link. • Allows the link aggregation control instances to reach agreement on the identity of the Link Aggregation Group (*LAG*) to which the link belongs, and then to move the link to that LAG. • Enables the transmission and reception processes for the link to function in an orderly manner.
Link management process	link-management	Manages traffic engineering links.
Local policy decision function process	local-policy-decision-function	Regulates the collection of statistics related to applications and application groups and tracking of information about dynamic subscribers and static interfaces.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Logical system *multiplexer* process	logical-system-mux or lrmuxd	Manages multiple instances of the routing protocols process (rpd) on a machine running logical routers.
MAC validation process	mac-validation	Configures MAC address validation that enables a network device to validate if received packets contain a trusted IP source and an Ethernet MAC source address.
Management Information Base II process	mib-process	Provides the device's *MIB* II agent.
Mobile IP process	mobile-ip	Configures Junos OS Mobile IP features.
NFS mount requests process	mountd-service	(Some EX Series switches and MX Series routers only) Completes internal NFS mount requests for MS-PIC and MS-MPC.
MPLS Periodic Traceroute process	mpls-traceroute	Enables tracing of forwarding equivalence classes (*FECs*) for *LDP* Layered Service Providers (*LSPs*).
Multiservice process	mspd	Configures multiservice edge routers.
Multicast Snooping process	multicast-snooping	(EX Series switches and MX Series routers only) Makes Layer 3 information, such as the MAC addresses of members of a multicast group, known to Layer 2 devices, such as *VLAN* switches.
DNS server process	named-service	Enables a device to resolve hostnames into addresses.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Bidirectional Forwarding Detection (BFD) process	neighbor-liveness	Displays the process that specifies the maximum length of time that the device waits for its neighbor to re-establish an LDP session.
Remote *NFS* server process	nfsd-service	Provides remote file access for applications that need NFS-based transport.
Network time process	ntp	Provides the mechanisms to synchronize time and coordinate time distribution in a large, diverse network.
Packet-triggered dynamic subscribers and policy control (PTCP) process	packet-triggered-subscribers	Enables the application of policies to dynamic subscribers that are controlled by a subscriber termination device.
Peer selection service process	peer-selection-service	Enables peer selection.
Periodic packet management process	periodic-packet-services	Processes a variety of time-sensitive periodic tasks so that other processes can more optimally direct their resources.
Packet Forwarding Engine process	pfed	Gathers and reports Packet Forwarding Engine statistics.
Packet gateway service process	pgcp-service or pgcpd	Configures the Packet Gateway Control Protocol (*PGCP*) that is required for the border gateway function (*BGF*) feature.
Pragmatic General Multicast process	pgm	Enables a reliable transport layer for multicast applications.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
PIC services logging process	pic-services-logging or fsad (the file system access daemon)	Enables PICs to send special logging information to the Routing Engine for archiving on the hard drive.
Point-to-Point Protocol (*PPP*) process	ppp	Enables transporting IP traffic across point-to-point links.
Universal edge PPP process	ppp-service	Enables transporting IP traffic across universal edge routers.
Point-to-Point Protocol over Ethernet process	pppoe	Allows users to connect to a network of hosts over a bridge or access concentrator.
Process health monitor process	process-monitor or pmond	Extends the SNMP *RMON* alarm infrastructure to provide predefined monitoring for a selected set of object instances (such as file system usage, CPU usage, and memory usage) and dynamic object instances (such as Junos OS processes). **NOTE**: The process health monitor process is enabled by default on the Routing Engines of MX Series routers, even when no service interfaces are configured. To disable this process, include the `disable` statement at the `[edit system processes process-monitor]` hierarchy level.
Redundancy interface management process	redundancy-interface-process	Serves as an active or backup process of an application server and can be configured to process traffic for more than one logical application server.
Remote operations process	remote-operations	Provides the *ping* and *traceroute* MIBs.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
Resource cleanup process	resource-cleanup	Enables cleaning of resources by entities other than the application itself.
Routing process	routing	Directs forwarding on the basis of routing tables, which maintain a record of the routes to various network destinations.
Traffic sampling control process	sampling	Performs packet sampling based on particular input interfaces and various fields in the packet header.
Session Border Control (*SBC*) configuration process	sbc-configuration-process	Configures the session border controller functionality that enables delivery of voice, video, and other multimedia services with assured quality and security.
SDK service process	sdk-service	Runs on the Routing Engine and enables communication between the SDK application and Junos OS. Although the SDK service process is present on the router, it is turned off by default.
Secure Neighbor Discovery (*SND*) protocol process	secure-neighbor-discovery or send	(EX Series switches and MX Series routers only) Provides support for protecting *NDP* messages.
Service Deployment System (*SDX*) process	service-deployment	Enables Junos OS to work with the Session and Resource Control (*SRC*) software.
Simple Network Management Protocol (SNMP) process	snmp	Enables the monitoring of network devices from a central location, and provides the device's SNMP primary agent.

Table 3: Junos OS Routing-Specific Processes *(Continued)*

Process	Name	Description
SONET Automatic Protection Switching (APS) process	sonet-aps	Monitors any SONET interface that participates in *APS*.
Static subscribers process	static-subscribers	Associates subscribers with statically configured interfaces, and provides dynamic service activation and activation for these subscribers.
Tunnel OAM process	tunnel-oamd	Enables the Operations, Administration, and Maintenance of Layer 2 tunneled networks.
Virtual Router Redundancy Protocol (*VRRP*) process	vrrp	(EX Series switches and MX Series routers only) Enables hosts on a LAN to make use of redundant routing platforms on that LAN without requiring more than the static configuration of a single default route on the hosts.
Watchdog timer process	watchdog	Enables the watchdog timer when Junos OS encounters a problem.

Default Directories for Junos OS File Storage on the Network Device

IN THIS SECTION

Generally, Junos OS files are stored in the following directories on the device:

- **/altconfig**—When you back up the currently running and active file system partitions on the device to standby partitions using the `request system snapshot` command, the **/config** directory is backed up to /

altconfig. Normally, the **/config** directory is on the CompactFlash card and **/altconfig** is on the hard disk.

- **/altroot**—When you back up the currently running and active file system partitions on the router to standby partitions using the `request system snapshot` command, the root file system (/) is backed up to **/altroot**. Normally, the root directory is on the CompactFlash card and **/altroot** is on the hard drive.
- **/config**—This directory is located on the primary boot device, that is, on the permanent storage from which the device booted (generally the CompactFlash card (device **wd0**) or internal flash storage). This directory contains the current operational router or switch configuration and the last three committed configurations, in the files **juniper.conf**, **juniper.conf.1**, **juniper.conf.2**, and **juniper.conf.3**, respectively.
- **/var**—This directory is located either on the hard drive (device **wd2**) or internal flash storage. It contains the following subdirectories:
  - **/home**—Contains users' home directories, which are created when you create user access accounts. For users using SSH authentication, their **.ssh** file, which contains their SSH key, is placed in their home directory. When a user saves or loads a configuration file, that file is loaded from the user's home directory unless the user specifies a full pathname.
  - **/db/config**—Contains up to 46 additional previous versions of committed configurations, which are stored in the files **juniper.conf.4.gz** through **juniper.conf.49.gz**.
  - **/log**—Contains system log and tracing files.
  - **/tmp**—Contains core files. The software saves up to five core files, numbered from 0 through 4. File number 0 is the oldest core file and file number 4 is the newest core file. To preserve the oldest core files, the software overwrites the newest core file, number 4, with any subsequent core file.

Each device ships with removable media (device **wfd0**) that contains a backup copy of Junos OS.

Directories on the Logical System

In addition to saving the configuration of logical systems in the current **juniper.conf** file, each logical system has an individual directory structure created in the **/var/logical-systems/*logical-system-name*** directory.

The **/var/logical-systems/*logical-system-name*** directory contains the following subdirectories:

- **/config**—Contains the current operational configuration specific to the logical system.
- **/log**—Contains system log and tracing files specific to the logical system.

To maintain backward compatibility for the log files with previous versions of Junos OS, a symbolic link (symlink) from the **/var/logs/*logical-system-name*** directory to the **/var/logical-systems/*logical-system-name*** directory is created when a logical system is configured.

- **/tmp**—Contains temporary files specific to the logical system.

This file system for each logical system enables logical system users to view trace logs and modify logical system files. Logical system administrators have full access to view and modify all files specific to the logical system.

Logical system users and administrators can save and load configuration files at the logical-system hierarchy level using the `save` and `load` configuration mode commands. In addition, they can also issue the `show log`, `monitor`, and `file` operational mode commands at the logical-system hierarchy level.

RELATED DOCUMENTATION

Format for Specifying Filenames and URLs in Junos OS CLI Commands | **66**

Junos OS Support for IPv4, IPv6, and MPLS Routing Protocols

Junos OS implements full IP routing functionality, providing support for IP version 4 and IP version 6 (IPv4 and IPv6, respectively). The routing protocols are fully interoperable with existing IP routing protocols, and they have been developed to provide the scale and control necessary for the Internet core.

Junos OS supports the following unicast routing protocols:

- BGP—Border Gateway Protocol version 4 is an *EGP* that guarantees loop-free exchange of routing information between routing domains (also called autonomous systems). BGP, in conjunction with Junos OS routing policies, provides a system of administrative checks and balances that can be used to implement peering and transit agreements.

- ICMP—Internet Control Message Protocol router discovery enables hosts to discover the addresses of operational routers on the subnet.

- IS-IS—Intermediate System to Intermediate System is a link-state *IGP* for IP networks that uses the *SPF* algorithm, which also is referred to as the *Dijkstra* algorithm, to determine routes. The Junos OS supports a new and complete implementation of the protocol, addressing issues of scale, convergence, and resilience.

- OSPF—Open Shortest Path First is an IGP that was developed for IP networks by the Internet Engineering Task Force (*IETF*). OSPF is a link-state protocol that makes routing decisions based on the *SPF* algorithm.

  OSPF Version 2 supports IPv4. OSPF Version 3 supports IPv6. The fundamental mechanisms of OSPF such as flooding, designated router (*DR*) election, area-based topologies, and the *SPF* calculations remain unchanged in OSPF Version 3. Some differences exist either because of changes in protocol semantics between IPv4 and IPv6, or because of the need to handle the increased address size of IPv6.

