Juniper NETWORKS | Engineering Simplicity

Junos® OS Evolved

Common Criteria Evaluated Configuration Guide for PTX10001-36MR Device

Published
2025-05-21

RELEASE
23.4R1

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

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

Junos® OS Evolved Common Criteria Evaluated Configuration Guide for PTX10001-36MR Device
23.4R1

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About This Guide

Use this guide to configure and evaluate PTX10001-36MR device for Common Criteria (CC) compliance. Common Criteria for information technology is an international agreement signed by several countries that permit the evaluation of security products against a common set of standards.

RELATED DOCUMENTATION

Common Criteria and FIPS Certifications

1

CHAPTER

Overview

IN THIS CHAPTER

Common Criteria Evaluated Configuration Overview

IN THIS SECTION

This document describes the steps required to duplicate the configuration of the device running Junos OS Evolved when the device is evaluated. This is referred to as the evaluated configuration. The following list describes the standards to which the device has been evaluated:

- CPP_ND_V2.2E—https://www.niap-ccevs.org/MMO/PP/CPP_ND_V2.2E.pdf

Common Criteria Overview

Common Criteria for information technology is an international agreement signed by several countries that permits the evaluation of security products against a common set of standards. In the Common Criteria Recognition Arrangement (CCRA) at https://www.commoncriteriaportal.org/ccra/, the participants agree to mutually recognize evaluations of products performed in other countries. All evaluations are performed using a common methodology for information technology security evaluation.

For more information on Common Criteria, see https://www.commoncriteriaportal.org/.

Target of Evaluation (TOE) is a device or system subjected to evaluation based on Collaborative Protection Profile (cPP).

Supported Platforms

For the features described in this document, the following platforms are supported to qualify NDcPPv2.2e:

- PTX10001-36MR (https://www.juniper.net/us/en/products/routers/ptx-series/ptx10001-36mr-packet-transport-router.html)

RELATED DOCUMENTATION

Get Started With Your Device | 7

Junos OS Evolved in FIPS Mode Overview

IN THIS SECTION

Federal Information Processing Standards (FIPS) 140-3 defines security four levels for hardware, firmware, and software that perform cryptographic functions. The Juniper Networks PTX devices running the Junos Evolved operating system (Junos OS Evolved) in *FIPS mode* comply with the FIPS 140-3 Level 1 standard within a modifiable operational environment.

Operating PTX devices in a FIPS 140-3 Level 1 environment requires enabling and configuring FIPS mode on the devices from the Junos OS Evolved command-line interface (CLI).

The security-administrator enables FIPS mode and sets up keys and passwords for the system and other FIPS users.

Supported Modules

The following cryptographic modules are supported on the devices:

- Juniper Linux Kernel Cryptographic Module version 2.0
- Juniper OpenSSL Cryptographic Module version 3.0

These modules enter FIPS mode only after self-test and integrity check is passed, and FIPS mode level is set.

About the Cryptographic Boundary on Your Device

FIPS 140-3 compliance requires a defined cryptographic boundary around each cryptographic module on a device. Junos OS Evolved in FIPS mode defines two cryptographic boundaries: Kernel and OpenSSL. When operating in an approved mode of operation, these programs, or programs accessing elements inside of these cryptographic boundaries, should ensure to only use approved configurations as specified in this guide to ensure the proper FIPS 140-3 compliance. No CSPs will be allowed outside of the cryptographic boundary and zeroization of any CSPs will occur when the software comprising the cryptographic boundary is terminated.

How FIPS Mode Differs from Non-FIPS Mode

Junos OS Evolved in FIPS mode differs in the following ways from Junos OS Evolved in non-FIPS mode:

- Self-tests of all cryptographic algorithms are performed at the program startup. Self-test failure means that the cryptographic module is terminated.
- Self-tests of random number and key generation are performed continuously.
- Weak or unencrypted management connections must not be configured.

Validated Version of Junos OS Evolved in FIPS Mode

To determine whether a Junos OS Evolved release is FIPS 140-3 Level 1 certified, see the compliance page on the Juniper Networks Web site (https://apps.juniper.net/compliance/fips.html).

RELATED DOCUMENTATION

Identify Secure Product Delivery | 8

Overview of FIPS Terminology and Supported Cryptographic Algorithms

IN THIS SECTION

Use the definitions of FIPS terms, and supported algorithms to help you understand Junos OS Evolved in FIPS mode.

Terminology

Common Criteria

Common Criteria for information technology is an international agreement signed by several countries that permits the evaluation of security products against a common set of standards.

Critical security parameter (CSP)

Security-related information—for example, secret and private cryptographic keys and authentication data such as passwords and personal identification numbers (PINs)—whose disclosure or modification can compromise the security of a cryptographic module or the information it protects. For details, see "Overview of Roles and Services for Junos OS Evolved in FIPS" on page 11.

Cryptographic module

The set of hardware, software, and firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

FIPS

Federal Information Processing Standards. FIPS 140-3 specifies requirements for security and cryptographic modules. Junos OS Evolved in FIPS mode complies with FIPS 140-3 Level 1.

KATs

Known answer tests. System self-tests that validate the output of cryptographic algorithms approved for FIPS and test the integrity of Junos OS Evolved modules.

Security Administrator

Person with appropriate permissions who is responsible for securely enabling, configuring, monitoring, and maintaining Junos OS Evolved in FIPS mode of operation on a device. For details, see "Overview of the Operational Environment for Junos OS Evolved in FIPS Mode" on page 13.

SSH

A protocol that uses strong authentication and encryption for remote access across a nonsecure network. SSH provides remote login, remote program execution, file copy, and other functions. It is intended as a secure replacement for **rlogin**, **rsh**, and **rcp** in a UNIX environment. To secure the information sent over administrative connections, use SSHv2 for CLI configuration. In Junos OS Evolved, SSHv2 is enabled by default, and SSHv1, which is not considered secure, is disabled.

Supported Cryptographic Algorithms

BEST PRACTICE: For FIPS 140-3 compliance, use only FIPS-approved cryptographic algorithms in Junos OS Evolved in FIPS mode.

The following cryptographic algorithms are supported in FIPS mode. Symmetric methods use the same key for encryption and decryption, while asymmetric methods use different keys for encryption and decryption.

AES

The Advanced Encryption Standard (AES), defined in FIPS PUB 197. The AES algorithm uses keys of 128 or 256 bits to encrypt and decrypt data in blocks of 128 or 256 bits.

Diffie-Hellman

A method of key exchange across a nonsecure environment (such as the Internet). The Diffie-Hellman algorithm negotiates a session key without sending the key itself across the network by allowing each party to pick a partial key independently and send part of that key to the other. Each side then calculates a common key value. This is a symmetrical method—keys are typically used only for a short time, discarded, and regenerated.

ECDH

Elliptic Curve Diffie-Hellman. A variant of the Diffie-Hellman key exchange algorithm that uses cryptography based on the algebraic structure of elliptic curves over finite fields. ECDH allows two parties, each having an elliptic curve public-private key pair, to establish a shared secret over an insecure channel. The shared secret can be used either as a key or to derive another key for encrypting subsequent communications using a symmetric key cipher.

ECDSA

Elliptic Curve Digital Signature Algorithm. A variant of the Digital Signature Algorithm (DSA) that uses cryptography based on the algebraic structure of elliptic curves over finite fields. The bit size of the elliptic curve determines the difficulty of decrypting the key. The public key believed to be needed for ECDSA is about twice the size of the security level, in bits. ECDSA using the P-256, P-384, and P-521 curves can be configured under OpenSSH.

HMAC

Defined as “Keyed-Hashing for Message Authentication” in RFC 2104, HMAC combines hashing algorithms with cryptographic keys for message authentication. For Junos OS Evolved in FIPS mode, HMAC uses the iterated cryptographic hash functions SHA-256, SHA-384, and SHA-512 along with a secret key.

RSA

Algorithm for public key cryptography that is based on the presumed difficulty of factoring large integers of up to 8192 bits. The RSA algorithm involves five steps: key generation, sign, verify signature, encryption, and decryption. FIPS provides the use of SSHv2 with RSA, but should use keys of 2048-bits or 3072-bits in length and no smaller. The RSA algorithm is used in the validation of Juniper Networks signed binaries and is also available and used with the `ssh` command.

SHA-256, SHA-384, and SHA-512

Secure hash algorithms (SHA) belonging to the SHA-2 standard defined in FIPS PUB 180-2. Developed by NIST, SHA-256 produces a 256-bit hash digest, SHA-384 produces a 384-bit hash digest, and SHA-512 produces a 512-bit hash digest.

Get Started With Your Device

IN THIS SECTION

The following topics help you to validate, secure the product delivery, and perform the initial set up of PTX devices.

Identify Secure Product Delivery

There are several mechanisms provided in the delivery process to ensure that a customer receives a product that has not been tampered with. The customer should perform the following checks upon receipt of a device to verify the integrity of the platform.

- Shipping label—Ensure that the shipping label correctly identifies the correct customer name and address as well as the device.
- Outside packaging—Inspect the outside shipping box and tape. Ensure that the shipping tape and the box are intact and without any damage.
- Inside packaging—Inspect the bag and seal. Ensure that the bag is not opened. Ensure that the seal remains intact.

If you identify a problem during the inspection, you should contact the supplier. Provide the order number, tracking number, and a description of the identified problem to the supplier.

Additionally, there are several checks that can be performed to ensure that you have received a box sent by Juniper Networks and not a different company masquerading as Juniper Networks. You should perform the following checks upon receipt of a device to verify the authenticity of the device:

- Verify that the device was ordered using a purchase order. Juniper Networks devices are never shipped without a purchase order.
- When a device is shipped, a shipment notification is sent to the e-mail address provided by you when the order is taken. Verify that this e-mail notification was received. Verify that the e-mail contains the following information:
  - Purchase order number
  - Juniper Networks order number used to track the shipment
  - Carrier tracking number used to track the shipment
  - List of items shipped including serial numbers
  - Address and contacts of both the supplier and the customer
- Verify that the shipment was initiated by Juniper Networks. To verify that a shipment was initiated by Juniper Networks, you should perform the following tasks:
  - Compare the carrier tracking number of the Juniper Networks order number listed in the Juniper Networks shipping notification with the tracking number on the package received.
  - Log on to the Juniper Networks online customer support portal at https://support.juniper.net/support/ to view the order status. Compare the carrier tracking number or the Juniper Networks

order number listed in the Juniper Networks shipment notification with the tracking number on the package received.

Management Interfaces Overview

The following management interfaces can be used in the evaluated configuration:

- Local Management Interfaces—The RJ-45 console port on the device is configured as RS-232 data terminal equipment (DTE). You can use the command-line interface (CLI) over this port to configure the device from a terminal.

- Remote Management Protocols—You can manage the device remotely over any Ethernet interface. It is recommended to use SSHv2 to perform remote management either via the CLI, or using NETCONF.

2

CHAPTER

Configure Roles and Authentication Methods

IN THIS CHAPTER

Overview of Roles and Services for Junos OS Evolved in FIPS

IN THIS SECTION

Junos OS Evolved allows a wide range of capabilities for identity-based users. In FIPS mode, the various range of capabilities are typically defined to assign each identity-based user one of two user roles: Security Administrator and FIPS user. These roles are defined in terms of Junos OS Evolved user capabilities. The Security Administrator may also choose to create additional roles consistent with the operational guidelines of their organization. Such additional roles might include specific permissions to various Junos OS Evolved commands as they are useful for rules such as: Security Officer, Audit Officer, and any other administrative roles as may be prudent to delegate. The creation of other administrative roles is outside the scope of this guide.

Any role that is intended to interact with the FIPS modules should fall into the class of either a Security Administrator role or FIPS user role, or a subset of the Security Administrator role as determined by local policies of the organization using the device.

Security Administrator performs all FIPS-mode-related configuration tasks and issue all statements and commands for Junos OS Evolved in FIPS mode. Security Administrator and FIPS user configurations must follow the guidelines for Junos OS Evolved in FIPS mode.

Security Administrator Role and Responsibilities

The Security Administrator is the person responsible for enabling, configuring, monitoring, and maintaining Junos OS Evolved in FIPS mode on a router. The Security Administrator securely installs Junos OS Evolved on the device, enables FIPS mode, establishes keys and passwords for other users and software modules, and initializes the device before network connection.