- RIP—Routing Information Protocol version 2 is a distance-vector IGP for IP networks based on the *Bellman-Ford* algorithm. RIP dynamically routes packets between a subscriber and a service provider without the subscriber having to configure BGP or to participate in the service provider's *IGP* discovery process.

Junos OS also provides the following routing and Multiprotocol Label Switching (*MPLS*) applications protocols:

- *Unicast* routing protocols:
  - BGP
  - ICMP
  - IS-IS
  - OSPF Version 2
  - RIP Version 2
- Multicast routing protocols:
  - DVMRP—Distance Vector Multicast Routing Protocol is a *dense-mode* (*flood-and-prune*) multicast routing protocol.
  - IGMP—Internet Group Management Protocol versions 1 and 2 are used to manage membership in multicast groups.
  - MSDP—Multicast Source Discovery Protocol enables multiple Protocol Independent Multicast (*PIM*) *sparse mode* domains to be joined. A rendezvous point (*RP*) in a PIM sparse mode domain has a peer relationship with an RP in another domain, enabling it to discover multicast sources from other domains.
  - PIM sparse mode and dense mode—Protocol-Independent Multicast is a multicast routing protocol. PIM sparse mode routes to multicast groups that might span wide-area and interdomain internets. PIM dense mode is a flood-and-prune protocol.

    - SAP/SDP—Session Announcement Protocol and Session Description Protocol handle conference session announcements.

- MPLS applications protocols:

    - LDP—The Label Distribution Protocol provides a mechanism for distributing labels in non-traffic-engineered applications. LDP enables routers to establish label-switched paths (LSPs) through a network by mapping network layer routing information directly to data-link layer switched paths. LSPs created by LDP can also traverse LSPs created by the Resource Reservation Protocol (*RSVP*).

    - MPLS—Multiprotocol Label Switching, formerly known as tag switching, enables you to manually or dynamically configure LSPs through a network. It lets you direct traffic through particular paths rather than rely on the IGP least-cost algorithm to choose a path.

    - RSVP—The Resource Reservation Protocol version 1 provides a mechanism for engineering network traffic patterns that is independent of the shortest path decided upon by a routing protocol. RSVP itself is not a routing protocol; it operates with current and future unicast and multicast routing protocols. The primary purpose of RSVP is to support dynamic signaling for MPLS LSPs.

RELATED DOCUMENTATION

Junos OS Overview

Junos OS Routing and Forwarding Tables

A major function of the Junos OS routing protocol process is to maintain the Routing Engine's routing tables and use these tables to determine the active routes to network destinations. The routing protocol process then installs these routes into the Routing Engine's forwarding table. The Junos OS kernel then copies this forwarding table to the Packet Forwarding Engine.

The routing protocol process maintains multiple routing tables. By default, it maintains the following three routing tables. You can configure additional routing tables to suit your requirements.

- *Unicast* routing table—Stores routing information for all unicast routing protocols running on the router. BGP, IS-IS, OSPF, and RIP all store their routing information in this routing table. You can configure additional routes, such as static routes, to be included in this routing table. BGP, IS-IS, OSPF, and RIP use the routes in this routing table when advertising routing information to their neighbors.

- Multicast routing table (cache)—Stores routing information for all the running multicast protocols. *DVMRP* and *PIM* both store their routing information in this routing table, and you can configure additional routes to be included in this routing table.

- MPLS routing table—Stores *MPLS* path and label information.

With each routing table, the routing protocol process uses the collected routing information to determine active routes to network destinations.

For *unicast* routes, the routing protocol process determines active routes by choosing the most preferred route, which is the route with the lowest preference value. By default, the route's preference value is simply a function of how the routing protocol process learned about the route. You can modify the default preference value using routing policy and with software configuration parameters.

For *multicast* traffic, the routing protocol process determines active routes based on traffic flow and other parameters specified by the multicast routing protocol algorithms. The routing protocol process then installs one or more active routes to each network destination into the Routing Engine's forwarding table.

RELATED DOCUMENTATION

Routing Policy Overview | **27**

Routing Policy Overview

By default, all routing protocols place their routes into the *routing table*. When advertising routes, the routing protocols by default advertise only a limited set of routes from the routing table. Specifically, each routing protocol exports only the active routes that were learned by that protocol. In addition, the interior gateway protocols (IS-IS, OSPF, and RIP) export the direct (interface) routes for the interfaces on which they are explicitly configured.

You can control the routes that a protocol places into each table and the routes from that table that the protocol advertises. You do this by defining one or more routing policies and then applying them to the specific routing protocol.

Routing policies applied when the routing protocol places routes into the routing table are referred to as *import policies* because the routes are being imported into the routing table. Policies applied when the routing protocol is advertising routes that are in the routing table are referred to as *export policies* because the routes are being exported from the routing table. In other words, the terms *import* and *export* are used with respect to the routing table.

A routing policy enables you to control (filter) which routes a routing protocol imports into the routing table and which routes a routing protocol exports from the routing table. A routing policy also enables you to set the information associated with a route as it is being imported into or exported from the routing table. Filtering imported routes enables you to control the routes used to determine active routes. Filtering routes being exported from the routing table enables you to control the routes that a protocol advertises to its neighbors.

A defined routing policy specifies the conditions to use to match a route and the action to perform on the route when a match occurs. For example, when a routing table imports routing information from a routing protocol, a routing policy might modify the route's preference, mark the route with a color to identify it and allow it to be manipulated later, or prevent the route from even being installed in a routing table. When a routing table exports routes into a routing protocol, a policy might assign metric values, modify the BGP community information, tag the route with additional information, or prevent the route from being exported altogether. You also can define policies for redistributing the routes learned from one protocol into another protocol.

RELATED DOCUMENTATION

Junos OS Routing and Forwarding Tables | **26**

Junos OS Support for IPv4, IPv6, and MPLS Routing Protocols | **24**

Junos OS Support for VPNs

Junos OS supports several types of virtual private networks (*VPNs*), including:

- Layer 2 VPNs link a set of sites that share routing information, and whose connectivity is controlled by a collection of policies. A Layer 2 VPN is not aware of routes within your network. It simply provides private links between sites over the service provider's existing public Internet *backbone*.

- Layer 3 VPNs are the same as a Layer 2 VPN, but it is aware of routes within your network, requiring more configuration on the part of the service provider than a Layer 2 VPN. The sites that make up a Layer 3 VPN are connected over a service provider's existing public Internet backbone.

- An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. As with other types of VPNs, an EVPN consists of customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) routers. The PE routers can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. Either an MX Series 5G Universal Routing Platform or a standalone switch can be configured to act as an MES. You can deploy multiple EVPNs within a service provider network, each providing network connectivity to a customer while ensuring that the traffic sharing on that network remains private.

- Interprovider VPNs supply connectivity between two VPNs in separate autonomous systems (*AS*s). This functionality can be used by a VPN user with connections to several Internet service providers (*ISP*s), or different connections to the same ISP in various geographic regions.
- Carrier-of-carrier VPNs allow a VPN service provider to supply VPN service to a someone who is also a service provider. The latter service provider supplies Internet or VPN service to an end user.

RELATED DOCUMENTATION

Junos OS Overview | 3

Configuring FIB Localization

IN THIS SECTION

FIB Localization Overview

On Juniper Networks devices, the forwarding table on the Packet Forwarding Engine, also referred to as forwarding information base (FIB), maintains the complete set of active IPv4 (inet) and IPv6 (inet6) routes. In Junos OS Release 11.4 and later, you can configure FIB localization for a Packet Forwarding Engine. FIB-localization characterizes Packet Forwarding Engines in a router as either “FIB-remote” or “FIB-local”.

FIB-local Packet Forwarding Engines install all routes from the default inet and inet6 route tables into the Packet Forwarding Engine forwarding hardware. FIB-remote Packet Forwarding Engines do not install all the routes for the inet and inet6 routing tables. However, they do maintain local and multicast routes.

FIB-remote Packet Forwarding Engines create a default (0/0) route in the Packet Forwarding Engine forwarding hardware for the inet and inet6 table. The default route references a next-hop or a unilist of next-hops that identify the FIB-local Packet Forwarding Engines that can perform full IP table lookups for received packets.

FIB-remote Packet Forwarding Engines forward received packets to the set of FIB-local Packet Forwarding Engines. The FIB-local Packet Forwarding Engines then perform full IP longest-match lookup on the destination address and forward the packet appropriately. The packet might be forwarded out of an egress interface on the same FIB-local Packet Forwarding Engine that performed the lookup or an egress interface on a different FIB-local or FIB-remote Packet Forwarding Engine. The packet might also be forwarded out of an FPC where FIB localization is not configured. The packet might also be received locally at the Routing Engine.

When FIB localization is configured on a router with some Flexible PIC Concentrators (FPCs) being FIB-remote and some others being FIB-local, packets arriving on the interface of the FIB-remote FPC are forwarded to one of the FIB-local FPCs for route lookup and forwarding.

The advantage of configuring FIB localization is that it enables upgrading the hardware forwarding table capacity of FIB-local Packet Forwarding Engines while not requiring upgrades to the FIB-remote Packet Forwarding Engines. In a typical network deployment, FIB-local Packet Forwarding Engines are core-facing, while FIB-remote Packet Forwarding Engines are edge-facing. The FIB-remote Packet Forwarding Engines also load-balance traffic over the available set of FIB-local Packet Forwarding Engines.

FIB localization is currently supported on specific Junos OS devices, including the T320, T640, T1600, and MX Series routers. To see if your hardware supports FIB localization, see the Juniper Networks Feature Explorer.

NOTE: On MX Series routers, you can configure multiservices Dense Port Concentrators (DPCs) as FIB-remote. However, only Modular Port Concentrators (MPCs) can be configured as FIB-local. FIB-localization is supported only for redundant link services intelligent queuing interfaces that carry Multilink Point-to-Point Protocol (MLPPP) traffic.

Example: Configuring Packet Forwarding Engine FIB Localization

IN THIS SECTION

This example shows how to configure Packet Forwarding Engine FIB localization.

Requirements

Before you begin:

1. Configure device interfaces and loopback interface addresses.

2. Configure static routes.

3. Configure OSPF and OSPFv3 and make sure that OSPF adjacencies and OSPF routes to loopback addresses are established.

This example uses the following hardware and software components:

- A T320, T640,T1600, or MX Series router.

- Junos OS Release 11.4 or later running on the router for T-Series routers. Junos OS Release 12.3 or later running on the router for MX Series routers.

Overview

In this example, you configure the chassis for IPv4 and IPv6 routes and FIB localization on Router R0 and then configure the edge-facing Packet Forwarding Engines on FPC0 as `fib-remote` and the core-facing Packet Forwarding Engines on FPC1 and FPC2 as `fib-local`. You then configure a routing policy named `fib-policy` with the `no-route-localize` option to ensure that all routes from a specified route filter are installed on the FIB-remote FPC.

Configuration

IN THIS SECTION

Procedure

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.

NOTE: Configuring FIB local results in a reboot of the related line card to activate the changes.

R0

```
set chassis fpc 0 route-localization fib-remote
set chassis fpc 1 route-localization fib-local
set chassis fpc 2 route-localization fib-local
set chassis route-localization inet
set chassis route-localization inet6
set policy-options policy-statement fib-policy term a from route-filter 10.4.4.4/32 exact
set policy-options policy-statement fib-policy term a then no-route-localize
set policy-options policy-statement fib-policy term b from route-filter fec0:4444::4/128 exact
set policy-options policy-statement fib-policy term b then no-route-localize
set policy-options policy-statement fib-policy then accept
set routing-options forwarding-table export fib-policy
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the Junos OS CLI, see the Junos OS CLI User Guide.

To configure Packet Forwarding Engine FIB localization:

1. Configure route localization or FIB localization for IPv4 and IPv6 traffic.

   ```
   [edit chassis]
   user@R0# set route-localization inet
   user@R0# set route-localization inet6
   ```

2. Configure the Packet Forwarding Engine of an FPC as either `fib-local` or `fib-remote`.

   ```
   [edit chassis]
   user@R0# set fpc 0 route-localization fib-remote
   user@R0# set fpc 1 route-localization fib-local
   user@R0# set fpc 2 route-localization fib-local
   ```

3. Configure the routing policy by including the `no-route-localize` statement to enable the forwarding table policy to mark route prefixes such that the routes are installed into forwarding hardware on the FIB-remote Packet Forwarding Engines.

   ```
   [edit policy-options]
   user@R0# set policy-statement fib-policy term a from route-filter 10.4.4.4/32 exact
   user@R0# set policy-statement fib-policy term a then no-route-localize
   user@R0# set policy-statement fib-policy term b from route-filter fec0:4444::4/128 exact
   user@R0# set policy-statement fib-policy term b then no-route-localize
   user@R0# set policy-statement fib-policy then accept
   ```

4. Enable the routing policy in the forwarding table by configuring the forwarding table with the `fib-policy` statement.

   ```
   [edit routing-options]
   user@R0# set forwarding-table export fib-policy
   ```

   NOTE: At least, one Packet Forwarding Engine must be configured as `fib-local` for the commit operation to be successful. If you do not configure `fib-local` for the Packet Forwarding Engine, the CLI displays an appropriate error message and the commit fails.

Results

From configuration mode, confirm your configuration by entering the `show chassis` and `show policy-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R0# show chassis
fpc 0 {
    route-localization fib-remote;
}
fpc 1 {
    route-localization fib-local;
}
fpc 2 {
    route-localization fib-local;
}
route-localization {
```

```
    inet;
    inet6;
}
```

```
user@R0# show policy-options
policy-statement fib-policy {
    term a {
        from {
            route-filter 10.4.4.4/32 exact;
        }
        then no-route-localize;
    }
    term b {
        from {
            route-filter fec0:4444::4/128 exact;
        }
        then no-route-localize;
    }
    then accept;
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Policy Configuration

Purpose

Verify that the configured policy exists.

Action

Issue the `show policy fib-policy` command to check that the configured policy `fib-policy` exists.

```
user@R0> show policy fib-policy
Policy fib-policy:
    Term a:
        from
             route filter:
                 10.4.4.4/32 exact
        then no-route-localize
    Term b:
        from
             route filter:
                 fec0:4444::4/128 exact
        then no-route-localize
    Term unnamed:
        then accept
```

Verifying FIB-Localization Configuration

Purpose

Verify FIB-localization configuration details by using the `show route localization` and `show route localization detail` commands.

Action

```
user@R0> show route localization
FIB localization ready FPCs (and FIB-local Forwarding Engine addresses)
  FIB-local:  FPC2(4,5)
  FIB-remote: FPC0, FPC1
  Normal:     FPC3, FPC4, FPC5, FPC6, FPC7
```

```
user@R0> show route localization detail
FIB localization ready FPCs (and FIB-local Forwarding Engine addresses)
  FIB-local:  FPC2(4,5)
  FIB-remote: FPC0, FPC1
  Normal:     FPC3, FPC4, FPC5, FPC6, FPC7
FIB localization configuration
  Protocols:  inet, inet6
  FIB-local:  FPC2
  FIB-remote: FPC0, FPC1
Forwarding Engine addresses
  FPC0: 1
  FPC1: 2
  FPC2: 4, 5
  FPC3: 6
  FPC4: 8
  FPC5: 11
  FPC6: 13
  FPC7: 15
```

Verifying Routes After the Policy Is Applied

Purpose

Verify that routes with the `no-route-localize` policy option are installed on the `fib-remote` FPC.

Action

```
user@R0> show route 10.4.4.4/32 extensive
```

```


inet.0: 30 destinations, 30 routes (29 active, 0 holddown, 1 hidden)
10.4.4.4/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.4.4.4/32 -> {10.130.0.2 Flags no-localize}
                                          ^^^^^^^^^^^^^^^^^
```