The permissions that distinguish the Security Administrator from other FIPS users are `secret`, `security`, `maintenance`, and `control`. For FIPS compliance, assign the Security Administrator to a login class that contains all of these permissions. A user with the Junos OS Evolved maintenance permission can read sensitive files containing private information on the configuration of the device.

> **NOTE**: There is no relationship between the FIPS 140-3 maintenance mode and the similarly named Junos OS Evolved maintenance permission.

Among the tasks related to Junos OS Evolved in FIPS mode, the Security Administrator is expected to:

- Set the initial root password. The length of the password should be atleast 10 characters.
- Examine log and audit files for events of interest.

FIPS User Role and Responsibilities

All FIPS users who are part of operator/read-only/superuser or super-user class, including the Security Administrator, can view the configuration. Only the user assigned as the Security Administrator can modify the configuration.

FIPS user can view status output but cannot reboot or zeroize the device.

What Is Expected of All FIPS Users

All FIPS users, including the Security Administrator, must observe security guidelines at all times.

All FIPS users must:

- Keep all passwords confidential.
- Store routers and documentation in a secure area.
- Deploy routers or switches in secure areas.
- Check audit files periodically.
- Conform to all other FIPS 140-3 security rules.
- Follow these guidelines:
    - Users are trusted.
    - Users abide by all security guidelines.

- Users do not deliberately compromise security.
- Users behave responsibly at all times.

Overview of the Operational Environment for Junos OS Evolved in FIPS Mode

IN THIS SECTION

A Juniper Networks router running the Juniper Networks Junos operating system (Junos OS) Evolved in FIPS mode provides an enhanced software operational environment that is different from the environment of a device in non-FIPS mode.

Software Environment for Junos OS Evolved in FIPS Mode

The Junos OS Evolved in FIPS mode software environment is established after the Security Administrator successfully enables FIPS mode on a device. This Junos OS Evolved Release image that includes FIPS mode is available on the Juniper Networks website and can be configured on a functioning router.

The minimum length of the passwords must be 10 characters and require the use of at least three of the five defined character sets (uppercase and lowercase letters, digits, punctuation marks, and keyboard characters, such as % and &, not included in the other four categories). All passwords and keys used to authenticate peers must be at least 10 characters in length, and in some cases the length must match the digest size.

NOTE: Do not attach the router to a network until the Security Administrator completes configuration from the local console connection.

For strict compliance, do not examine core and crash dump information on the local console in Junos OS Evolved in FIPS mode because some CSPs might be shown in plain text.

Critical Security Parameters

Critical security parameters (CSPs) are security-related information such as cryptographic keys and passwords that can compromise the security of the cryptographic module or the security of the information protected by the module if they are disclosed or modified.

> **BEST PRACTICE**: For FIPS compliance, configure the device over SSH connections because they are encrypted connections.

Local passwords are hashed with the SHA256 or SHA512 algorithm. Junos OS Evolved in FIPS mode cannot boot into single-user mode without the correct root password.

Overview of Password Specifications and Guidelines for Junos OS in FIPS Mode

All passwords established for users by the Security Administrator must conform to the following Junos OS in FIPS mode requirements. Attempts to configure passwords that do not conform to the following specifications result in an error.

- *Length*: Passwords must contain at least 10 characters.
- *Character set requirements*: Passwords must contain at least three of the following five defined character sets:
  - Uppercase letters
  - Lowercase letters
  - Digits
  - Punctuation marks
  - Keyboard characters not included in the other four sets—such as the percent sign (%) and the ampersand (&)

- *Authentication requirements*: All passwords and keys used to authenticate peers must contain at least 10 characters, and in some cases the number of characters must match the digest size.
- *Password encryption*: To change the default encryption method (SHA512) include the `format` statement at the `[edit system login password]` hierarchy level.

Guidelines for strong passwords: Strong, reusable passwords can be based on letters from a favorite phrase or word and then concatenated with other unrelated words, along with added digits and punctuation. In general, a strong password is:

- Easy to remember so that users are not tempted to write it down.
- Made up of mixed alphanumeric characters and punctuation. For FIPS compliance include at least one change of case, one or more digits, and one or more punctuation marks.
- Changed periodically.
- Not divulged to anyone.

Characteristics of weak passwords: Do not use the following weak passwords:

- Words that might be found in or exist as a permuted form in a system files such as `/etc/passwd`.
- The hostname of the system (always a first guess).
- Any word or phrase that appears in a dictionary or other well-known source, including dictionaries and thesauruses in languages other than English; works by classical or popular writers; or common words and phrases from sports, sayings, movies or television shows.
- Permutations on any of the above—for example, a dictionary word with letters replaced with digits (`r00t`) or with digits added to the end.
- Any machine-generated password. Algorithms reduce the search space of password-guessing programs and so must not be used.

Download Software Package from Juniper Networks

Before you begin to download the software, ensure that you have a Juniper Networks Web account and a valid support contract. To obtain an account, complete the registration form at the Juniper Networks website: https://userregistration.juniper.net/.

You can download the the required Junos OS Evolved software package from the Juniper Networks website:

To download software package from Juniper Networks:

1. Using a Web browser, navigate to https://support.juniper.net/support/downloads/.
2. Enter the product name that you want to download.
3. Click the package link in the Downloads column.
4. A login screen appears. Log in to the Juniper Networks authentication system using the username (generally your e-mail address) and password supplied by Juniper Networks representatives.
5. Download the software.

RELATED DOCUMENTATION

Installation and Upgrade Guide

Install Software Package on Your Device

Follow the below procedure to install a software package:

- (Optional) Back up the current software configuration to a second storage option. See the Junos® OS Evolved Software Installation and Upgrade Guide for instructions on performing this task.

- Ensure that you have downloaded the software package, see "Download Software Package from Juniper Networks" on page 15.

Junos OS Evolved is delivered in signed package that contain digital signatures to ensure the Juniper Networks software is running. When installing the software package, Junos OS Evolved validates the signatures and the public key certificates used to digitally sign the software package. If the signature or certificate is found to be invalid (for example, when the certificate validity period has expired or cannot be verified against the root CA stored in the Junos OS Evolved internal store), the installation process fails.

To install software upgrades on a device with a single Routing Engine:

1. You must connect to the console port on the device from your management device, and log in to the Junos OS Evolved CLI.
2. Enter the following command to install the Junos OS Evolved package.

   ```
   user@device> request system software add <package>
   ```

   You can specify any of the following path:

   - For a software package in a local directory on the device, use /var/tmp/package.iso.

- For a software package on a remote server, use one of the following paths:
    - ftp://hostname/pathname/package.iso
    - http://hostname/pathname/package.iso

  The `package.iso` refers to the respective Junos OS Evolved image name.

3. Reboot the device to load the installation:

   ```
   user@device> request system reboot
   ```

4. After the reboot has completed, log in and use the `show version` command to verify that the new version of the software is successfully installed.

   ```
   root@host> show version
   Hostname: hostname
   Model: ptx10001-36mr
   Junos: 23.4R1.10-EVO
   Yocto: 3.0.2
   Linux Kernel: 5.2.60-yocto-standard-ge8e43b6
   JUNOS-EVO OS 64-bit [junos-evo-install-ptx-fixed-x86-64-23.4R1.10-EVO]
   ```

Overview of Zeroization to Clear System Data for FIPS Mode

Zeroization completely erases all configuration information on the Routing Engines, including all plain-text passwords, secrets, and private keys for SSH, local encryption, local authentication and IPsec.

⚠ **CAUTION**: Perform system zeroization with care. After the zeroization process is complete, no data is left on the Routing Engine. The device is returned to the factory default state, without any configured users or configuration files.

Zeroization can be time-consuming. Although all configurations are removed in a few seconds, the zeroization process goes on to overwrite all media, which can take considerable time depending on the size of the media.

Zeroize the System

To zeroize your device, follow the below procedure:

1. Login to the device as Security Administrator and from CLI, enter

```
security-administrator@host> request system zeroize
Zeroization : Erase all data, including configuration and log files ? [yes,no]
(no) yes
re0:
```

2. To initiate the zeroization process, type **yes** at the prompt:

```
Erase all data, including configuration and log files? [yes, no] (no)
yes
re0:
--------------------------------------------------------------------------
warning: zeroizing re0
...
...
```

The entire operation can take considerable time depending on the size of the media, but all critical security parameters (CSPs) are removed within a few seconds. The physical environment must remain secure until the zeroization process is complete.

Enable FIPS Mode

As Security Administrator, you must establish a root password conforming to the FIPS password requirements in "Overview of Roles and Services for Junos OS Evolved in FIPS" on page 11. When you enable FIPS mode in Junos OS Evolved on the device, you cannot configure passwords unless they meet this standard.

Local passwords are encrypted with the secure hash algorithm SHA256 or SHA512.

To enable FIPS mode in Junos OS Evolved on the device:

1. Login to the device using `root`.

```
host login: root
Password:
Last login: Tue May 28 15:44:39 IST 2024 from 10.32.196.40 on pts/0
--- JUNOS 23.4R1.10-EVO Linux (none) 5.2.60-yocto-standard-ge8e43b6 #1 SMP PREEMPT Sun Dec 17
00:14:17 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
[vrf:none] root@host:~#
[vrf:none] root@host:~# cli
root@host>
```

2. Configure root authentication with password at least 10 characters or more.

```
root@host> edit
Entering configuration mode
[edit]
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
[edit]
root@host# commit
commit complete
```

3. Load configuration onto device and commit new configuration. Configure Security Administrator and login with Security Administrator credentials.
4. Configure FIPS by setting `set system fips level` *1* and `commit`.

```
[edit]
root@hostname# set system fips level 1
```

5. After deleting and reconfiguring CSPs, commit will go through and device needs reboot to enter FIPS mode.

```
[edit]
security-administrator@host# commit
[edit]
system reboot is required to transition to FIPS level 1
commit complete
```

6. Reboot the device:

```
[edit]
security-administrator@host# run request system reboot
```

7. After rebooting the device, FIPS self-tests will run and device enters FIPS mode.

```
security-administrator@host:fips>
```

Configure Security Administrator and FIPS User Identification and Access

IN THIS SECTION

Security Administrator and FIPS users perform all configuration tasks for Junos OS in FIPS mode and issue all Junos OS in FIPS mode statements and commands. Security Administrator and FIPS user configurations must follow Junos OS in FIPS mode guidelines.

Configure Security Administrator Access

Junos OS in FIPS mode offers a finer granularity of user permissions than those mandated by FIPS 140-3.

For FIPS 140-3 compliance, any FIPS user with the `secret`, `security`, `maintenance`, and `control` permission bits set is a Security Administrator. In most cases the `super-user` class suffices for the Security Administrator.

To configure login access for a Security Administrator:

1. Log in to the device with the root password if you have not already done so, and enter configuration mode:

   ```
   root@host# edit
   Entering configuration mode
   [edit]
   root@host#
   ```

2. Name the user `security-administrator` and assign the Security Administrator a user ID (for example, `6400`, which must be a unique number associated with the login account in the range of 100 through 64000) and a class (for example, `super-user`). When you assign the class, you assign the permissions—for example, `secret`, `security`, `maintenance`, and `control`.

   ```
   [edit]
   root@host# set system login user username uid value class class-name
   ```

   For example:

   ```
   [edit]
   root@host# set system login user security-administrator uid 6400 class super-user
   ```

3. Following the guidelines in "Overview of Password Specifications and Guidelines for Junos OS in FIPS Mode" on page 14, assign the Security Administrator a plain-text password for login authentication. Set the password by typing a password after the prompts `New password` and `Retype new password`.

   ```
   [edit]
   root@host#  set system login user username class class-name authentication (plain-test-
   password | encrypted-password)
   ```

   For example:

   ```
   [edit]
   root@host#  set system login user security-administrator class super-user authentication
   plain-text-password
   ```

4. Optionally, display the configuration:

```
[edit]
root@host#edit system
[edit system]
root@host#show
login {
    user security-administrator {
        uid 6400;
        authentication {
            encrypted-password "<cipher-text>"; ## SECRET-DATA
        }
        class super-user;
    }
}
```

5. If you are finished configuring the device, commit the configuration and exit:

```
[edit]
root@host# commit
commit complete
root@host# exit
```

Configure FIPS User Login Access

A `fips-user` is defined as any FIPS user that does not have the `secret`, `security`, `maintenance`, and `control` permission bits set.