```
          *Static Preference: 5
                  Next hop type: Router, Next hop index: 629
                  Next-hop reference count: 3
                  Next hop: 10.130.0.2 via ge-1/0/4.0, selected
                  State: <Active Int="">
                           Age: 10:33
                           Task: RT
                           Announcement bits (1): 0-KRT
                           AS path: I</Active
>
```

RELATED DOCUMENTATION

fib-local

fib-remote

no-route-localize

route-localization

CHAPTER 2

Junos OS Security Overview

IN THIS CHAPTER

Junos OS Features for Device Security

IN THIS SECTION

Device security consists of three major elements: Physical security of the hardware, operating system security, and security that can be affected through configuration.

Physical security involves restricting access to the device. Exploits that can easily be prevented from remote locations are extremely difficult or impossible to prevent if an attacker can gain access to the device's management port or console. The inherent security of Junos OS also plays an important role in router security. Junos OS is extremely stable and robust, and provides features to protect against attacks, allowing you to configure the device to minimize vulnerabilities.

The following are Junos OS features available to improve device security:

Methods of Remote Access for Device Management

When you first install Junos OS, all remote access to the device is disabled, thereby ensuring that remote access is possible only if deliberately enabled by an authorized user. You can establish remote communication with a device in one of the following ways:

- Out-of-band management: Enables connection to the device through an interface dedicated to device management. Juniper Networks devices support out-of-band management with a dedicated management Ethernet interface, as well as EIA-232 console and auxiliary ports. On all devices other than the TX Matrix Plus, T1600, T1600 or T4000 devices connected to a TX Matrix Plus device in a routing matrix, and PTX Series Packet Transport Routers, the management interface is fxp0. On a TX Matrix Plus, T1600, T1600 or T4000 devices in a routing matrix, and PTX Series Packet Transport Routers, the management Ethernet Interface is labeled em0. The management Ethernet interface connects directly to the Routing Engine. No transit traffic is allowed through this interface, providing complete separation of customer and management traffic and ensuring that congestion or failures in the transit network do not affect the management of the device.

- Inband management: Enables connection to the devices using the same interfaces through which customer traffic flows. Although this approach is simple and requires no dedicated management resources, it has two disadvantages:

  - Management flows and transit traffic flows are mixed together. Any attack traffic that is mixed with the normal traffic can affect the communication with the device.

  - The links between device components might not be totally trustworthy, leading to the possibility of wiretapping and replay attacks.

For management access to the device, the standard ways to communicate with the device from a remote console are with Telnet and SSH. SSH provides secure encrypted communications and is therefore useful for inband device management. Telnet provides unencrypted, and therefore less secure, access to the device.

Junos OS Supported Protocols and Methods for User Authentication

On a device, you can create local user login accounts to control who can log in to the device and the access privileges they have. A password, either an SSH key or a Message Digest 5 (MD5) password, is associated with each login account. To define access privileges, you create login classes into which you group users with similar jobs or job functions. You use these classes to explicitly define what commands their users are and are not allowed to issue while logged in to the device.

The management of multiple devices by many different personnel can create a user account management problem. One solution is to use a central authentication service to simplify account management, creating and deleting user accounts only on a single, central server. A central authentication system also simplifies the use of one-time password systems such as SecureID, which offer protection against password sniffing and password replay attacks (attacks in which someone uses a captured password to pose as a device administrator).

Junos OS supports two protocols for central authentication of users on multiple devices:

- Terminal Access Controller Access Control System Plus (TACACS+).
- Remote Authentication Dial-In User Service (RADIUS), a multivendor IETF standard whose features are more widely accepted than those of TACACS+ or other proprietary systems. All one-time-password system vendors support RADIUS.

Junos OS also supports the following authentication methods:

- Internet Protocol Security (IPsec). IPsec architecture provides a security suite for the IPv4 and IPv6 network layers. The suite provides such functionality as authentication of origin, data integrity, confidentiality, replay protection, and nonrepudiation of source. In addition to IPsec, Junos OS supports the Internet Key Exchange (IKE), which defines mechanisms for key generation and exchange, and manages security associations (SAs).
- MD5 authentication of MSDP peering sessions. This authentication provides protection against spoofed packets being introduced into a peering session.
- SNMPv3 authentication and encryption. SNMPv3 uses the user-based security model (USM) for message security and the view-based access control model (VACM) for access control. USM specifies authentication and encryption. VACM specifies access-control rules.

Junos OS Plain-Text Password Requirements

Junos OS has special requirements when you create plain-text passwords on a device. The default requirements for plain-text passwords are as follows:

- The password must be between 6 and 128 characters long.
- You can include uppercase letters, lowercase letters, numbers, punctuation marks, and any of the following special characters:
  ! @ # $ % ^ & * , + = < > : ;
  Control characters are not recommended.
- The password must contain at least one change of case or character class.

You can change the requirements for plain-text passwords.

You can include the `plain-text-password` statement at the following hierarchy levels:

- `[edit system diag-port-authentication]`
- `[edit system pic-console-authentication]`

- `[edit system root-authentication]`
- `[edit system login user` *`username`* `authentication]`

Junos OS Support for Routing Protocol Security Features and IPsec

The main task of a device is to forward user traffic toward its intended destination based on the information in the device's routing and forwarding tables. You can configure routing policies that define the flows of routing information through the network, controlling which routes the routing protocols place in the routing tables and which routes they advertise from the tables. You can also use routing policies to change specific route characteristics, change the BGP route flap-damping values, perform per-packet load balancing, and enable *class of service* (CoS).

Attackers can send forged protocol packets to a device with the intent of changing or corrupting the contents of its routing table or other databases, which can degrade the functionality of the device. To prevent such attacks, you must ensure that devices form routing protocol peering or neighboring relationships with trusted peers. One way to do this is by authenticating routing protocol messages. The Junos OS BGP, IS-IS, OSPF, RIP, and RSVP protocols all support HMAC-MD5 authentication, which uses a secret key combined with the data being protected to compute a hash. When the protocols send messages, the computed hash is transmitted with the data. The receiver uses the matching key to validate the message hash.

Junos OS supports the IPsec security suite for the IPv4 and IPv6 network layers. The suite provides such functionality as authentication of origin, data integrity, confidentiality, replay protection, and nonrepudiation of source. Junos OS also supports IKE, which defines mechanisms for key generation and exchange, and manages SAs.

Junos OS Support for Firewall Filters

Firewall filters allow you to control packets transiting the device to a network destination and packets destined for and sent by the device. You can configure firewall filters to control which data packets are accepted on and transmitted from the physical interfaces, and which local packets are transmitted from the physical interfaces and the Routing Engine. Firewall filters provide a means of protecting your device from excessive traffic. Firewall filters that control local packets can also protect your device from external aggressions, such as DoS attacks.

To protect the Routing Engine, you can configure a *firewall filter* only on the device's loopback interface. Adding or modifying filters for each interface on the device is not necessary. You can design firewall filters to protect against ICMP and Transmission Control Protocol (TCP) connection request (SYN) floods and to rate-limit traffic being sent to the Routing Engine.

Junos OS Support Distributed Denial-of-Service Protection

A denial-of-service attack is any attempt to deny valid users access to network or server resources by using up all the resources of the network element or server. Distributed denial-of-service attacks involve an attack from multiple sources, enabling a much greater amount of traffic to attack the network. The attacks typically use network protocol control packets to trigger a large number of exceptions to the device's control plane. This results in an excessive processing load that disrupts normal network operations.

Junos OS DDoS protection enables the device to continue functioning while under an attack. It identifies and suppresses malicious control packets while enabling legitimate control traffic to be processed. A single point of DDoS protection management enables network administrators to customize profiles for their network control traffic. Protection and monitoring persists across graceful Routing Engine switchover (GRES) and unified in-service-software-upgrade (ISSU) switchovers. Protection is not diminished as the number of subscribers increases.

To protect against DDoS attacks, you can configure policers for host-bound exception traffic. The policers specify rate limits for individual types of protocol control packets or for all control packet types for a protocol. You can monitor policer actions for packet types and protocol groups at the level of the device, Routing Engine, and line cards. You can also control logging of policer events.

Flow detection is an enhancement to DDoS protection that supplements the DDoS policer hierarchies by using a limited amount of hardware resources to monitor the arrival rate of host-bound flows of control traffic. Flow detection is much more scalable than a solution based on filter policers. Filter policers track all flows, which consumes a considerable amount of resources. In contrast, flow detection only tracks flows it identifies as suspicious, using far fewer resources to do so.

The flow detection application has two interrelated components, detection and tracking. Detection is the process where flows suspected of being improper are identified and subsequently controlled. Tracking is the process where flows are tracked to determine whether they are truly hostile and when these flows recover to within acceptable limits.

Junos OS Auditing Support for Security

Junos OS logs significant events that occur on the device and within the network. Although logging itself does not increase security, you can use the system logs to monitor the effectiveness of your security policies and device configurations. You can also use the logs when reacting to a continued and deliberate attack as a means of identifying the source address, device, or port of the attacker's traffic. You can configure the logging of different levels of events, from only critical events to all events, including informational events. You can then inspect the contents of the system log files either in real time or later.

Debugging and troubleshooting are much easier when the timestamps in the system log files of all devices are synchronized, because events that span the network might be correlated with synchronous entries in multiple logs. Junos OS supports the Network Time Protocol (NTP), which you can enable on the device to synchronize the system clocks of devices and other networking equipment. By default, NTP operates in an unauthenticated mode. You can configure various types of authentication, including an HMAC-MD5 scheme.

RELATED DOCUMENTATION

Overview of IPsec

Junos OS System Log Overview

Junos OS Default Settings for Device Security

Junos OS protects against common network device security weaknesses with the following default settings:

- Junos OS does not forward directed broadcast messages. Directed broadcast services send ping requests from a spoofed source address to a broadcast address and can be used to attack other Internet users. For example, if broadcast ping messages were allowed on the 200.0.0.0/24 network, a single ping request could result in up to 254 responses to the supposed source of the ping. The source would actually become the victim of a denial-of-service (DoS) attack.

- Generally, by default, only console access to the device is enabled. Remote management access to the device and all management access protocols, including Telnet, FTP, and SSH (Secure Shell), are disabled by default, unless the device setup specifically includes a factory-installed DHCP configuration.

- Junos OS does not support the SNMP set capability for editing configuration data. Although the software supports the SNMP set capability for monitoring and troubleshooting the network, this support exposes no known security issues. (You can configure the software to disable this SNMP set capability.)

- Junos OS ignores martian (intentionally non-routable) IP addresses that contain the following prefixes: 0.0.0.0/8, 127.0.0.0/8, 128.0.0.0/16, 191.255.0.0/16, 192.0.0.0/24, 223.255.55.0/24, and 240.0.0.0/4. Martian addresses are reserved host or network addresses about which all routing information should be ignored.

CHAPTER 3

Junos OS Configuration Overview

IN THIS CHAPTER

Junos OS Configuration Basics

Usually, your Juniper Networks device comes with Junos OS installed on it, unless you specifically order it without the operating system. When Junos OS is pre-installed, you simply power on the device and all software starts automatically. You just need to configure the device so it will be ready to participate in the network.

To configure the Junos OS, you must specify a hierarchy of configuration statements which define the preferred software properties. You can configure all properties of the Junos OS, including interfaces, general routing information, routing protocols, and user access, as well as some system hardware properties. After you have created a candidate configuration, you commit the configuration to be evaluated and activated by Junos OS.

RELATED DOCUMENTATION

Junos OS Configuration from External Devices | **48**

Methods for Configuring Junos OS | **45**

Initial Router or Switch Configuration Using Junos OS | **52**

Methods for Configuring Junos OS

IN THIS SECTION

Depending on specific device support, you can use the methods shown in Table 4 on page 45 to configure Junos OS. For more information, see the Juniper Networks Feature Explorer.

Table 4: Methods for Configuring Junos OS

Method	Description
Command-line interface (CLI)	Create the configuration for the device using the CLI. You can enter commands from a single command line, and scroll through recently executed commands.
ASCII file	Load an ASCII file containing a configuration that you created earlier, either on this system or on another system. You can then activate and run the configuration file, or you can edit it using the CLI and then activate it.
J-Web graphical user interface (GUI)	Use the J-Web GUI to configure the device. J-Web enables you to monitor, configure, troubleshoot, and manage the router on a client by means of a Web browser. The J-Web GUI is supported on only certain Juniper Networks devices. For more information, see the Juniper Networks Feature Explorer.
Junos XML management protocol (API)	Client applications use the Junos XML management protocol to monitor and configure Juniper Networks devices. The Junos XML management protocol is customized for Junos OS, and operations in the API are equivalent to those in the CLI.

Table 4: Methods for Configuring Junos OS *(Continued)*

Method	Description
NETCONF application programming interface (API)	Client applications use the NETCONF XML management protocol to monitor and configure supported devices. The NETCONF XML management protocol includes features that accommodate the configuration data models of multiple vendors.
Configuration commit scripts	Create scripts that run at commit time to enforce custom configuration rules. Commit scripts are written in Python, Stylesheet Language Alternative syntaX (SLAX), or Extensible Stylesheet Language Transformations (XSLT).

The following sections describe the methods you can use to configure Junos OS:

Junos OS Command-Line Interface

The Junos OS CLI is a straightforward terminal-based command interface. You use Emacs-style keyboard sequences to move around on a command line and scroll through a buffer that contains recently executed commands. You type commands on a single line, and the commands are executed when you press the Enter key. The CLI also provides command help and command completion.

ASCII File

You can load an ASCII file containing a configuration that you created earlier, either on this system or another system. You can then activate and run the configuration file as is, or you can edit it using the CLI and then activate it.

J-Web Package

As an alternative to entering CLI commands, Junos OS supports the J-Web GUI. The J-Web user interface enables 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.

The J-Web user interface is an optional, licensed software package (jweb package) on M Series and TSeries routers. The jweb package is not included in jinstall and jbundle software bundles. It must be installed separately. To install the package on M Series and T Series routers, follow the procedure described in the Software Installation and Upgrade Guide.

J-Web supports weak (56-bit) encryption by default. This enables non-US customers to install J-Web and use HTTPS connections for J-Web access. US customers can also install the jcrypto strong encryption package. This package automatically overrides the weak encryption.

NOTE: Because the J-Web package is bundled separately from other packages, it is possible to have a version mismatch between J-Web and other Junos OS packages you have installed.

To check for a version mismatch, use the `show system alarms` CLI command. If the version number does not match exactly, a system alarm appears.

Junos XML Management Protocol Software

The Junos XML Management Protocol is an XML-based protocol that client applications use to monitor and configure Juniper Networks devices. It uses an XML-based data encoding for the configuration data and remote procedure calls. This API is customized for Junos OS, and operations in the API are equivalent to CLI commands.

NETCONF XML Management Protocol Software

The NETCONF XML management protocol is an XML-based protocol that client applications use to monitor and configure network devices. It uses an XML-based data encoding for the configuration data and remote procedure calls. NETCONF includes features that accommodate the configuration data models of multiple vendors. Juniper Networks provides a set of Perl modules that enable Perl client applications to communicate with the NETCONF server on Junos devices. The Perl modules enable you to develop custom applications for configuring and monitoring Junos devices.

Configuration Commit Scripts

You can create and use scripts that run at commit time to enforce custom configuration rules. If a configuration breaks the custom rules, the script can generate actions that the Junos OS performs. These actions include:

- Generating custom error messages
- Generating custom warning messages
- Generating custom system log messages
- Making changes to the configuration

Configuration commit scripts also enable you to create macros, which expand simplified custom aliases for frequently used configuration statements into standard Junos OS configuration statements. Commit scripts are written in Python, Stylesheet Language Alternative syntaX (SLAX), or Extensible Stylesheet Language Transformations (XSLT).

RELATED DOCUMENTATION

CLI Explorer
CLI User Guide
Junos OS Automation Scripting User Guide
Junos OS Configuration from External Devices | **48**
NETCONF XML Management Protocol Developer Guide
Software Installation and Upgrade Guide

Junos OS Configuration from External Devices

You can configure Junos OS network device from a *system console* connected to the console port or by using *Telnet* to access the device remotely. External management hardware can be connected to the Routing Engine and the Junos OS through these ports:

- Console port
- Auxiliary port
- Ethernet management port

NOTE: See hardware guide for your particular Junos OS device for instructions about how to connect external hardware to the console, auxiliary, and/or Ethernet management ports. Capabilities and features can vary depending on device model.

RELATED DOCUMENTATION

Methods for Configuring Junos OS | **45**
Configuring Junos OS to Set Console and Auxiliary Port Properties | **72**

The 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 current 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. (The remaining rollback files, 4 through 49, are located in **/var/db/config**.)

If the recovery configuration file **rescue.conf.gz** exists, this file is also located 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.

  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 `synchronization` command.

RELATED DOCUMENTATION

Configuring Junos OS for the First Time on a Device with a Single Routing Engine

Configuration Groups Overview

IN THIS SECTION

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

How Configuration Groups Work

Configuration groups enable 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. 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 Juniper Networks devices. For example, you can group statements that are repeated in many places in the configuration, such as when configuring interfaces. By grouping statements, you can limit configuration updates to just the group.

You can also use wildcards in a configuration group. Any object that matches the wildcard expression inherits the group configuration data.

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 you can use throughout the configuration but that are known only to the 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. The target automatically inherits data values that you change in the configuration group. The target does not need to contain the inherited information. However, the inherited values can be overridden in the target without affecting the source from which they were inherited.

This inheritance model enables you to see only the instance-specific information without seeing the inherited details. A command pipe in configuration mode enables you to display the inherited data.

2

PART

Configuring and Administering Junos Devices

CHAPTER 4

Configuring Junos Devices

IN THIS CHAPTER

Initial Router or Switch Configuration Using Junos OS

This topic provides an overview of initial network device configuration tasks using Junos OS.

When you turn on a device for the first time, Junos OS automatically boots and starts. You must enter basic configuration information so the device is on the network and you can log in to it over the network.

To configure the device initially, you must connect through the console port.

When you first connect to the console of a device that has not yet been configured, log in as the user `root`. At first, the root account requires no password. You can see that you are the user `root`, because the command prompt shows the username `root@#`.

You must start the Junos OS command-line interface (CLI) using the command `cli`. The command prompt `root@>` indicates that you are the user `root` and that you are in Junos OS operational mode. Enter Junos OS configuration mode by typing the command `configure`. The command prompt `root@#` indicates that you are in the Junos OS configuration mode.

When you first configure a device, you should configure the following basic properties:

- Device hostname
- Domain name
- IP address of the device management Ethernet interface. To find the management Ethernet interface that you should use for configuration, see Supported Routing Engines by Router.
- IP address of a backup router
- IP address of one or more DNS name servers on your network
- Password for the root account

RELATED DOCUMENTATION

Configuring Junos OS for the First Time on a Device with a Single Routing Engine | 53

Configuring Junos OS for the First Time on a Device with Dual Routing Engines | 58

Supported Routing Engines by Router

Junos OS Configuration Using the CLI

Configuring Junos OS for the First Time on a Device with a Single Routing Engine

To configure the Junos OS for the first time on a router with a single Routing Engine and no base configuration, follow these steps:

1. Connect to the device through the console port.
2. Power on the device and wait for it to boot.

   The Junos OS boots automatically. The boot process is complete when you see the `login:` prompt on the console.
3. Log in as the user `root`.

   Initially, the `root` user account requires no password. You can see that you are the `root` user, because the prompt on the device shows the username `root@#`.

4. Start the Junos OS command-line interface (CLI):

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

5. Enter Junos OS configuration mode:

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

6. Configure the hostname of the device. We do not recommend spaces in the router name. However, if the name does include spaces, enclose the entire name in quotation marks (" ").

```
[edit]
root@# set system host-name hostname
```

7. Set the root password, entering either a clear-text password that the system will encrypt, a password that is already encrypted, or an SSH public key string.

   Choose one of the following:

   a. To enter a clear-text password, use the following command:

   ```
   [edit]
   root@# set system root-authentication plain-text-password
   New password: type password
   Retype new password: retype password
   ```

   b. To enter a password that is already encrypted, use the following command:

   ```
   [edit]
   root@# set system root-authentication encrypted-password encrypted-password
   ```

   c. To enter an SSH public key, use the following command:

   ```
   [edit]
   root@# set system root-authentication ssh-rsa key
   ```

8. Configure the device domain name:

   ```
   [edit]
   root@# set system domain-name domain-name
   ```

   NOTE: Before you begin the next step, see Supported Routing Engines by Router to find the management Ethernet interface that you should use to perform this configuration.

9. Configure the IP address and prefix length for the device management Ethernet interface. The management Ethernet interface provides a separate out-of-band management network for the device.

   - For devices that use management Ethernet interface fxp0:

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

   - For devices that use management Ethernet interface em0:

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

10. Configure the IP address of a backup or default network device. Choose a device that is directly connected to the local router by way of the management interface. This backup is used only when it is booting and only or when the Junos routing software (the routing protocol process, rpd) is not running.

    For devices with two Routing Engines, the backup Routing Engine, `RE1`, uses the backup device as a default gateway after the device boots. This enables you to access the backup Routing Engine. (`RE0` is the default primary Routing Engine.)

    NOTE: The backup Routing Engine does not support more than 16 backup routing destinations. If you configure more than 16 destinations on the backup Routing

Engine, the Junos OS ignores any destination addresses after the sixteenth address and displays a commit-time warning message to this effect.

```
[edit]
root@# set system backup-router address
```

11. Configure the IP address of a DNS server. The router uses the DNS name server to translate hostnames into IP addresses.

```
[edit]
root@# set system name-server address
```

12. Optionally, display the configuration statements:

```
[edit]
root@ show
system {
    host-name hostname;
    domain-name domain.name;
    backup-router address;
    root-authentication {
        (encrypted-password "password" | public-key);
        ssh-dsa "public-key";
        ssh-ecdsa "public-key";
        ssh-rsa "public-key";
    }
    name-server {
        address;
    }
    interfaces {
        fxp0 {
            unit 0 {
                family inet {
                    address address ;
                }
            }
        }
    }
}
```

On devices that use management Ethernet interface em0, you will see em0 in place of fxp0 in the `show` command output.

13. Commit the configuration, which activates the configuration on the device:

```
[edit]
root@# commit
```

After committing the configuration, you see the newly configured hostname appear after the username in the prompt—for example, `user@hostname#`.

A basic configuration for Junos OS is now set on the device.

If you want to configure additional Junos OS properties at this time, remain in the CLI configuration mode and add the necessary configuration statements. You need to commit your configuration changes to activate them on the device.

14. Exit from the CLI configuration mode.

```
[edit]
root@hostname# exit
root@hostname>
```

15. Back up the configuration.

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

The `request system snapshot` command causes the root file system to be backed up to **/altroot**, and **/config** to be backed up to **/altconfig**. The root and **/config** file systems are on the device's CompactFlash card, and the **/altroot** and **/altconfig** file systems are on the device's hard drive.

NOTE: After you issue the `request system snapshot` command, you cannot easily return to the previous configuration, because the running copy and the backup copies are identical.

RELATED DOCUMENTATION

Initial Router or Switch Configuration Using Junos OS | 52

Configuring Junos OS for the First Time on a Device with Dual Routing Engines

If a device has dual Routing Engines, you can create configuration groups and use the same configuration for both Routing Engines. This ensures that the configuration will not change during a failover scenario because of the identical configuration shared between the Routing Engines.

Configure the hostnames and addresses of the two Routing Engines using configuration groups at the `[edit groups]` hierarchy level. Use the reserved configuration group `re0` for the Routing Engine in slot 0 and `re1` for the Routing Engine in slot 1 to define Routing Engine-specific parameters. Configuring `re0` and `re1` groups enables both Routing Engines to use the same configuration file.

Use the `apply-groups` statement to apply the apply the configuration to the device.

The `commit synchronize` command commits the same configuration on both Routing Engines. The command makes the active or applied configuration the same for both Routing Engines with the exception of the groups, `re0` being applied to only `RE0` and `re1` being applied only to `RE1`. If you do not synchronize the configurations between two Routing Engines and one of them fails, the router may not forward traffic correctly, because the backup Routing Engine may have a different configuration.

To initially configure a device with dual Routing Engines that have no base configuration, follow these steps:

1. If you have not already done so, refer "Configuring Junos OS for the First Time on a Device with a Single Routing Engine" on page 53 and follow the steps to initially configure the backup Routing Engine.
2. 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@host# set groups re0
```

3. Navigate to the `groups re0` level of the configuration hierarchy.

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

4. Specify the device hostname.

```
[edit groups re0]
root@host# 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>
```

NOTE: Before you begin the next step, see Supported Routing Engines by Router to find the management Ethernet interface that you should use to perform this configuration.

5. Configure the IP address and prefix length for the device management Ethernet interface. The management Ethernet interface provides a separate out-of-band management network for the device.

   - For devices using the management Ethernet interface fxp0:

   ```
   [edit groups re0]
   root@host# set interfaces fxp0 unit 0 family inet address address/prefix-length
   ```

   - For devices that use the management Ethernet interface em0:

   ```
   [edit groups re0]
   root@host# set interfaces em0 unit 0 family inet address address/prefix-length
   ```

6. Set the loopback interface address for the `re0` configuration group:

```
[edit groups re0]
root@host# set interfaces lo0 unit 0 family inet address address/prefix-length
```

7. Return to the top level of the hierarchy.

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

   The next steps repeat for `re1` the same steps as were done for the `re0` configuration group.