As the Security Administrator you set up FIPS users. FIPS users cannot be granted permissions normally reserved for the Security Administrator—for example, permission to zeroize the system.

To configure login access for a FIPS user:

1. Log in to the device with your Security Administrator password if you have not already done so, and enter configuration mode:

```
security-administrator@host:fips> edit
Entering configuration mode
```

```
[edit]
security-administrator@host:fips#
```

2. Give the user, a username, and assign the user a user ID (for example, `6401`, which must be a unique number in the range of 1 through 64000) and a class. When you assign the class, you assign the permissions—for example, `clear`, `network`, `resetview`, and `view-configuration`.

```
[edit]
security-administrator@host:fips# set system login user username uid value class class-name
```

For example:

```
[edit]
security-administrator@host:fips# set system login user fips-user1 uid 6401 class read-only
```

3. Following the guidelines in "Overview of Password Specifications and Guidelines for Junos OS in FIPS Mode" on page 14, assign the FIPS user a plain-text password for login authentication. Set the password by typing a password after the prompts `New password` and `Retype new password`.

```
[edit]
security-administrator@host:fips# set system login user username class class-name
authentication (plain-text-password | encrypted-password)
```

For example:

```
[edit]
security-administrator@host:fips# set system login user fips-user1 class read-only
authentication plain-text-password
```

4. Optionally, display the configuration:

```
[edit]
security-administrator@host:fips# edit system
[edit system]
security-administrator@host:fips# show
login {
    user fips-user1 {
        uid 6401;
        authentication {
            encrypted-password "<cipher-text>"; ## SECRET-DATA
        }
        class read-only;
    }
}
```

5. If you are finished configuring the device, commit the configuration and exit:

```
[edit]
security-administrator@host:fips# commit
security-administrator@host:fips#  exit
```

RELATED DOCUMENTATION

Overview of Roles and Services for Junos OS Evolved in FIPS | 11

3

CHAPTER

Juniper Linux Kernel Cryptographic Module

IN THIS CHAPTER

Juniper Linux Kernel Cryptographic Module Overview

IN THIS SECTION

Junos OS Evolved Linux Kernel Cryptographic API Module version 2.0 provides cryptographic services to kernel applications through C language Application Program Interface (API) and to applications that run in a user space through an AF_ALG socket type interface. The module utilizes instructions from processor to optimize and increase the performance of cryptographic algorithms.

Cryptographic Boundary

The Cryptographic Logical Boundary for kernel consists of all kernel objects and integrity check files used to perform integrity tests.

Supported Cryptographic Algorithms

You must use FIPS approved cryptographic algorithms in FIPS mode. If you use non-approved algorithms the system would not be in a FIPS certified state.

Performing Self-Test

IN THIS SECTION

The cryptographic module enforces security rules to ensure that the Juniper Networks Junos OS Evolved in FIPS mode meets the security requirements of FIPS 140-3 Level 1. To validate the output of cryptographic algorithms approved for FIPS and test the integrity of some system modules, the device performs series of known answer test (KAT) self-tests.

The KAT self-tests are performed automatically at startup.

If the KATs are completed successfully, the dmesg log is updated to display the tests that are executed. You can view the logs by executing `journalctl | grep self-test` on the device shell.

```
[vrf:none] root@evoptx10k-b:~# journalctl | grep self-test
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for rsa-generic (rsa) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for cipher_null-generic (cipher_null) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for ecb-cipher_null (ecb(cipher_null)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha1-generic (sha1) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha256-generic (sha256) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha224-generic (sha224) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha512-generic (sha512) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha384-generic (sha384) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha3-224-generic (sha3-224) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha3-256-generic (sha3-256) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha3-384-generic (sha3-384) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha3-512-generic (sha3-512) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for aes-generic (aes) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for crc32c-generic (crc32c) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for crct10dif-generic (crct10dif) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for ctr(aes-generic) (ctr(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_ctr_aes128 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_ctr_aes192 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_ctr_aes256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_sha1 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_sha512 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_sha256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_hmac_sha1 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_hmac_sha512 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for hmac(sha256-generic) (hmac(sha256))
passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_pr_hmac_sha256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_ctr_aes128 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_ctr_aes192 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_ctr_aes256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_sha1 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_sha512 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_sha256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_hmac_sha1 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_hmac_sha512 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for drbg_nopr_hmac_sha256 (stdrng) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for aes-asm (aes) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for crc32c-intel (crc32c) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha1-ssse3 (sha1) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha1-avx (sha1) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha1-avx2 (sha1) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha256-ssse3 (sha256) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha224-ssse3 (sha224) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha256-avx (sha256) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha224-avx (sha224) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha256-avx2 (sha256) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha224-avx2 (sha224) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha512-ssse3 (sha512) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha384-ssse3 (sha384) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha512-avx (sha512) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha384-avx (sha384) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha512-avx2 (sha512) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for sha384-avx2 (sha384) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for crct10dif-pclmul (crct10dif) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for jitterentropy_rng (jitterentropy_rng)
passed
May 21 10:27:34 evoptx10k-b kernel: lrng_selftest: LRNG self-tests passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for aes-aesni (aes) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for ecb-aes-aesni (ecb(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for cbc-aes-aesni (cbc(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for ctr-aes-aesni (ctr(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for xts-aes-aesni (xts(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for rfc4106-gcm-aesni (rfc4106(gcm(aes)))
passed
```

```
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for generic-gcm-aesni (gcm(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for pkcs1pad(rsa-generic,sha256)
(pkcs1pad(rsa,sha256)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for hmac(sha1-avx2) (hmac(sha1)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for cbc(aes-aesni) (cbc(aes)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for hmac(sha512-avx2) (hmac(sha512)) passed
May 21 10:27:34 evoptx10k-b kernel: alg: self-tests for cmac(aes-aesni) (cmac(aes)) passed
[vrf:none] root@evoptx10k-b:~#
```

Self-test failure results in a FIPS error state and the device automatically reboots after encountering a FIPS error state.

Integrity Validation

To validate the integrity, set the FIPS level and reboot the device and verify the integrity logs.

If there is an integrity failure, the modules stops and generates a FIPS error state.

You can check the logs for a successful integrity. For example:

```
[vrf:none] root@re0:~# journalctl | grep "FIPS integrity"

Sep 23 22:10:18 re0 unknown: FIPS integrity check passed for bzImage-re-64b.bin
Sep 23 22:10:18 re0 unknown: FIPS integrity check passed for initrd_Yocto_2.2_x86_64.fs
```

4

CHAPTER

Juniper OpenSSL Cryptographic Module

IN THIS CHAPTER

Juniper OpenSSL Cryptographic Module Overview

IN THIS SECTION

Junos OS Evolved OpenSSL Cryptographic Module version 3.0 provides cryptographic primitive APIs for Junos OS Evolved user space. This module provides cryptographic services to applications that runs in the user space of Junos OS Evolved through C language Application Program Interface (API).

Cryptographic Boundary

The Cryptographic Logical Boundary for OpenSSL consists of all shared libraries and integrity check files used to perform integrity tests.

Supported Cryptographic Algorithms

You must use FIPS approved cryptographic algorithms in FIPS mode. Table 1 on page 31 lists the approved cryptographic algorithms that you can use in FIPS mode.

Table 1: Cryptographic Algorithms

Cipher	Shared Secret/Diffie Hellman Key Generation	MAC	Keys	KDF
AES-128-CTR	ecdh-sha2-nistp256 (NIST P-256 ECDH)	hmac-sha2-256	RSA (2048 bit key sizes)	SSHKDF
AES-192-CTR	ecdh-sha2-nistp384 (NIST P-384 ECDH)	hmac-sha2-512	ECDSA P-256	

Table 1: Cryptographic Algorithms *(Continued)*

Cipher	Shared Secret/Diffie Hellman Key Generation	MAC	Keys	KDF
AES-256-CTR	ecdh-sha2-nistp521 (NIST P-521 ECDH)		ECDSA P-384	
	dh-group14-sha1		ECDSA P-521	

.

Performing Self-Test

The cryptographic module enforces security rules to ensure that the Juniper Networks Junos OS Evolved in FIPS mode meets the security requirements of FIPS 140-3 Level 1. To validate the output of cryptographic algorithms approved for FIPS and test the integrity of some system modules, the device performs series of known answer test (KAT) self-tests.

Self-tests are executed in the background with no output unless there is a failure. If there is a failure, the module will core dump.

The self-tests details are as follows:

```
SHA1 : (KAT_Digest) : Pass
SHA2 : (KAT_Digest) : Pass
SHA3_256 : (KAT_Digest) : Pass
SHA3_512 : (KAT_Digest) : Pass
AES_GCM : (KAT_Cipher) : Pass
AES_ECB_Decrypt : (KAT_Cipher) : Pass
RSA : (KAT_Signature) : Pass
ECDSA : (PCT_Signature) : Pass
ECDSA : (PCT_Signature) : Pass
TLS13_KDF_EXTRACT : (KAT_KDF) : Pass
TLS13_KDF_EXPAND : (KAT_KDF) : Pass
TLS12_PRF : (KAT_KDF) : Pass
PBKDF2 : (KAT_KDF) : Pass
SSHKDF : (KAT_KDF) : Pass
```

```
KBKDF : (KAT_KDF) : Pass
HKDF : (KAT_KDF) : Pass
SSKDF : (KAT_KDF) : Pass
X963KDF : (KAT_KDF) : Pass
X942KDF : (KAT_KDF) : Pass
DH : (KAT_KA) : Pass
DH_FFDHE_2048 : (KAT_KA) : Pass
DH_MODP_2048 : (KAT_KA) : Pass
ECDH : (KAT_KA) : Pass
RSA_Encrypt : (KAT_AsymmetricCipher) : Pass
RSA_Decrypt : (KAT_AsymmetricCipher) : Pass
RSA_Decrypt : (KAT_AsymmetricCipher) : Pass
HMAC_SHA1 : (KAT_HMAC) : Pass
HMAC_SHA224 : (KAT_HMAC) : Pass
HMAC_SHA256 : (KAT_HMAC) : Pass
HMAC_SHA384 : (KAT_HMAC) : Pass
HMAC_SHA512 : (KAT_HMAC) : Pass
ECDSA : (KAT_Signature) : Pass
ECDSA : (KAT_Signature) : Pass
HMAC : (Module_Integrity) : Pass
```

5

CHAPTER

Configure Administrative Credentials and Privileges

IN THIS CHAPTER

Associated Password Rules for an Authorized Administrator Overview

The authorized administrator is associated with a defined login class, and the administrator is assigned with all permissions. Data is stored locally for fixed password authentication.

NOTE: Do not use control characters in passwords.

Use the following guidelines and configuration options for passwords and when selecting passwords for authorized administrator accounts. Passwords should be:

- Easy to remember so that users are not tempted to write it down.
- Changed periodically.
- Private and not shared with anyone.
- Contain a minimum of 10 characters. The minimum password length is 10 characters.

```
[ edit ]
security-administrator@host:fips# set system login password minimum-length 10
```

- Include both alphanumeric and punctuation characters, composed of any combination of upper and lowercase letters, numbers, and special characters such as, “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”. There should be at least a change in one case, one or more digits, and one or more punctuation marks.
- Contain character sets. Valid character sets include uppercase letters, lowercase letters, numbers, punctuation, and other special characters.

```
[ edit ]
security-administrator@host:fips# set system login password change-type character-sets
```

- Contain the minimum number of character sets or character set changes. The minimum number of character sets required in plain-text passwords in Junos FIPS is 3.

```
[ edit ]
security-administrator@host:fips# set system login password minimum-changes 3
```

- The hashing algorithm for user passwords can be either SHA256 or SHA512 (SHA512 is the default hashing algorithm).

  ```
  [ edit ]
  security-administrator@host:fips# set system login password format sha512
  ```

- If you are finished configuring the device, commit the configuration:

  ```
  [edit]
  security-administrator@host:fips# commit
  ```

NOTE: The device supports ECDSA (P-256, P-384, and P-521) and RSA (2048, 3072, and 4092 modulus bit length) key-types.

NOTE: The new hash algorithm affect only those passwords that are generated after commit.

Weak passwords are:

- Words that might be found in or exist as a permuted form in a system file such as **/etc/passwd**.
- The hostname of the system (always a first guess).
- Any words appearing in a dictionary. This includes dictionaries other than English, and words found in works such as Shakespeare, Lewis Carroll, Roget's Thesaurus, and so on. This prohibition includes common words and phrases from sports, sayings, movies, and television shows.
- Permutations on any of the above. For example, a dictionary word with vowels replaced with digits (for example f00t) or with digits added to the end.
- Any machine-generated passwords. Algorithms reduce the search space of password-guessing programs and so should not be used.

Strong reusable passwords can be based on letters from a favorite phrase or word, and then concatenated with other, unrelated words, along with additional digits and punctuation.

Configure a Network Device Collaborative Protection Profile Authorized Administrator

An account for `root` is always present in a configuration and is not intended for use in normal operation. In the evaluated configuration, the `root` account is restricted to the initial installation and configuration of the evaluated device.

An NDcPPv2.2e authorized administrator must have all permissions, including the ability to change the device configuration.

To configure an authorized administrator:

1. Create a login class named security-admin with all permissions.

```
[edit]
security-administrator@host:fips# set system login class security-admin permissions all
```

2. Configure the hashed algorithm for plain-text passwords as sha512.

```
[edit]
security-administrator@host:fips#  set system login password format sha512
```

3. Commit the changes.

```
[edit]
security-administrator@host:fips#  commit
```

4. Define your NDcPPv2.2e user authorized administrator.

```
[edit]
security-administrator@host:fips# set system login user NDcPPv2-user class security-admin
authentication encrypted-password
```

or

```
[edit]
security-administrator@host:fips# set system login user NDcPPv2-user class security-admin
authentication plain-text-password
```

5. Load an SSH key file that was previously generated using ssh-keygen. This command loads RSA (SSH version 2), or ECDSA (SSH version 2).

```
[edit]
security-administrator@host:fips# set system root-authentication load-key-file url:filename
```

6. Set the log-key-changes configuration statement to log when SSH authentication keys are added or removed.

```
[edit]
security-administrator@host:fips# set system services ssh log-key-changes
```

> **NOTE**: When the log-key-changes configuration statement is enabled and committed (with the commit command in configuration mode), Junos OS Evolved logs the changes to the set of authorized SSH keys for each user (including the keys that were added or removed). Junos OS Evolved logs the differences since the last time the log-key-changes configuration statement was enabled. If the log-key-changes configuration statement was never enabled, then Junos OS Evolved logs all the authorized SSH keys.

7. Commit the changes.

```
[edit]
security-administrator@host:fips# commit
```

For details on how to start with shell mode, see Overview for Junos OS Guide.

> **NOTE**: The root password should be reset following the change to sha256 / sha512 for the password storage format. This ensures the new password is protected using a sha256 / sha512 hash. To reset the root password, use set system root-authentication *plain-text-password* password command, and confirm the new password when prompted.

RELATED DOCUMENTATION

Associated Password Rules for an Authorized Administrator Overview | 35

Configure Global System NTP Server

To configure global system NTP server and set the date.

1. Configure the global system NTP server

```
[edit]
security-administrator@hostname:fips# set groups global system ntp server NTP-SERVER
security-administrator@hostname:fips# set system ntp authentication-key 1 type sha256
security-administrator@hostname:fips# set system ntp authentication-key 1 value Key1234
security-administrator@hostname:fips# set system ntp trusted-key 1
```

2. Configure the date

```
security-administrator@hostname:fips> set date ntp
```

Customize Time

To customize time, disable NTP and set the date.

1. Disable NTP.

```
[edit]
security-administrator@hostname:fips# deactivate groups global system ntp
security-administrator@hostname:fips# deactivate system ntp
security-administrator@hostname:fips# set system processes ntpd disable
security-administrator@hostname:fips# commit
security-administrator@hostname:fips# exit
```

2. Configure the date and time. Date and time (*current-time*) format is YYYYMMDDHHMM.ss.

```
security-administrator@hostname:fips> set date current-time
security-administrator@hostname:fips> set cli timestamp
```

Inactivity Timeout Period Configuration, and Local and Remote Idle Session Termination

IN THIS SECTION

Configure Session Termination

Terminate the session after the security administrator specifies inactive timeout period.

1. Set the idle timeout.

```
[edit]
security-administrator@host:fips#  set system login class security-admin idle-timeout 2
```

2. Configure the login access privileges.

```
[edit]
security-administrator@host:fips#  set system login class security-admin permissions all
```

3. Commit the configuration.

```
[edit]
security-administrator@host:fips# commit
```

```
commit complete
```

4. Set the password.

```
[edit]
security-administrator@host:fips# set system login user NDcPPv2-user authentication plain-
text-password
New password:
Retype new password:
```

5. Define login class.

```
[edit]
security-administrator@host:fips# set system login user NDcPPv2-user class security-admin
```

6. Commit the configuration.

```
[edit]
security-administrator@host:fips# commit
```

```
commit complete
```

Sample Output for Local Administrative Session Termination

```
con host
Trying a.b.c.d...
'autologin': unknown argument ('set ?' for help).
Connected to device.example.com
Escape character is '^]'.

Type the hot key to suspend the connection: <CTRL>Z
FreeBSD/amd64 (host) (ttyu0)
login: NDcPPv2-user
Password:
Last login: Sun Jun 23 22:42:27 from 10.224.33.70

--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20220816.a81ed05_buil
NDcPPv2-user@host> Warning: session will be closed in 1 minute if there is no activity
```

```
Warning: session will be closed in 10 seconds if there is no activity
Idle timeout exceeded: closing session

FreeBSD/amd64 (host) (ttyu0)
```

```
con host
Trying a.b.c.d...
'autologin': unknown argument ('set ?' for help).
Connected to device.example.com
Escape character is '^]'.

Type the hot key to suspend the connection: <CTRL>Z
FreeBSD/amd64 (host) (ttyu0)
login: NDcPPv2-user
Password:
Last login: Sun Jun 23 22:42:27 from 10.224.33.70

--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20230321.be5f9c0_buil
NDcPPv2-user@host> Warning: session will be closed in 1 minute if there is no activity
Warning: session will be closed in 10 seconds if there is no activity
Idle timeout exceeded: closing session

FreeBSD/amd64 (host) (ttyu0)
```

Sample Output for Remote Administrative Session Termination

```
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:48:05 2019
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20220816.a81ed05_buil
NDcPPv2-user@host> exit

Connection to host closed.
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:50:50 2019 from 10.224.33.70
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20220816.a81ed05_buil
NDcPPv2-user@host> Warning: session will be closed in 1 minute if there is no activity
```

```
Warning: session will be closed in 10 seconds if there is no activity
Idle timeout exceeded: closing session

Connection to host closed.
```

```
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:48:05 2019
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20230321.be5f9c0_buil
NDcPPv2-user@host> exit

Connection to host closed.
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:50:50 2019 from 10.224.33.70
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20230321.be5f9c0_buil
NDcPPv2-user@host> Warning: session will be closed in 1 minute if there is no activity
Warning: session will be closed in 10 seconds if there is no activity
Idle timeout exceeded: closing session

Connection to host closed.
```

Sample Output for User Initiated Termination

```
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:48:05 2019
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20220816.a81ed05_buil
NDcPPv2-user@host> exit

Connection to host closed.
```

```
ssh NDcPPv2-user@host
Password:
Last login: Sun Jun 23 22:48:05 2019
--- JUNOS 23.4R1.8 Kernel 64-bit  JNPR-12.1-20230321.be5f9c0_buil
NDcPPv2-user@host> exit
```

```
Connection to host closed.
```

6

CHAPTER

Configure SSH and Console Connection

IN THIS CHAPTER

Configure a System Login Message and Announcement

A system login message appears before the user logs in and a system login announcement appears after the user logs in. By default, no login message or announcement is displayed on the device.

To configure a system login message through console or management interface, use the following command:

```
[edit]
security-administrator@host:fips# set system login message login-message-banner-text
```

To configure system announcement, use the following command:

```
[edit]
security-administrator@host:fips# set system login announcement system-announcement-text
```

Commit the configuration:

```
[edit]
security-administrator@host:fips# commit
```

NOTE:

- If the message text contains any spaces, enclose it in quotation marks.
- You can format the message using the following special characters:
  - \n—New line
  - \t—Horizontal tab
  - \'—Single quotation mark
  - \"—Double quotation mark

- \\—Backslash

Configure SSH on the Evaluated Configuration for NDcPPv2.2e

SSH through remote management interface allowed in the evaluated configuration. This topic describes how to configure SSH for remote management of TOE. The following algorithms that needs to be configured to validate SSH for NDcPPv2.2e.

To configure SSH on the TOE:

1. Specify the permissible SSH host-key algorithms for the system services.

```
[edit]
security-administrator@host:fips# set system services ssh hostkey-algorithm-list rsa
security-administrator@host:fips# set system services ssh hostkey-algorithm-list ecdsa-sha2-nistp256
security-administrator@host:fips# set system services ssh hostkey-algorithm-list ecdsa-sha2-nistp384
security-administrator@host:fips# set system services ssh hostkey-algorithm-list ecdsa-sha2-nistp521
```

2. Specify the SSH key-exchange for Diffie-Hellman keys for the system services.

```
[edit]
security-administrator@host:fips# set system services ssh key-exchange dh-group14-sha1
security-administrator@host:fips# set system services ssh key-exchange ecdh-sha2-nistp256
security-administrator@host:fips# set system services ssh key-exchange ecdh-sha2-nistp384
security-administrator@host:fips# set system services ssh key-exchange ecdh-sha2-nistp521
```

3. Specify all the permissible message authentication code algorithms for SSHv2

```
[edit]
security-administrator@host:fips# set system services ssh macs hmac-sha1
```

```
security-administrator@host:fips# set system services ssh macs hmac-sha2-256
security-administrator@host:fips# set system services ssh macs hmac-sha2-512
```

4. Specify the ciphers allowed for protocol version 2.

```
[edit]security-administrator@host:fips# set system services ssh ciphers aes128-ctr
security-administrator@host:fips# set system services ssh ciphers aes256-ctr
```

5. Commit the changes:

```
[edit]
security-administrator@host:fips# commit
```

NOTE: To disable SSH service, you can deactivate and commit the SSH configurations:

```
security-administrator@host:fips# deactivate system services ssh
```

NOTE: To disable Netconf service, you can deactivate and commit the netconf configurations:

```
security-administrator@host:fips# deactivate system services netconf ssh
```

Supported SSH hostkey algorithm:

```
rsa   Allow generation of RSA host-key
ecdsa-sha2-nistp256   Allow generation of ecdsa-sha2-nistp256 host-key
ecdsa-sha2-nistp384   Allow generation of ecdsa-sha2-nistp384 host-key
ecdsa-sha2-nistp521   Allow generation of ecdsa-sha2-nistp521 host-key
```

Supported SSH key-exchange algorithm:

```
dh-group14-sha1      The RFC 4253 mandated group14 with SHA1 hash
ecdsa-sha2-nistp256  Allow generation of ECDSA host-key with NIST P-256 curve
```

```
ecdsa-sha2-nistp384  Allow generation of ECDSA host-key with NIST P-384 curve
ecdsa-sha2-nistp521  Allow generation of ECDSA host-key with NIST P-521 curve
```

Supported MACs algorithm:

```
hmac-sha1            Hash-based MAC using Secure Hash Algorithm (SHA1)
hmac-sha2-256        Hash-based MAC using Secure Hash Algorithm (SHA2)
hmac-sha2-512        Hash-based MAC using Secure Hash Algorithm (SHA2)
```

Supported SSH ciphers algorithm:

```
aes128-ctr           128-bit AES with Counter Mode
aes256-ctr           256-bit AES with Counter Mode
```

Limit the Number of User Login Attempts for SSH Sessions

An administrator may login remotely to a device through SSH. Administrator credentials are stored locally on the device. If the remote administrator presents a valid username and password, access to the TOE is granted. If the credentials are invalid, the TOE allows the authentication to be retried after an interval that starts after 1 second and increases exponentially. If the number of authentication attempts exceed the configured maximum, no authentication attempts are accepted for a configured time interval. When the interval expires, authentication attempts are again accepted.

You configure the amount of time the device gets locked after failed attempts. The amount of time in minutes before the user can attempt to log in to the device after being locked out due to the number of failed login attempts specified in the `tries-before-disconnect` statement. When a user fails to correctly login after the number of allowed attempts specified by the `tries-before-disconnect` statement, the user must wait the configured amount of minutes before attempting to log in to the device again. During this lockout-period the remote session user still have access to the TOE through the console as the root user.