8. Create the configuration group `re1`.

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

9. Navigate to the `groups re1` level of the configuration hierarchy.

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

10. Specify the device hostname.

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

> **NOTE**: Before you begin the next step, see Supported Routing Engines by Router to find the management Ethernet interface that you should use to perform this configuration.

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

   - For devices that use the management Ethernet interface fxp0:

```
[edit groups re1]
root@host# set interfaces fxp0 unit 0 family inet address address/prefix-length
```

- For devices that use the management Ethernet interface em0:

  ```
  [edit groups re1]
  root@host# set interfaces em0 unit 0 family inet address address/prefix-length
  ```

12. Set the loopback interface address for `re1` configuration group:

    ```
    [edit groups re1]
    root@host# set interfaces lo0 unit 0 family inet address address/prefix-length
    ```

13. Once both configuration groups have been set up, return to the top level of the hierarchy.

    ```
    [edit groups re1]
    root@host# top
    ```

14. Use the `apply-groups` statement to apply the configuration to the device.

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

15. Configure Routing Engine redundancy:

    ```
    [edit]
    root@host# set chassis redundancy routing-engine 0 master
    root@host# set chassis redundancy routing-engine 1 backup
    ```

16. Save the configuration change on both Routing Engines:

    ```
    [edit]
    user@host> commit synchronize
    ```

After the configuration changes are saved, complete the management console configuration.

1. Set the root password by choosing one of the following:

   - To enter a clear-text password, use the following command:

     ```
     [edit]
     root@host# set system root-authentication plain-text-password
     ```

```
New password: type password
Retype new password: retype password
```

- To enter a password that is already encrypted, use the following command:

```
[edit]
root@host# set system root-authentication encrypted-password encrypted-password
```

- To enter an SSH public key, use the following command:

```
[edit]
root@host# set system root-authentication ssh-rsa key
```

2. Configure the IP address of the DNS server.

```
[edit ]
root@host# set system name-server address
```

3. Configure the router domain name:

```
[edit ]
root@host# set system domain-name domain-name
```

4. Configure the IP address of a backup or default network device. A backup device is used only while the routing protocol process is not running. Choose a backup device that is directly connected to the local device by way of the management interface. The device uses this backup only when it is booting and or when the Junos routing software (the routing protocol process, rpd) is not running. For more information, see *Configuring a Backup Router*.

   For devices with two Routing Engines, the backup Routing Engine, `RE1`, uses the backup as a default gateway after the device boots. This enables you to access the backup Routing Engine. (`RE0` is the default primary Routing Engine.)

   NOTE: The backup router Routing Engine does not support more than 16 backup destinations. If you configure more than 16 destinations on the backup Routing Engine, the Junos OS ignores any destination addresses after the sixteenth address and displays a commit-time warning message to this effect.

```
[edit]
root@host# set system backup-router address
```

5. Optionally, display the configuration statements:

```
[edit]
root@ show
system {
    host-name hostname;
    domain-name domain.name;
    backup-router address;
    root-authentication {
        (encrypted-password "password" | public-key);
        ssh-dsa "public-key";
        ssh-ecdsa "public-key";
        ssh-rsa "public-key";
    }
    name-server {
        address;
    }
    interfaces {
        fxp0 {
            unit 0 {
                family inet {
                    address address ;
                }
            }
        }
    }
}
```

On devices that use management Ethernet interface em0, you will see em0 in place of fxp0 in the `show` command output.

6. After you are satisfied that the configuration is successfully running, issue the `request system snapshot` command to back up the new configuration on both primary and backup Routing Engines.

```
 {master}
user@host> request system snapshot
```

   The root file system is backed up to **/altroot**, and **/config** is backed up to **/altconfig**. The root and **/config** file systems are on the device’s CompactFlash card, and the **/altroot** and **/altconfig** file systems are on the device’s hard drive.

   NOTE: After you issue the `request system snapshot` command, you cannot return to the previous configuration, because the running copy and backup copy are identical.

For information about creating configuration groups, see Junos OS CLI User Guide.

For information about configuring high availability features for redundant Routing Engine systems and the `re0` group, see Junos OS High Availability User Guide.

RELATED DOCUMENTATION

Configuring Automatic Mirroring of the CompactFlash Card on the Hard Disk | **68**

Configuring Junos OS for the First Time on a Device with a Single Routing Engine | **53**

Default Directories for Junos OS File Storage on the Network Device | **22**

Format for Specifying IP Addresses, Network Masks, and Prefixes in Junos OS Configuration Statements | **65**

Initial Router or Switch Configuration Using Junos OS | **52**

Supported Routing Engines by Router

How to Improve Commit Time When Using Configuration Groups

You use configuration groups to apply configurations across other hierarchies without re-entering configuration data. You can specify every configuration detail in a configuration groups. You can also use wildcards in configuration groups to configure ranges of data, without detailing each configuration line. Another way to use configuration groups is to create 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 affected 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 enables the system to build the inheritance path for each configuration group inside the database rather than in the process memory. This change can improve commit time performance. However, it can also increase the database size.

Creating and Activating a Candidate Configuration

You can enter software configuration statements using the CLI to create a candidate configuration that contains a hierarchy of statements. 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.

The CLI always maintains a copy of previously committed versions of the software configuration. If you need to return to a previous configuration, you can do this from within the CLI.

RELATED DOCUMENTATION

Junos OS Commit Model for Configurations

Format for Specifying IP Addresses, Network Masks, and Prefixes in Junos OS Configuration Statements

Many statements in the Junos OS configuration include an option to specify an IP address or route prefix. This option is represented as *`destination-prefix/prefix-length`*. Specifically, the route prefix, followed by a slash and the destination prefix length. For example, 192.168.1.10/32.

You enter all IP addresses in classless mode. You can enter the IP address with or without a prefix length, in standard dotted notation (for example, 1.2.3.4), or hexadecimal notation as a 32-bit number in network-byte order (for example, 0x01020304). If you omit any octets, they are assumed to be zero. Specify the prefix length as a decimal number from 1 through 32.

RELATED DOCUMENTATION

Format for Specifying Filenames and URLs in Junos OS CLI Commands | 66

Format for Specifying Filenames and URLs in Junos OS CLI Commands

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.

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

- *`filename`*—A file in the user's current directory on the local CompactFlash card (not applicable on the QFX Series). You can use wildcards to specify multiple source files or a single destination file. Wildcards are not supported in 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`*—A file on the local flash drive.
- *`filename`* or *`path/filename`*—File on the local hard drive.
- `a:`*`filename`* or `a:`*`path/filename`*—A file on the local removable media. 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** server. 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*/%2F*path/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***—A 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.

  NOTE: You cannot specify an HTTP(s) URL for a file as a destination, because HTTP(s) URLs are not writable. However you can specify HTTP(s) URL for a file as a source.

- **re0:/*path*/*filename*** or **re1:/*path*/*filename***—A file on a local Routing Engine.

- **sftp://*hostname*/*path*/*filename***—File on an SFTP server. You can also specify ***hostname*** as ***username@hostname***. The default path is the user's home directory. To specify an absolute path, you can use ***//path***.

RELATED DOCUMENTATION

Default Directories for Junos OS File Storage on the Network Device | 22

Format for Specifying IP Addresses, Network Masks, and Prefixes in Junos OS Configuration Statements | 65

Mapping the Name of the Router to IP Addresses

While using the Domain Name System (DNS) is an easier and more scalable way to resolve IP addresses from hostnames, you might want to manually map the hostname to a static IP address for the following reasons:

- You might not have a DNS entry for the device.

- You might not want the computer to contact the DNS server to resolve a particular IP address—you might use this particular IP address frequently, or it might be just for testing or development purposes.

To map a device's hostname to one or more IP addresses:

1. Include the `inet` statement at the `[edit system static-host-mapping` *`hostname`*`]` hierarchy level.

```
user@host# set system static-host-mapping hostname inet < ip-addresses >
```

2. Verify the configuration with the `show` command.

```
[edit system]
user@host# show
static-host-mapping {
    hostname {
        inet [ ip-addresses ];
    }
}
```

RELATED DOCUMENTATION

Configuring a Device’s Unique Identity for the Network

Configuring a DNS Name Server for Resolving Hostnames into Addresses

Configuring Automatic Mirroring of the CompactFlash Card on the Hard Disk

You can direct the device hard disk to automatically mirror the contents of the CompactFlash card. When you include the `mirror-flash-on-disk` statement, the hard disk maintains a synchronized mirror copy of the CompactFlash card contents. Data written to the CompactFlash card is simultaneously updated in the mirrored copy of the hard disk. If the CompactFlash card fails to read data, the hard disk automatically retrieves its mirrored copy of the CompactFlash card.

NOTE: We recommend that you disable flash-to-disk mirroring when you upgrade or downgrade the router.

You cannot issue the `request system snapshot` command while flash-to-disk mirroring is enabled.

To configure the mirroring of the CompactFlash card to the hard disk, include the `mirror-flash-on-disk` statement at the `[edit system]` hierarchy level:

```
[edit system]
mirror-flash-on-disk;
```

NOTE: After you have enabled or disabled the `mirror-flash-on-disk` statement, you must reboot the device for your changes to take effect. To reboot, issue the `request system reboot` command.

NOTE: This feature is not supported in Junos OS Release 20.1.

RELATED DOCUMENTATION

Configuring Junos OS for the First Time on a Device with a Single Routing Engine | 53

Using Junos OS to Specify the Number of Configurations Stored on the CompactFlash Card | 69

Using Junos OS to Specify the Number of Configurations Stored on the CompactFlash Card

By default, Junos OS saves the current configuration and three previous versions of the committed configuration on the CompactFlash card, with an additional 46 older versions stored on the hard drive. 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 **juniper.conf.3.gz**. These four files are located in the CompactFlash card in the directory **/config**.

In addition to saving the current configuration and the current operational version, you can also specify how many previous versions of the committed configurations you want stored on the CompactFlash card in the directory **/config**. The remaining previous versions of committed configurations (4 through 49) are stored in the directory **/var/db/config** on the hard disk. This is useful when you have very large configurations that might not fit on the CompactFlash card.

To specify how many previous versions of the committed configurations you want stored on the CompactFlash card, include the `max-configurations-on-flash` statement at the `[edit system]` hierarchy level:

```
[edit system]
max-configurations-on-flash number;
```

number is a value from 0 through 49.