The lockout-period must be greater than zero. The range at which you can configure the lockout-period is one through 43,200 minutes.

```
[edit system login]
security-administrator@host:fips# set retry-options lockout-period number
```

You can configure the device to limit the number of attempts to enter a password while logging through SSH. Use the following command to limit the number of login attempts.

```
[edit system login]
security-administrator@host:fips# set retry-options tries-before-disconnect number
```

Here, `tries-before-disconnect` is the number of times a user can attempt to enter a password when logging in. The connection closes if a user fails to log in after the number specified. The range is from 1 through 10, and the default value is 10.

The local administrator access will be maintained even if the remote administration is made permanently or temporarily unavailable due to the multiple failed login attempts. The console login for local administration will be available to the users during the lockout period.

You can also configure a delay, in seconds, before a user can try to enter a password after a failed attempt.

```
[edit system login]
security-administrator@host:fips# set retry-options backoff-threshold number
```

Here, `backoff-threshold` is the threshold for the number of failed login attempts before the user experiences a delay in being able to enter a password again. Use the `backoff-factor` option to specify the length of the delay in seconds. The range is from 1 through 3, and the default value is 2 seconds.

In addition, the device can be configured to specify the threshold for the number of failed attempts before the user experiences a delay in entering the password again.

```
[edit system login]
security-administrator@host:fips# set retry-options backoff-factor number
```

Here, `backoff-factor` is the length of time, in seconds, before a user can attempt to log in after a failed attempt. The delay increases by the value specified for each subsequent attempt after the threshold. The range is from 5 through 10, and the default value is 5 seconds.

You can control user access through SSH by configuring `ssh root-login deny`.

```
[edit system]
security-administrator@host:fips# set services ssh root-login deny
```

The SSH2 protocol provides secure terminal sessions utilizing the secure encryption. The SSH2 protocol enforces running the key-exchange phase and changing the encryption and integrity keys for the session. Key exchange is done periodically, after specified seconds or after specified bytes of data have passed over the connection. You can configure thresholds for SSH rekeying, FCS_SSHS_EXT.1.8 and FCS_SSHC_EXT.1.8. The TSF ensures that within the SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of the transmitted data. When either of the thresholds are reached, a rekey must be performed.

```
[edit system]
security-administrator@host:fips# set services ssh rekey time-limit number
```

Time limit before renegotiating session keys is 1 through 1440 minutes.

```
[edit system]
security-administrator@host:fips# set services ssh rekey data-limit  number
```

Data limit before renegotiating session keys is 51200 through 4294967295 byte.

NOTE: For SSH connection being unintentionally broken, we need to re-initiate the SSH connection to log in back to TOE.

7

CHAPTER

Configure the Remote Syslog Server

IN THIS CHAPTER

Sample Syslog Server Configuration on a Linux System

A secure Junos OS Evolved environment requires auditing of events and storing them in a local audit file. The recorded events are simultaneously sent to an external syslog server. A syslog server receives the syslog messages streamed from the device. The syslog server must have an SSH client with NETCONF support configured to receive the streamed syslog messages.

Use the configuration details and establish a session between the target of evaluation (TOE) and the audit server. Examine the traffic that passes between the audit server and the TOE during several activities, and the generated audit data to be transferred to the audit server.

Examine the TOE Summary Specification (TSS) to ensure that it specifies the means by which the audit data is transferred to the external audit server and how the trusted channel is provided.

The NDcPP logs capture the following events:

- Committed changes
- System startup
- Login and logout of users
- Failure to establish an SSH session
- Establishment or termination of an SSH session
- Changes to the system time
- Initiation of a system update

To configure event logging to a remote server when the SSH connection to the ToE is initiated from the remote system log server.

1. Generate an RSA public key on the remote syslog server.

   ```
   $ ssh-keygen -b 2048 -t rsa -C 'syslog-monitor key pair' -f ~/.ssh/syslog-monitor
   ```

   You will be prompted to enter the desired pass phrase. The storage locations for the `syslog-monitor` key pair is displayed.

2. On the TOE, create a class named `monitor` that has permission to trace events.

   ```
   [edit system login]
   security-administrator@host:fips# set class monitor permissions trace
   ```

3. Create a user named `syslog-mon` with the class monitor, and with authentication that uses the syslog-monitor key pair from the key pair file located on the remote syslog server.

   ```
   [edit system login]
   security-administrator@host:fips# set user syslog-mon class monitor authentication ssh-rsa
   "public-key"
   ```

4. Set up NETCONF with SSH.

   ```
   [edit system services]
   security-administrator@host:fips# set netconf ssh
   ```

5. Configure syslog to log all the messages at */var/log/messages.*.

   ```
   [edit system]
   security-administrator@host:fips# set syslog file messages any any
   commit
   ```

6. On the remote system log server, start up the SSH agent `ssh-agent`. The start up is required to simplify the handling of the syslog-monitor key.

   ```
   $ eval `ssh-agent -s`
   ```

7. On the remote syslog server, add the `syslog-monitor` key pair to the `ssh-agent`.

   ```
   $ ssh-add ~/.ssh/syslog-monitor
   ```

   You will be prompted to enter the desired passphrase. Enter the same passphrase used in Step 1.

8. After logging in to the `external_syslog_server` session, establish a tunnel to the device and start NETCONF.

```
security-administrator@host:fips# $ssh syslog-mon@NDcPP_TOE -s netconf > test.out
```

9. After NETCONF is established, configure a system log events message stream. This RPC will cause the NETCONF service to start transmitting messages over the SSH connection that is established.

   <rpc><get-syslog-events><stream>messages</stream></get-syslog-events></rpc>

10. The examples for syslog messages are listed below. Monitor the event log generated for admin actions on TOE are received on syslog server. Examine the traffic that passes between the audit server and the TOE, observing that these data are not viewed during this transfer, and that they are successfully received by the audit server. Match the logs between local event logging and remote event logged in syslog server and record the particular software (name, version) used on the audit server during testing.

The following output shows test log results for syslog-server.

```
host@ssh-keygen -b 2048 -t rsa -C 'syslog-monitor key pair' -f ~/.ssh/syslog-monitor
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/host/.ssh/syslog-monitor.
Your public key has been saved in /home/host/.ssh/syslog-monitor.pub.
The key fingerprint is:
ef:75:d7:68:c5:ad:8d:6f:5e:7a:7e:9b:3d:f1:4d:3f syslog-monitor key pair
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|                 |
|               ..|
|        S       +|
|         .     Bo|
|          . . *.X|
|         . . o E@|
|          .   .BX|
+-----------------+
[host@nms5-vm-linux2 ~]$ cat /home/host/.ssh/syslog-monitor.pub
ssh-rsa
 AAAAB3NzaC1yc2EAAAADAQABAAABAQCrUREJUBpjwAoIgRrGy9zgt+
D2pikk3Q/Wdf8I5vr+njeqJhCx2bUAkrRbYXNILQQAZbg7kLfi/8TqqL
eon4HOP2e6oCSorKdx/GrOTzLONL4fh0EyuSAk8bs5JuwWNBUokV025
gzpGFsBusGnlj6wqqJ/sjFsMmfxyCkbY+pUWb8m1/A9YjOFT+6esw+9S
tF6Gbg+VpbYYk/Oday4z+z7tQHRFSrxj2G92aoliVDBLJparEMBc8w
LdSUDxmgBTM2oadOmm+kreBUQjrmr6775RJn9H9YwIxKOxGm4SFnX/Vl4
R+lZ9RqmKH2wodIEM34K0wXEHzAzNZ01oLmaAVqT
syslog-monitor key pair
[host@nms5-vm-linux2 ~]$ eval `ssh-agent -s`
Agent pid 1453
[host@nms5-vm-linux2 ~]$ ssh-add ~/.ssh/syslog-monitor
Enter passphrase for /home/host/.ssh/syslog-monitor:
Identity added: /home/host/.ssh/syslog-monitor (/home/host/.ssh/syslog-monitor)
```

Net configuration channel

```
host@nms5-vm-linux2 ~]$ ssh syslog-mon@starfire -s netconf

 this is NDcPP test device

<!-- No zombies were killed during the creation of this user interface --
<!-- user syslog-mon, class j-monitor -><hello>
  <capabilities>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:confirmed-commit:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:validate:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:url:1.0?protocol=http,ftp,file</
capability>
    <capability>http://xml.juniper.net/netconf/junos/1.0</capability>
    <capability>http://xml.juniper.net/dmi/system/1.0</capability>
  </capabilities>
  <session-id4129/session-id>
</hello>
]]>]]>
```

The following output shows event logs generated on the TOE that are received on the syslog server.

```
Jan 20 17:04:51  starfire sshd[4182]: error: Could not load host key: /etc/ssh/ssh_host_dsa_key
Jan 20 17:04:51  starfire sshd[4182]: error: Could not load host key: /etc/ssh/ssh_host_ecdsa_key
Jan 20 17:04:53  starfire sshd[4182]: Accepted password for sec-admin from 10.209.11.24 port
55571 ssh2
```

```
Jan 20 17:04:53  starfire mgd[4186]: UI_AUTH_EVENT: Authenticated user 'sec-admin' at permission
level 'j-administrator'
Jan 20 17:04:53  starfire mgd[4186]: UI_LOGIN_EVENT: User 'sec-admin' login, class 'j-
administrator' [4186], ssh-connection '10.209.11.24 55571 10.209.14.92 22', client-mode 'cli'
```

Net configuration channel

```
host@nms5-vm-linux2 ~]$ ssh syslog-mon@starfire -s netconf
 this is NDcPP test device

<!-- No zombies were killed during the creation of this user interface --
<!-- user syslog-mon, class j-monitor -><hello>
  <capabilities>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:confirmed-commit:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:validate:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:capability:url:1.0?protocol=http,ftp,file</
capability>
    <capability>http://xml.juniper.net/netconf/junos/1.0</capability>
    <capability>http://xml.juniper.net/dmi/system/1.0</capability>
  </capabilities>
  <session-id4129/session-id>
</hello>
]]>]]>
```

The following output shows that the local syslogs and remote syslogs received were similar.

```
Local :
an 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Redundancy
interface management process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/rdd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/rdd', PID 4317,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Dynamic
flow capture service checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/dfcd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/dfcd', PID 4318,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress:
Connectivity fault management process checking new configuration
```

```
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/cfmd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/cfmd', PID 4319,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Layer 2
address flooding and learning process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/l2ald'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/l2ald', PID 4320,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Layer 2
Control Protocol process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/l2cpd'
Jan 20 17:09:30  starfire l2cp[4321]: Initializing PNAC state machines
Jan 20 17:09:30  starfire l2cp[4321]: Initializing PNAC state machines complete
Jan 20 17:09:30  starfire l2cp[4321]: Initialized 802.1X module and state machinesJan 20
17:09:30  starfire l2cp[4321]: Read acess profile () config
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/l2cpd', PID 4321,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Multicast
Snooping process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/mcsnoopd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/mcsnoopd', PID
4325, status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: commit
wrapup...
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress:
activating '/var/etc/ntp.conf'
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: start ffp
activate
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/ffp'
Jan 20 17:09:30  starfire ffp[4326]: "dynamic-profiles": No change to profiles
...............
```

```
Remote :
an 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Redundancy
interface management process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/rdd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/rdd', PID 4317,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Dynamic
flow capture service checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/dfcd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/dfcd', PID 4318,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress:
Connectivity fault management process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/cfmd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/cfmd', PID 4319,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Layer 2
address flooding and learning process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/l2ald'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/l2ald', PID 4320,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Layer 2
Control Protocol process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/l2cpd'
Jan 20 17:09:30  starfire l2cp[4321]: Initializing PNAC state machines
Jan 20 17:09:30  starfire l2cp[4321]: Initializing PNAC state machines complete
Jan 20 17:09:30  starfire l2cp[4321]: Initialized 802.1X module and state machinesJan 20
17:09:30  starfire l2cp[4321]: Read acess profile () config
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/l2cpd', PID 4321,
status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: Multicast
Snooping process checking new configuration
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/mcsnoopd'
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_STATUS: Cleanup child '/usr/sbin/mcsnoopd', PID
4325, status 0
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: commit
wrapup...
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress:
activating '/var/etc/ntp.conf'
Jan 20 17:09:30  starfire mgd[4186]: UI_COMMIT_PROGRESS: Commit operation in progress: start ffp
activate
Jan 20 17:09:30  starfire mgd[4186]: UI_CHILD_START: Starting child '/usr/sbin/ffp'
Jan 20 17:09:30  starfire ffp[4326]: "dynamic-profiles": No change to profiles
...............
```

8

CHAPTER

Configure Audit Log Options

IN THIS CHAPTER

Configure Audit Log Options in the Evaluated Configuration

IN THIS SECTION

The following section describes how to configure audit log options in the evaluated configuration.

Configure Audit Log Options

To configure audit log options:

1. Specify the number of files to be archived in the system logging facility.

```
[edit system syslog]
security-administrator@host:fips# set archive files 2
```

2. Specify the file in which to log data.

```
[edit system syslog]
security-administrator@host:fips# set file syslog any any
```

3. Specify the size of files to be archived.

```
[edit system syslog]
security-administrator@host:fips# set file syslog archive size 10000000
```

4. Specify the priority and facility in messages for the system logging facility.

```
[edit system syslog]
security-administrator@host:fips# set file syslog explicit-priority
```

5. Commit the changes:

```
[edit]
security-administrator@host:fips# commit
```

The system overwrites the audit data when the space for audit data is full as per the selection in FAU_STG_EXT.1.3.

Sample Code Audits of Configuration Changes

This sample code audits all changes to the configuration secret data and sends the logs to a file named **Audit-File:**

```
[edit system]
syslog {
    file Audit-File {
        authorization info;
        change-log info;
        interactive-commands info;
    }
}
```

This sample code expands the scope of the minimum audit to audit all changes to the configuration, not just secret data, and sends the logs to a file named **Audit-File:**

```
[edit system]
syslog {
    file Audit-File {
        any any;
        authorization info;
        change-log any;
        interactive-commands info;
        kernel info;
        pfe info;
    }
}
```

Example: System Logging of Configuration Changes

This example shows a sample configuration and makes changes to users and secret data. It then shows the information sent to the audit server when the secret data is added to the original configuration and committed with the `load` command.

```
[edit system]
location {
    country-code US;
    building B1;
}
...
login {
    message "UNAUTHORIZED USE OF THIS ROUTER\n\tIS STRICTLY PROHIBITED!";
        user admin {
            uid 2000;
             class super-user;
        authentication {
            encrypted-password “$ABC123”;
               # SECRET-DATA
        }
    }
}
radius-server 192.0.2.15 {
    secret “$ABC123” # SECRET-DATA
}
services {
    ssh;
}
syslog {
    user *{
        any emergency;
    }
    file messages {
        any notice;
        authorization info;
    }
    file interactive-commands {
        interactive-commands any;
    }
}
...
...
```

The new configuration changes the secret data configuration statements and adds a new user.

```
security-administrator@host:fips# show | compare
[edit system login user admin authentication]
-    encrypted-password “$ABC123”; # SECRET-DATA
+    encrypted-password “$ABC123”; # SECRET-DATA
[edit system login]
+    user admin2 {
+        uid 2001;
+        class operator;
+        authentication {
+            encrypted-password “$ABC123”;
                    # SECRET-DATA
+        }
+     }
[edit system radius-server 192.0.2.15]
-    secret “$ABC123”; # SECRET-DATA
+    secret “$ABC123”; # SECRET-DATA
```

Table 2 on page 64 shows sample for syslog auditing for NDcPPv2.1:

Table 2: Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FAU_GEN.1	None	None	-
FAU_GEN.2	None	None	-
FAU_STG_EXT.1	None	None	-
FAU_STG.1	None	None	-
FCS_CKM.1	None	None	-
FCS_CKM.2	None	None	-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FCS_CKM.4	None	None	
FCS_COP.1/DataEncryption	None	None	-
FCS_COP.1/SigGen	None	None	-
FCS_COP.1/Hash	None	None	-
FCS_COP.1/KeyedHash	None	None	-
FCS_COP.1(1)	None	None	-
FCS_COP.1	None	None	-
FCS_RBG_EXT.1	None	None	-
FIA_PMG_EXT.1	None	None	-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address)	Successful Local Login
			`login[27004]: pam_unix(login:session): session opened for user root by LOGIN(uid=0)`
			`login[27325]: ROOT LOGIN  on '/dev/ttyS0'`
			Unsuccessful Local Login
			`May 20 02:42:49 evoptx10k-b login[1342]: pam_unix(login:auth): authentication failure; logname=LOGIN uid=0 euid=0 tty=/dev/ttyS0 ruser= rhost=  user=root May 20 02:42:51 evoptx10k-b login[1342]: FAILED LOGIN (1) on '/dev/ttyS0' FOR 'root', Authentication failure`
			Successful Remote Login
			Jan 3 09:32:07 mgd[47035]: UI_AUTH_EVENT: Authenticated user 'test1' assigned to class 'j-read-only' Jan 3 09:32:07 mgd[47035]: UI_LOGIN_EVENT: User 'test1' login, class 'j-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			read-only' [47035], ssh-connection '10.1.5.153 36784 10.1.2.68 22', client-mode 'cli' Unsuccessful Remote Login `sshd[7352]: notice: Login failed for user 'root' from host`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address)	Successful Local Login
			`login[27004]: pam_unix(login:session): session opened for user root by LOGIN(uid=0)`
			`login[27325]: ROOT LOGIN  on '/dev/ttyS0'`
			Unsuccessful Local Login
			`May 20 02:42:49 evoptx10k-b login[1342]: pam_unix(login:auth): authentication failure; logname=LOGIN uid=0 euid=0 tty=/dev/ttyS0 ruser= rhost=  user=root May 20 02:42:51 evoptx10k-b login[1342]: FAILED LOGIN (1) on '/dev/ttyS0' FOR 'root', Authentication failure`
			Successful Remote Login
			Jan 3 09:32:07 mgd[47035]: UI_AUTH_EVENT: Authenticated user 'test1' assigned to class 'j-read-only' Jan 3 09:32:07 mgd[47035]: UI_LOGIN_EVENT: User 'test1' login, class 'j-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			read-only' [47035], ssh-connection '10.1.5.153 36784 10.1.2.68 22', client-mode 'cli' Unsuccessful Remote Login `sshd[7352]: notice: Login failed for user 'root' from host`
FIA_UAU.7	None	None	-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None	May 21 02:19:10 evoptx10k-b mgd[32755]: UI_CMDLINE_READ_LINE: User 'root', command 'request system software add /var/tmp/junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO.iso ' May 21 02:19:11 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : Download and Validate in Progress May 21 02:19:17 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Running pre-checks for 'junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO' May 21 02:19:19 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Pre-checks pass successfully, copying files to software area May 21 02:19:20 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Starting upgrade : /var/tmp/junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO.iso May 21 02:19:21 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Upgrade version : junos-evo-install-ptx-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

			`fixed-x86-64-23.4R2.6-EVO`
			`May 21 02:19:22 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Validating existing configs. See /var/log/validation_config.log for config validation logs.`

FMT_MTD.1/CoreData	None	None	-

FMT_SMF.1	Ability to start and stop services	None	Login as security-officer
			`security-officer@host:fips> request system reboot`
			`Reboot the system ? [yes,no] (no) yes`

	Ability to configure audit behaviour (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full)	None	`security-officer@host:fips#set system syslog archive files`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

	Ability to modify the behaviour of the transmission of audit data to an external IT entity	None	Generate an RSA public key on the remote syslog server
			ssh-keygen -b 2048 -t rsa -C 'syslog-monitor key pair' -f ~/.ssh/syslog-monitor
			[edit system login]
			security-officer@host:fips# set class monitor permissions trace
			[edit system login]
			security-officer@host:fips# set user syslog-mon class monitor authentication ssh-rsa "public-key"
			[edit system services]
			security-administrator@host:fips# set netconf ssh
			[edit system]
			security-officer@host:fips# set syslog file messages any any commit
			on the remote syslog server
			$ eval `ssh-agent -s`
			$ ssh-add ~/.ssh/syslog-monitor

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

	Ability to configure the cryptographic functionality	None	`security-officer@host:fips#set system services ssh`
			`security-officer@host:fips#set system services ssh ciphers aes128-ctr`

	Ability to configure thresholds for SSH rekeying	None	`security-officer@host:fips#set system services ssh`
			`security-officer@host:fips#set system services ssh rekey data-limit 51200`
			`security-officer@host:fips#set system services ssh rekey time-limit 1`

	Ability to re-enable an Administrator account	None	`root@fips#set system login user security-officer authentication plain-text-password`
			`New password:`
			`Retype new password:`
			`root@fips#set system login user security-officer class super-user`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

	Reset the password for security-officer	None	`root@fips#set system login user security-officer authentication plain-text-password`
			`New password:`
			`Retype new password:`

	Syslog check	None	Verify resetting passwords behavior through audit logs
			`root@fips>show log /var/log/messages1 | grep "UI_CFG_AUDIT_SET: User 'security-officer' set: \[system login user security-officer authentication \].*unconfigured" | except regress|count`
			`Count: 2 lines`

	Ability to set the time which is used for time-stamps	None	Login as security-officer and modify the time stamp
			`security-officer@fips-mx-b:fips>set date 202901010101.01`
			`Mon Jan  1 01:01:01 PST 2029`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

	Ability to manage the cryptographic keys	None	Ability to manage the trusted public keys database
			`Host_machine#ssh-keygen`
			`-t rsa -f $HOME/.ssh/`
			`id_ssh_rsa_2048 -N  -b`
			`2048`
			`Generating public/`
			`private rsa key pair.`
			`/root/.ssh/`
			`id_ssh_toby_rsa_2048`
			`already exists.`
			`Overwrite (y/n)? Your`
			`identification has been`
			`saved in /root/.ssh/`
			`id_ssh_toby_rsa_2048.`
			`Your public key has`
			`been saved in /`
			`root/.ssh/`
			`id_ssh_toby_rsa_2048.pub`
			`.`
			`The key fingerprint is:`
			`SHA256:m8ToMFz77/3rLDCK2`
			`rNFv9MaXpB0qmZUqAJMAEIX6`
			`X0 root@fips-qnc-`
			`lnx1.englab.juniper.net`
			`The key's randomart`
			`image is:`
			`+---[RSA 2048]----+`
			`|*o.oo            |`
			`|.o..     .       |`
			`|  + . . . o .    |`
			`|   + o E o +     |`
			`|    = = S +      |`
			`|     = = =o.     |`
			`|      ..O.o+.    |`
			`|     .o+.o.=o.   |`
			`|    ..oo .*o.+=. |`
			`+----[SHA256]-----+`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			Toby-1960280-10.48.155.181% cat $HOME/.ssh/id_ssh_rsa_2048.pub ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDMACOnJHF0UU+3fLO5ji7y9yBBQqolFjgGZ4PZsxOBW44NTYw1yp3cddih9XLEo5rGctThJfth6qIwLTkLdmw8FUIKvqU3szRztEuO/OKgchhi3E0YoPLBZI5M++Qth5e+hA65M/8Rub4CH2xkt2IIMZRDi51SLYecY0eIpGYs77o+u93x/rAe5BjooAfKe8UCwJRr2yxuZU/Xd2U0d6fFVASYIE8dvYI83chrLCC/WbaB3jUZk7tRumPlyq05vT0RXxzbzpffonRYsaaRnxPoc8xDr9uyDsiIQnA8cMM7H6ZxNHTfPOWSds1fraLEZsrsTOMrMBln5RNBZTc8sgbB root@fips-qnc-lnx1.englab.juniper.net security-officer@host:fips#set system login user syslog-mon authentication ssh-rsa "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDMACOnJHF0UU+3fLO5ji7y9yBBQqolFjgGZ4PZsxOBW44NTYw1yp3cddih9XLEo5rGctThJfth6qIwLTkLdmw8FUIKvqU3szRztEuO/OKgchhi3E0YoPLBZI5M+