RELATED DOCUMENTATION

Configuring Automatic Mirroring of the CompactFlash Card on the Hard Disk | 68

Back Up Configurations to an Archive Site

IN THIS SECTION

You can configure a device to transfer its configuration to an archive file periodically.

Configure the Transfer of the Active Configuration

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

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

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

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, SCP-style remote file specification, or SFTP URL. The URL type **file:** is also supported. When you specify the archive site, do not add a forward slash (/) to the end of the URL.

NOTE:

- The URL type **file:** is supported only for local files.
- When using the FTP option, specify a double forward slash (//) after the host:port. For example: **ftp://username@host<:port>//url-path**

file:/path/ is the minimal representation of a local file with no authority field and an absolute path that begins with a slash "/" as defined in RFC 8089.

file:///path is an example for a traditional file URI for a local file with an empty authority as defined in RFC 8089.

NOTE: When specifying a URL in a 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*"**

To configure the device to periodically transfer its 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.

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

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

If the network device reaches the archive server through a specific routing instance, configure the `routing-instance` statement at the `[edit system archival configuration]` hierarchy level, and specify the routing instance.

```
[edit system archival configuration]
    routing-instance routing-instance;
```

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: The time included in the destination filename is in Coordinated Universal Time (UTC).

See KB76300 for details on why SFTP transfers on commit fail if SSH-host keys are not configured.

Configuring Junos OS to Set Console and Auxiliary Port Properties

Most Juniper Networks devices have a console port and an auxiliary port for connecting terminals to the router or switch. The console port is enabled by default, and its speed is 9600 baud. The auxiliary port is disabled by default.

To configure the properties for the console and auxiliary ports, include the `ports` statement at the `[edit system]` hierarchy level:

```
[edit system]
ports {
    auxiliary {
        disable;
        insecure;
        type terminal-type;
    }
    console {
        authentication-order;
        disable;
        insecure;
        log-out-on-disconnect;
        type terminal-type;
    }
}
```

By default, the terminal type is set to `unknown`. To change the terminal type, include the `type` statement, specifying a *`terminal-type`* of `ansi`, `vt100`, `small-xterm`, or `xterm`. The first three terminal types set a screen size of 80 columns by 24 lines. The last type, `xterm`, sets the size to 80 columns by 65 rows.

By default, the console session is not logged out when the data carrier is lost on the console modem control lines. To change this default and log out the session automatically when the data carrier on the console port is lost, include the `log-out-on-disconnect` statement. You can use the `show system users` command to verify the console session is logged out.

By default, terminal connections to the console and auxiliary ports are secure. When you configure the console as insecure, root logins are not allowed to establish terminal connections. In addition, superusers and anyone with a user identifier (UID) of 0 are not allowed to establish terminal connections in multiuser mode when you configure the console as insecure. To disable root login connections to the console and auxiliary ports, include the `insecure` statement. This option can be used to prevent someone from attempting password recovery by booting into single-user mode, if they do not know the root password.

To disable console login, include the `disable` statement. By default, console login is enabled.

NOTE: For Common Criteria compliance, the console port must be disabled.

RELATED DOCUMENTATION

Methods for Configuring Junos OS | **45**

CHAPTER 5

Monitoring Junos Devices

IN THIS CHAPTER

Junos OS Tools for Monitoring

The primary method of monitoring and troubleshooting Junos OS, routing protocols, network connectivity, and the device hardware is to enter commands from the CLI. The CLI enables you to display information in the routing tables and routing protocol-specific data, and to check network connectivity using `ping` and `traceroute` commands.

The J-Web GUI is a Web-based alternative to using CLI commands to monitor, troubleshoot, and manage the device.

Junos OS includes SNMP software, which enables you to manage routers. The SNMP software consists of an SNMP master agent and a MIB II agent, and supports MIB II SNMP version 1 traps and version 2 notifications, SNMP version 1 `Get` and `GetNext` requests, and version 2 `GetBulk` requests.

The software also supports tracing and logging operations so that you can track events that occur—both normal device operations and error conditions—and track the packets that are generated by or pass through the device. Logging operations use a syslog-like mechanism to record system-wide, high-level operations, such as interfaces going up or down and users logging in to or out of the device. Tracing operations record more detailed messages about the operation of routing protocols, such as the various types of routing protocol packets sent and received, and routing policy actions.

RELATED DOCUMENTATION

Junos OS Features for Device Security | 38

Methods for Configuring Junos OS | 45

Tracing and Logging Junos OS Operations

Tracing and logging operations allow you to track events that occur in the device—both normal operations and error conditions—and to track the packets that are generated by or passed through the device. The results of tracing and logging operations are placed in files in the **/var/log** directory.

Remote Tracing

Junos OS provides an option to do remote tracing for specific processes, which greatly reduces use of device internal storage for tracing and is analogous to remote system logging. You configure remote tracing system-wide using the `tracing` statement at the `[edit system]` hierarchy level. By default, remote tracing is not configured. You can disable remote tracing for specific processes using the `no-remote-trace` statement at the `[edit` *`process-name`* `traceoptions]` hierarchy level. This feature does not alter local tracing functionality in any way, and logging files are stored on the device.

Junos OS supports remote tracing for the following processes:

- chassisd—Chassis-control process
- eventd—Event-processing process
- cosd—Class-of-service process
- spd—Adaptive-services process

To enable system-wide remote tracing, include the `destination-override syslog host` statement at the `[edit system tracing]` hierarchy level. This specifies the remote host running the system log process (syslogd), which collects the traces. Traces are written to file(s) on the remote host per the syslogd configuration in **/etc/syslog.conf**. By default remote tracing is *not* configured.

To override the system-wide remote tracing configuration for a particular process, include the `no-remote-trace` statement at the `[edit` *`process-name`* `traceoptions]` hierarchy. When `no-remote-trace` is enabled, the process does local tracing.

NOTE: When remote tracing is configured, traces will go to the remote host.

To collect traces, use the `local0` facility as the selector in **/etc/syslog.conf** on the remote host. To separate traces from various processes into different files, include the process name or trace-file name if it is specified at the [edit *process-name* traceoptions file] hierarchy level, in the Program field in **/etc/syslog.conf**. If your syslog server supports parsing hostname and program name, then you can separate traces from the various processes.

Logging Operations

Logging operations use a system logging mechanism similar to the UNIX syslogd utility to record system-wide, high-level operations, such as interfaces going up or down and users logging in to or out of the device. You configure these operations by using the `syslog` statement at the `[edit system]` hierarchy level, as described in *Junos OS System Log Overview*, and by using the `options` statement at the `[edit routing-options]` hierarchy level, as described in the Junos OS Routing Protocols Library for Routing Devices.

Tracing Operations

Tracing operations record more detailed messages about the operation of routing protocols, such as the various types of routing protocol packets sent and received, and routing policy actions. You configure tracing operations using the `traceoptions` statement. You can define tracing operations in different portions of the router configuration:

- Global tracing operations: Define tracing for all routing protocols. You define these tracing operations at the `[edit routing-options]` hierarchy level of the configuration.

- Protocol-specific tracing operations: Define tracing for a specific routing protocol. You define these tracing operations in the `[edit protocols]` hierarchy when configuring the individual routing protocol. Protocol-specific tracing operations override any equivalent operations that you specify in the global `traceoptions` statement. If there are no equivalent operations, they supplement the global tracing options. If you do not specify any protocol-specific tracing, the routing protocol inherits all the global tracing operations.

- Tracing operations within individual routing protocol entities: Some protocols allow you to define more granular tracing operations. For example, in Border Gateway Protocol (BGP), you can configure peer-specific tracing operations. These operations override any equivalent BGP-wide operations or, if there are no equivalents, supplement them. If you do not specify any peer-specific tracing operations, the peers inherit, first, all the BGP-wide tracing operations and, second, the global tracing operations.

- Interface tracing operations: Define tracing for individual router interfaces and for the interface process itself. You define these tracing operations at the `[edit interfaces]` hierarchy level of the configuration as described in the Junos OS Network Interfaces Library for Routing Devices.

RELATED DOCUMENTATION

Junos OS Network Interfaces Library for Routing Devices

Junos OS Routing Protocols Library for Routing Devices

Junos OS System Log Overview

Understanding Dropped Packets and Untransmitted Traffic Using show Commands

Starting with Junos OS Release 14.2, packets that need to be forwarded to the adjacent network element or a neighboring device along a routing path might be dropped by a device owing to several factors. Some of the causes for such a loss of traffic or a block in transmission of data packets include overloaded system conditions, profiles and policies that restrict the bandwidth or priority of traffic, network outages, or disruption with physical cable faults. You can use a number of show commands to determine and analyze the statistical counters and metrics related to any traffic loss and take an appropriate corrective measure. The fields displayed in the output of the show commands help in diagnosing and debugging network performance and traffic-handling efficiency problems.

The following `show` commands and associated fields applicable for dropped packets enable you to view and analyze some of the system parameters for errors or disruption in transmitted packets.

`show interfaces extensive`—Display input and output packet errors or drops. Following are some of the `show interfaces extensive` input counters and their definitions.

Following are definitions for some of the output counters for `show interfaces extensive`:

Following are definitions for some of the Queue counters for `show interfaces extensive` (both outbound and inbound). This includes CoS queue number and its associated user-configured forwarding class name, and is displayed on IQ2 interfaces.

Errors	Sum of the incoming frame terminates and FCS errors.
Drops	Number of packets dropped by the input queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism.
Framing errors	Number of packets received with an invalid frame checksum (FCS).
Runts	Number of frames received that are smaller than the runt threshold.
Policed discards	Number of frames that the incoming packet match code discarded because they were not recognized or not of interest. Usually, this field reports protocols that the Junos OS does not handle.
L3 incompletes	Number of incoming packets discarded because they failed Layer 3 (usually IPv4) sanity checks of the header. For example, a frame with less than 20 bytes of available IP header is discarded. L3 incomplete errors can be ignored by configuring the `ignore-l3-incompletes` statement.