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			+Qth5e+hA65M/8Rub4CH2xkt2IIMZRDi51SLYecY0eIpGYs77o+u93x/rAe5BjooAfKe8UCwJRr2yxuZU/Xd2U0d6fFVASYIE8dvYI83chrLCC/WbaB3jUZk7tRumPlyq05vT0RXxzbzpffonRYsaaRnxPoc8xDr9uyDsiIQnA8cMM7H6ZxNHTfPOWSds1fraLEZsrsTOMrMBln5RNBZTc8sgbB root@fips-qnc-lnx1.englab.juniper.net" security-officer@host:fips#set system login user syslog-mon class super-user Security Administrator may unlock an account that is locked from remote access (for example, SSH): Thu May 09 15:09:46 [user@ttbg-shell011:~]ssh test@nms-mx304-a Password: Password: Password: Received disconnect from 10.209.4.145 port 22:2: Too many password failures for test Disconnected from 10.209.4.145 port 22 Thu May 09 20:01:19 [user@ttbg-shell011:~]

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			[edit] root@host# run show system login lockout User Lockout start Lockout end test 2024-05-09 20:01:04 IST 2024-05-09 20:05:04 IST [edit] root@host# sshd: LIBJNX_LOGIN_ACCOUNT_LOCKED: Account for user 'test' has been locked out from logins sshd: PAM_USER_LOCK_LOGIN_REQUESTS_DENIED: Login requests from host '10.220.196.34' are denied sshd: PAM_USER_LOCK_ACCOUNT_LOCKED: Account for user test is locked. [edit] root@host# run clear system login lockout user test [edit] root@host# run show system login lockout User accounts not locked [edit]

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

			`root@host# run show system uptime`
			`Current time: 2024-05-09 20:03:10 IST`
			`Time Source:  LOCAL CLOCK`
			`System booted: 2024-05-07 19:19:44 IST (2d 00:43 ago)`
			`Protocols started: 2024-05-07 19:22:16 IST (2d 00:40 ago)`
			`Last configured: 2024-05-09 20:00:29 IST (00:02:41 ago) by root`
			` 8:03PM  up 2 days, 43 mins, 1 users, load averages: 0.21, 0.15, 0.10`
			
			`[edit]`
			`root@host#`
			
			`mgd[78360]: LIBJNX_LOGIN_ACCOUNT_UNLOCKED: Account for user 'test' has been unlocked for logins`

FMT_SMR.2	None	None	
FPT_SKP_EXT.1	None	None	
FPT_APW_EXT.1	None	None	

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FPT_TST_EXT.1	None	None	Reboot the device to view the self-test during startup.

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None	May 21 02:19:10 evoptx10k-b mgd[32755]: UI_CMDLINE_READ_LINE: User 'root', command 'request system software add /var/tmp/junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO.iso ' May 21 02:19:11 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : Download and Validate in Progress May 21 02:19:17 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Running pre-checks for 'junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO' May 21 02:19:19 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Pre-checks pass successfully, copying files to software area May 21 02:19:20 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Starting upgrade : /var/tmp/junos-evo-install-ptx-fixed-x86-64-23.4R2.6-EVO.iso May 21 02:19:21 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Upgrade version : junos-evo-install-ptx-

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			fixed-x86-64-23.4R2.6-EVO May 21 02:19:22 evoptx10k-b mgd[32755]: UI_SWUPDATE_EVENT: : re0: Validating existing configs. See /var/log/validation_config.log for config validation logs. May 21 02:31:24 evoptx10k-b mgd[21958]: UI_CMDLINE_READ_LINE: User 'root', command 'request system software add negate-sign-byte-evo-test-package.new.tgz ' May 21 02:31:24 evoptx10k-b mgd[21958]: UI_SWUPDATE_EVENT: : Download and Validate in Progress May 21 02:31:29 evoptx10k-b mgd[21958]: UI_SWUPDATE_EVENT: : re0: External Upgrade FAILED. See /var/log/extern_upgrade_master.log file for detailed errors May 21 02:31:29 evoptx10k-b mgd[21958]: UI_SWUPDATE_EVENT: : re0: Check whether the signing keys are installed on all REs May 21 02:31:33 evoptx10k-b mgd[21958]:

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

			`UI_SWUPDATE_EVENT: : ERROR: Signing keys are not installed. Node:re0 Image: re0:/data/var/home/root/test/negate-sign-byte-evo-test-package.new.tgz`
			`May 21 02:31:33 evoptx10k-b mgd[21958]: UI_SWUPDATE_EVENT: : External software upgrade failed.`

FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).	`Apr 22 15:31:37 mgd[11121]: UI_CMDLINE_READ_LINE: User 'root', command 'set date 201904221532.00`
			`Apr 22 15:32:05 mgd[11121]: UI_CMDLINE_READ_LINE: User 'root', command 'show system uptime`
			`May 21 02:51:18 evoptx10k-b ntpdate[10914]: NTP: System clock updated from 2024-05-21/09:48:48.473679 UTC to 2024-05-21/09:51:18.780343 UTC`
			`May 21 02:51:18 evoptx10k-b systemd[1]: Started "NTP(Network Time Protocol) Daemon".`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FTA_SSL_EXT.1 (if “terminate the session is selected)	The termination of a local interactive session by the session locking mechanism.	None	Jan 3 11:59:29 cli: UI_CLI_IDLE_TIMEOUT: Idle timeout for user 'root' exceeded and session terminated
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None	Jan 3 11:26:23 cli: UI_CLI_IDLE_TIMEOUT: Idle timeout for user 'root' exceeded and session terminated
FTA_SSL.4	The termination of an interactive session.	None	Local Jan 3 11:47:25 mgd[52521]: UI_LOGOUT_EVENT: User 'root' logout Remote Jan 3 11:43:33 sshd[52425]: Received disconnect from 10.1.5.153 port 36800:11: disconnected by user
FTA_TAB.1	None	None	

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.	Initiation of the trusted path Jan 3 12:09:00 sshd[53492]: Accepted keyboard-interactive/pam for root from 10.1.5.153 port 36802 ssh2 Termination of the trusted path Jan 3 12:09:03 sshd[53492]: Received disconnect from 10.1.5.153 port 36802:11: disconnected by user Jan 3 12:09:36 sshd: Failure of the trusted path `sshd[3790]: notice: Login failed for user 'root' from host '10.32.196.40`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None	Initiation of the trusted path Jan 3 12:09:00 sshd[53492]: Accepted keyboard-interactive/pam for root from 10.1.5.153 port 36802 ssh2 Termination of the trusted path Jan 3 12:09:03 sshd[53492]: Received disconnect from 10.1.5.153 port 36802:11: disconnected by user Jan 3 12:09:36 sshd: Failure of the trusted path `sshd[3790]: notice: Login failed for user 'root' from host '10.32.196.40`
FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure	Dec 17 15:02:12 sshd[9842]: Unable to negotiate with 10.1.5.153 port 43836: no matching key exchange method found. Their offer: diffie-hellman-group1-sha1,ext-info-c

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
FIA_X509_EXT.2	None	None	
FMT_MOF.1/Functions	None	None	
FMT_MOF.1/Services	None	None	
FMT_MTD.1/CryptoKeys	None	None	

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

FIA_AFL.1	Administrator lockout due to excessive authentication failures	Origin of the attempt (e.g., IP address).	sshd - SSHD_LOGIN_ATTEMPTS_THRESHOLD: Threshold for unsuccessful authentication attempts (3) reached by user ' security-administrator'
			Login lockout configuration details:
			`[edit]`
			`root@host:fips# run show system login lockout`
			`User                    Lockout start              Lockout end`
			`security-administrator  2023-01-10 15:03:26 IST    2023-01-10 15:04:26 IST`
			Log for the login lockout configuration:
			Jan 10 15:03:26 host sshd[63687]: LIBJNX_LOGIN_ACCOUNT_LOCKED: Account for user 'security-administrator' has been locked out from logins
			Status of the session closed after the lockout period:

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

			`ssh security-administrator@host`
			`Password:`
			`Connection closed by 10.209.21.170 port 22`
			Log for the closed session after lockout period:
			Jan 10 15:04:10 host sshd[63694]: PAM_USER_LOCK_ACCOUNT_LOCKED: Account for user security-administrator is locked.
			Security Administrator may unlock an account that is locked from remote access (for example, SSH):
			`Thu May 09 15:09:46`
			`[user@ttbg-shell011:~]ssh test@host`
			`Password:`
			`Password:`
			`Password:`
			`Received disconnect from 10.209.4.145 port 22:2: Too many password failures for test`
			`Disconnected from 10.209.4.145 port 22`
			`Thu May 09 20:01:19`
			`[user@ttbg-shell011:~]`
			
			`[edit]`
			`root@host# run show`

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated
			system login lockout User Lockout start Lockout end test 2024-05-09 20:01:04 IST 2024-05-09 20:05:04 IST [edit] root@host# sshd: LIBJNX_LOGIN_ACCOUNT_LOCKED: Account for user 'test' has been locked out from logins sshd: PAM_USER_LOCK_LOGIN_REQUESTS_DENIED: Login requests from host '10.220.196.34' are denied sshd: PAM_USER_LOCK_ACCOUNT_LOCKED: Account for user test is locked. [edit] root@host# run clear system login lockout user test [edit] root@host# run show system login lockout User accounts not locked [edit] root@host# run show system uptime Current time:

Table 2: Auditable Events *(Continued)*

Requirement	Auditable Events	Additional Audit Record Contents	How event generated

			`2024-05-09 20:03:10 IST`
			`Time Source:  LOCAL`
			`CLOCK`
			`System booted:`
			`2024-05-07 19:19:44 IST`
			`(2d 00:43 ago)`
			`Protocols started:`
			`2024-05-07 19:22:16 IST`
			`(2d 00:40 ago)`
			`Last configured:`
			`2024-05-09 20:00:29 IST`
			`(00:02:41 ago) by root`
			` 8:03PM  up 2 days, 43`
			`mins, 1 users, load`
			`averages: 0.21, 0.15,`
			`0.10`
			
			`[edit]`
			`root@host#`
			
			`mgd[78360]:`
			`LIBJNX_LOGIN_ACCOUNT_UNL`
			`OCKED: Account for user`
			`'test' has been`
			`unlocked for logins`

9

CHAPTER

Configure Event Logging

IN THIS CHAPTER

Event Logging Overview

The evaluated configuration requires the auditing of configuration changes through the system log.

In addition, Junos OS Evolved can:

- Send automated responses to audit events (syslog entry creation).
- Allow authorized managers to examine audit logs.
- Send audit files to external servers.
- Allow authorized managers to return the system to a known state.

The logging for the evaluated configuration must capture the following events:

- Changes to secret key data in the configuration.
- Committed changes.
- Login/logout of users.
- System startup.
- Failure to establish an SSH session.
- Establishment/termination of an SSH session.
- Changes to the (system) time.
- Termination of a remote session by the session locking mechanism.
- Termination of an interactive session.

In addition, Juniper Networks recommends that logging also:

- Capture all changes to the configuration.
- Store logging information remotely.

Configure Event Logging to a Local File

You can configure storing of audit information to a local file with the `syslog` statement. This example stores logs in a file named **Audit-File**:

```
[edit system]
syslog {
    file Audit-File;
}
```

Interpret Event Messages

The following output shows a sample event message.

```
Feb 27 02:33:04  bm-a mgd[6520]: UI_LOGIN_EVENT: User 'security-officer' login, class 'j-super-
user' [6520], ssh-connection '', client-mode 'cli'
Feb 27 02:33:49  bm-a mgd[6520]: UI_DBASE_LOGIN_EVENT: User 'security-officer' entering
configuration mode
Feb 27 02:38:29  bm-a mgd[6520]: UI_CMDLINE_READ_LINE: User 'security-officer', command 'run
show log Audit_log | grep LOGIN
```

Table 3 on page 95 describes the fields for an event message. If the system logging utility cannot determine the value in a particular field, a hyphen (-) appears instead.

Table 3: Fields in Event Messages

Field	Description	Examples

timestamp	Time when the message was generated, in one of two representations:
	• *MMM-DD HH:MM:SS.MS+/-HH:MM,* is the month, day, hour, minute, second and millisecond in local time. The hour and minute that follows the plus sign (+) or minus sign (-) is the offset of the local time zone from Coordinated Universal Time (UTC).
	• *YYYY-MM-DDTHH:MM:SS.MSZ* is the year, month, day, hour, minute, second and millisecond in UTC.
		`Feb 27 02:33:04 is the timestamp expressed as local time in the United States.`
		`2012-02-27T09:17:15.719Z is  2:33 AM UTC on 27 Feb 2012.`

hostname	Name of the host that originally generated the message.	`router1`
process	Name of the Junos OS Evolved process that generated the message.	`mgd`
processID	UNIX process ID (PID) of the Junos OS Evolved process that generated the message.	`4153`
TAG	Junos OS Evolved system log message tag, which uniquely identifies the message.	`UI_DBASE_LOGOUT_EVENT`
username	Username of the user initiating the event.	`“admin”`
message-text	English-language description of the event .	`set: [system radius-server 1.2.3.4 secret]`

Log Changes to Secret Data

The following are examples of audit logs of events that change the secret data. Whenever there is a change in the configuration example, the syslog event should capture the below logs:

```
Jul 24 17:43:28  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' set: [system radius-server 1.2.3.4 secret]
Jul 24 17:43:28  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' set: [system login user admin authentication encrypted-password]
Jul 24 17:43:28  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' set: [system login user admin2 authentication encrypted-password]
```

Everytime a configuration is updated or changed, the syslog should capture these logs:

```
Jul 24 18:29:09  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' replace: [system radius-server 1.2.3.4 secret]
Jul 24 18:29:09  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' replace: [system login user admin authentication encrypted-password]
Jul 24 18:29:09  router1 mgd[4163]: UI_CFG_AUDIT_SET_SECRET: User 'admin' replace: [system login user admin authentication encrypted-password]
```

Login and Logout Events Using SSH

System log messages are generated whenever a user successfully or unsuccessfully attempts SSH access. Logout events are also recorded. For example, the following logs are the result of two failed authentication attempts, then a successful one, and finally a logout:

```
evoptx10k-b sshd[28663]: notice: Login failed for user 'root' from host '172.17.58.45'
evoptx10k-b sshd[28663]: Accepted keyboard-interactive/pam for root from 172.17.58.45 port 57956 ssh2
```

10

CHAPTER

Configure MACsec

IN THIS CHAPTER

Media Access Control Security (MACsec) in FIPS Mode Overview

Media Access Control Security (MACsec) is an 802.1AE IEEE industry-standard security technology that provides secure communication for all traffic on Ethernet links. MACsec provides point-to-point security on Ethernet links between directly connected nodes and is capable of identifying and preventing most security threats, including denial of service, intrusion, man-in-the-middle, masquerading, passive wiretapping, and playback attacks.

MACsec allows you to secure point to point Ethernet link for almost all traffic, including frames from the Link Layer Discovery Protocol (LLDP), Link Aggregation Control Protocol (LACP), Dynamic Host Configuration Protocol (DHCP), Address Resolution Protocol (ARP), and other protocols that are not typically secured on an Ethernet link because of limitations with other security solutions. MACsec can be used in combination with other security protocols such as IP Security (IPsec) and Secure Sockets Layer (SSL) to provide end-to-end network security.

MACsec is standardized in IEEE 802.1AE. The IEEE 802.1AE standard can be seen on the IEEE organization website at IEEE 802.1: BRIDGING & MANAGEMENT.

Each implementation of an algorithm is checked by a series of known answer test (KAT) self-tests and crypto algorithms validations (CAV). The following cryptographic algorithms are added specifically for MACsec.

- Advanced Encryption Standard (AES)-Cipher Message Authentication Code (CMAC)
- Advanced Encryption Standard (AES) Key Wrap

Pre-shared key configurations for both connectivity association key name (CKN) and connectivity association key (CAK):

```
[edit]
security-administrator@hostname:fips# prompt security macsec connectivity-association ca_name
pre-shared-key cak
New cak (secret):
Retype new cak (secret):
```

```
[edit]
security-administrator@hostname:fips# set security macsec connectivity-association ca_name pre-
shared-key ckn ckn
```

NOTE: In the above `set security macsec connectivity-association` *`ca_name`* `pre-shared-key ckn` *`ckn`* command, you need to define a user defined name for the *ca_name* variable option and a user defined connectivity association key name in hexadecimal format for *ckn* variable option.

A pre-shared key is exchanged between directly-connected links to establish a MACsec-secure link. The pre-shared-key includes the CKN and the CAK. The CKN is a 64-digit hexadecimal number and the CAK is a 32-digit hexadecimal number. The CKN and CAK must match on both ends of a link to create a MACsec-secured link.

NOTE: To maximize security, we recommend you to configure all 64 digits of a CKN and all 32 digits of a CAK. If you do not configure all 64 digits of a CKN or all 32 digits of a CAK, the system auto-configures all the remaining digits to 0. However, you will receive a warning message when you commit the configuration.

After the successful exchange and verification of the pre-shared keys by both ends of the link, the MACsec Key Agreement (MKA) protocol enables and manages the secure link. The MKA protocol then elects one of the two directly-connected switches as the key server. The key server then generates SAKs for encrypting and authenticating data frames and distributes SAKs to the other MACsec endpoint.

For example, you can configure a CKN of `37c9c2c45ddd012aa5bc8ef284aa23ff6729ee2e4acb66e91fe34ba2cd9fe311` and CAK of `228ef255aa23ff6729ee664acb66e91f` on connectivity association.

RELATED DOCUMENTATION

Understanding Media Access Control Security (MACsec)

Configure MACsec

IN THIS SECTION

We can configure MACsec to secure point-to-point Ethernet links connecting your device with MACsec-capable MICs. Each point-to-point Ethernet link that you want to secure using MACsec must be configured independently. We can enable MACsec on device-to-device links using static connectivity association key (CAK) security mode.

You can configure different interface rates such as 40G, 100G, and 10G in port mode and specific interface rates such as 100G, 40G, and 10G in pic mode. In pic mode you can configure only one type of interface speed.

Configuring MACsec on a Device Running Junos OS Evolved

To configure MACsec on a device running Junos OS Evolved:

1. Configure the MACsec security mode as for the connectivity association.

```
[edit]
security-administrator@host:fips#  set security macsec connectivity-association connectivity-association-name exclude-protocol protocol-name
security-administrator@host:fips#  set security macsec connectivity-association connectivity-association-name include-sci
security-administrator@host:fips#  set security macsec connectivity-association connectivity-association-name mka key-server-priority priority-number
security-administrator@host:fips#  set security macsec connectivity-association connectivity-association-name mka transmit-interval interval
security-administrator@host:fips#  set security macsec connectivity-association connectivity-association-name offset 30
```

> **NOTE**: Based on your requirement you can configure the `offset` *`offset-number`* value at the `set security macsec connectivity-association` *`connectivity-association-name`* hierarchy level to *`0`*, *`30`*, or *`50`*.

2. Create the pre-shared key by configuring the connectivity association key name (CKN) and connectivity association key (CAK).

```
[edit]
security-administrator@host:fips# prompt security macsec connectivity-association connectivity-association-name pre-shared-key cak
New cak (secret):
Retype new cak (secret):
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name pre-shared-key ckn hexadecimal-number
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name replay-protect replay-window-size number-of-packets
```

NOTE: Based on your requirement you can configure the *number-of-packets* value at the `set security macsec connectivity-association` *`connectivity-association-name`* `replay-protect replay-window-size` hierarchy level from `0` through `65535`.

3. Set the MACsec Key Agreement (MKA) secure channel details.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name secure-channel secure-channel-name direction (inbound | outbound)
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name secure-channel secure-channel-name id mac-address mac-address
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name secure-channel secure-channel-name id port-id port-id-number
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name secure-channel secure-channel-name offset (0|30|50)
security-administrator@host:fips# set security macsec connectivity-association connectivity-association-name secure-channel secure-channel-name security-association security-association-number key key-string
```

4. Set the MKA to security mode.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-mode security-mode
```

NOTE: `CA1` is an example of *connectivity-association-name* configured.

5. Assign the configured connectivity association with a specified MACsec interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-
association connectivity-association-name
```

Configuring Static MACsec with Layer 3 Traffic

To configure Static MACsec using Layer 3 traffic between device R0 and device R1:

In R0:

1. Create the preshared key by configuring the connectivity association key name (CKN) and connectivity association key (CAK)

```
[edit]
security-administrator@host:fips#  set security macsec connectivity-association CA1 pre-
shared-key ckn 2345678922334455667788992223334445556667778889992222333344445555
security-administrator@host:fips#  prompt security macsec connectivity-association CA1 pre-
shared-key cak
New cak (secret):
Retype new cak (secret):
security-administrator@host:fips#  set security macsec connectivity-association CA1 offset 30
```

2. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

3. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name traceoptions
```

```
file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name traceoptions flag all
```

4. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-mode static-cak
```

5. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-server-priority 1
```

6. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka transmit-interval 3000
```

7. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

8. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-association CA1
security-administrator@host:fips# set interfaces interface-name unit 0 family inet address 10.1.1.1/24
```

In R1:

1. Create the preshared key by configuring the connectivity association key name (CKN) and connectivity association key (CAK)

   ```
   [edit]
   security-administrator@host:fips#  set security macsec connectivity-association CA1 pre-
   shared-key ckn 2345678922334455667788992223334445556667778889992222333344445555
   security-administrator@host:fips#  prompt security macsec connectivity-association CA1 pre-
   shared-key cak
   New cak (secret):
   Retype new cak (secret):
   security-administrator@host:fips#  set security macsec connectivity-association CA1 offset 30
   ```

2. Set the trace option values.

   ```
   [edit]
   security-administrator@host:fips# set security macsec traceoptions file MACsec.log
   security-administrator@host:fips# set security macsec traceoptions file size 4000000000
   security-administrator@host:fips# set security macsec traceoptions flag all
   ```

3. Assign the trace to an interface.

   ```
   [edit]
   security-administrator@host:fips# set security macsec interfaces interface-name traceoptions
   file mka_xe size 1g
   security-administrator@host:fips# set security macsec interfaces interface-name traceoptions
   flag all
   ```

4. Configure the MACsec security mode as static-cak for the connectivity association.

   ```
   [edit]
   security-administrator@host:fips# set security macsec connectivity-association CA1 security-
   mode static-cak
   ```

5. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka
transmit-interval 3000
```

6. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

7. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-
association CA1
security-administrator@host:fips# set interfaces interface-name unit 0 family inet address
10.1.1.2/24
```

Configuring MACsec with keychain using Layer 3 Traffic

Synchronize both macsec endpoint devices to NTP as both device’s time should be the same for key start time triggers. To configure MACsec with keychain using Layer 3 traffic between device R0 and device R1:

In R0:

1. Assign a tolerance value to the authentication key chain.

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 tolerance 20
```

2. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

You can configure upto 64 keys. For example, you can refer the following keys:

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 key-name 2345678922334455667788992223334445556667778889992222333344445551
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 start-time 2018-03-20.20:35
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 key-name 2345678922334455667788992223334445556667778889992222333344445552
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 start-time 2018-03-20.20:37
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 key-name 2345678922334455667788992223334445556667778889992222333344445553
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 start-time 2018-03-20.20:39
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 key-name 2345678922334455667788992223334445556667778889992222333344445554
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 start-time 2018-03-20.20:41
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 key-name 2345678922334455667788992223334445556667778889992222333344445555
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 start-time 2018-03-20.20:43
```

Use the `prompt` command to enter a secret key value. For example, the secret key value is *2345678922334455667788992223334123456789223344556677889922233341*.

You can configure upto 64 secret keys. For example, you can refer the following secret keys:

```
[edit]
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 0 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 1 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 2 secret
New secret (secret):
Retype new secret (secret):
```

```
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key  3 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key  4 secret
New secret (secret):
Retype new secret (secret):
```

3. Associate the preshared keychain name with the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 pre-
shared-key-chain macsec-kc1
security-administrator@host:fips# set security macsec connectivity-association CA1 offset  50
security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-
suite gcm-aes-256
```

NOTE: The cipher value can also be set as **cipher-suite gcm-aes-128**.

4. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

5. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions flag all
```

6. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-mode static-cak
```

7. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-server-priority 1
```

8. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka transmit-interval 3000
```

9. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

10. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-association CA1
security-administrator@host:fips# set interfaces interface-name unit 0 family inet address 10.1.1.1/24
```

To configure MACsec with keychain for Layer 3 traffic:

In R1:

1. Assign a tolerance value to the authentication key chain.

   ```
   [edit]
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 tolerance 20
   ```

2. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

   You can configure upto 64 keys. For example, you can refer the following keys:

   ```
   [edit]
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 0 key-name 2345678922334455667788992223334445556667778889992222333344445551
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 0 start-time 2018-03-20.20:35
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 1 key-name 2345678922334455667788992223334445556667778889992222333344445552
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