L2 channel errors	Number of times the software did not find a valid logical interface for an incoming frame.
L2 mismatch timeouts	Number of malformed or short packets that caused the incoming packet handler to discard the frame as unreadable.
FIFO errors	Number of FIFO errors in the receive direction that are reported by the ASIC on the PIC. If this value is ever nonzero, the PIC is probably malfunctioning.
Resource errors	Error counter specific to the platform. For example on MX series routers, resource errors count PFE oversubscription drops.
Carrier transitions	Number of times the interface has gone from down to up. This number does not normally increment quickly, increasing only when the cable is unplugged, the far-end system is powered down and then up, or another problem occurs. If the number of carrier transitions increments quickly (perhaps once every 10 seconds), the cable, the far-end system, or the PIC or PIM is malfunctioning.
Errors	Sum of the outgoing frame terminates and FCS errors.
Drops	Number of packets dropped by the output queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism.
Collisions	Number of Ethernet collisions. The Gigabit Ethernet PIC supports only full-duplex operation, so for Gigabit Ethernet PICs, this number should always remain 0. If it is nonzero, there is a software bug.
Aged packets	Number of packets that remained in shared packet SDRAM so long that the system automatically purged them. The value in this field should never increment. If it does, it is most likely a software bug or possibly malfunctioning hardware.
FIFO errors	Number of FIFO errors in the send direction as reported by the ASIC on the PIC. If this value is ever nonzero, the PIC is probably malfunctioning.
HS link CRC errors	Number of errors on the high-speed links between the ASICs responsible for handling the router interfaces.
MTU errors	Number of packets whose size exceeded the MTU of the interface.
Resource errors	Error counter specific to the platform.
Queued packets	Number of queued packets.

Transmitted packets	Number of transmitted packets.
Dropped packets	Number of packets dropped by the ASIC's RED mechanism.

`show interfaces queue`—Display class-of-service (CoS) queue information for physical interfaces. Following are some of the `show interfaces queue` output fields and their definitions.

Queued packets	Number of queued packets.
Transmitted packets	Number of transmitted packets.
Dropped packets	Number of packets dropped by the ASIC's RED mechanism.
Tail-dropped packets	Number of packets dropped because of tail drop.
RL-dropped packets	Number of packets dropped due to rate limiting. For rate-limited interfaces hosted on MICs, MPCs, and Enhanced Queuing DPCs only, this statistic is not included in the queued traffic statistics.
RED-dropped packets	Number of packets dropped because of random early detection (RED). On M320 and M120 routers and most T Series routers, just the total number of dropped packets is displayed. For other M Series routers, as well as MX Series routers with enhanced DPCs, T Series routers with enhanced FPCs, and all J Series routers, the output classifies dropped packets into the following catetories: • `Low, non-TCP`—Number of low-loss priority non-TCP bytes dropped because of RED. • `Low, TCP`—Number of low-loss priority TCP packets dropped because of RED. • `High, non-TCP`—Number of high-loss priority non-TCP packets dropped because of RED. • `High, TCP`—Number of high-loss priority TCP packets dropped because of RED.

`show class-of-service fabric statistics summary`—Display class-of-service (CoS) switch fabric queue drop statistics. Following are the fabric queue statistics for dropped traffic:

Packets	Dropped packet count for high-priority and low-priority queues.
Bytes	Dropped byte count for high-priority and low-priority queues.
pps	Dropped packets-per-second count for high-priority and low-priority queues.

bps	Dropped bits-per-second count for high-priority and low-priority queues.

`show pfe statistics traffic fpc`—Display packet drops related to the entire FPC. Following are the FPC-level statistics for Packet Forwarding Engine hardware discards:

The following statistics are related to Packet Forwarding Engine local traffic for `show pfe statistics traffic fpc`:

Timeout	Number of packets discarded because of timeouts.
Truncated key	Number of packets discarded because of truncated keys.
Bits to test	Number of bits to test.
Data error	Number of packets discarded because of data errors.
Stack underflow	Number of packets discarded because of stack underflows.
Normal discard	Number of packets discarded because of discard routes. Packets are dropped silently without being further processed by the host. Normal discards are reported when packets match a firewall filter term that has an action of discard or when the final result of the route look-up is a next hop of discard.
Extended discard	The number of packets that are sent to the Routing Engine for further processing after being silently dropped. Extended discards are reported when packets match a firewall filter term that has an action of discard and an additional action that requires Routing Engine processing, such as log, count, sample, or syslog.
Invalid interface	Number of packets discarded because of invalid incoming interfaces.
Info cell drops	Number of information cell drops.
Fabric drops	Number of fabric drops.
Local packets input	Number of incoming packets from the local network.
Local packets output	Number of outgoing packets dispatched to a host in the local network.
Software input high drops	Number of incoming software packets of high-priority, dropped during transmission.
Software input medium drops	Number of incoming software packets of medium-priority, dropped during transmission.

Software input low drops	Number of incoming software packets of low-priority, dropped during transmission.
Software output drops	Number of outgoing software packets that were dropped during transmission.
Hardware input drops	Number of incoming hardware packets that were dropped during transmission.

The preceding commands represent only the main parameters that you can use to identify and monitor traffic drops or errors. Depending on your specific deployment scenario and network conditions, you might need to view the output of other relevant `show` commands to evaluate different factors that might be resulting in traffic transmission losses.

Log a User Out of the Device

Sometimes you may need to disconnect a user session if it does not terminate after a user logs out, or you may otherwise want to log a user out for some other reason.

To log a user out of all terminal sessions on a router, enter the following Junos OS CLI command:

```
user@host> request system logout username
```

```
user@host> show system users
10:07PM  up 13 days,  1:25, 2 users, load averages: 0.17, 0.05, 0.02
USER     TTY      FROM                              LOGIN@  IDLE WHAT
harry    p0       hpot-lt.cmpy.net                 10:07PM     - -cli (cl
lisa   p1       hpot-lt.cmpy.net                 10:06PM     - -cli (cl

user@host>  request system logout user harry
user@host>  show system users

10:07PM  up 13 days,  1:25, 1 user, load averages: 0.24, 0.06, 0.02
USER     TTY      FROM                              LOGIN@  IDLE WHAT
lisa   p1       hpot-lt.cmpy.net                 10:06PM     - -cli (cl
```

The sample output for the first `show system users` command shows there were two users on the router, harry and lisa. The `request system logout user` command was issued to log out user harry. Because there is no output to indicate that harry was logged out, the `show system users` command was issued again to verify that user harry was actually logged out of the router, while the user lisa remains logged in.

CHAPTER 6

Managing Junos OS Processes

IN THIS CHAPTER

Saving Core Files from Junos OS Processes

By default, when an internal Junos OS process generates a core file, the file and associated context information are saved for debugging purposes in a compressed tar file named ***process-name*.core.*core-number*.tgz** in the **/var/tmp/** and **/var/crash/** directories. For Junos OS Evolved, the output is saved in the **/var/core/** directory for Routing Engine core files and **/var/lib/ftp/in/** for FPC core files. The contextual information includes the configuration and system log message files.

To disable the saving of core files and associated context information, include the `no-saved-core-context` statement at the `[edit system]` hierarchy level:

```
[edit system]
no-saved-core-context;
```

To save the core files only, include the `saved-core-files` statement at the `[edit system]` hierarchy level and specify the number of files to save:

```
[edit system]
saved-core-files number;
```

`number` is the number of core files to save and can be a value from 1 through 10.

To save the core files along with the contextual information, include the `saved-core-context` statement at the `[edit system]` hierarchy level:

```
[edit system]
saved-core-context;
```

RELATED DOCUMENTATION

saved-core-context

saved-core-files

Viewing Core Files from Junos OS Processes

Viewing Core Files from Junos OS Processes

When an internal Junos OS process generates a core file, you can find the output at **/var/crash/** and **/var/tmp/**. For Junos OS Evolved, you can find the output core files at **/var/core/** for Routing Engine core files and **/var/lib/ftp/in/** for FPC core files. Using these directories provides a quick method of finding core issues across large networks.

Use the CLI command `show system core-dumps` to view core files.

```
root@host> show system core-dumps
-rw-------  1 root  wheel  268369920 Jun 18 17:59 /var/crash/vmcore.0
-rw-rw----  1 root  field    3371008 Jun 18 17:53 /var/tmp/rpd.core.0
-rw-r--r--  1 root  wheel   27775914 Jun 18 17:59 /var/crash/kernel.0
```

RELATED DOCUMENTATION

Saving Core Files from Junos OS Processes

Disabling Junos OS Processes

CAUTION: Never disable any of the software processes unless instructed to do so by a Customer Support engineer.

To disable a software process, specify the appropriate option in the `processes` statement at the `[edit system]` hierarchy level:

```
[edit system]
processes {
    process-name (enable | disable);
}
```

NOTE: The *`process-name`* variable is one of the valid process names. You can obtain a complete list of process names by using the CLI command completion feature.

RELATED DOCUMENTATION

processes

Configuring Failover to Backup Media If a Junos OS Process Fails | **84**

Viewing Core Files from Junos OS Processes

Configuring Failover to Backup Media If a Junos OS Process Fails

For network devices with redundant Routing Engines, you can configure the device to switch to backup media that contains a version of the system if a software process fails repeatedly, or to the other Routing Engine.

To configure automatic switchover to backup media if a software process fails, include the `failover` statement at the `[edit system processes` *`process-name`*`]` hierarchy level. If this statement is configured for a process, and that process fails four times within 30 seconds, the device reboots from either the alternative media or the other Routing Engine.:

```
[edit system processes]
process-name failover (alternate-media | other-routing-engine);
```

The value for *`process-name`* should be one of the valid process names.

RELATED DOCUMENTATION

Disabling Junos OS Processes | 84

Saving Core Files from Junos OS Processes | 82

Using Virtual Memory for Process Configuration Data

Configuration data for each process in Junos OS is stored in memory that is mapped within the address space of each process, requiring a fixed maximum space to be reserved in each process. This scheme works well until a process is managing many functions at commit time and negatively impacts the commit time, or simply needs more memory than the default allotment. For example, the `rpd` process might be managing many routes and require more space to store important information about the routes.

In circumstances that require more than the maximum memory-mapped size, you can use `virtual-memory-mapping` at the `[edit system configuration-database]` hierarchy level to make more memory available for the configuration database per process.

You can configure a portion of virtual memory at a fixed size for the initial portion of the configuration database, and you can specify an amount to be used for page-pooling. Page-pooling uses a small amount of memory to bring database pages into memory as needed, rather than mapping the entire configuration database into the virtual memory space for the process.

3

PART

Configuration Statements and Operational Commands

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Read this guide to learn about the syntax and options that make up the statements and commands. Also understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands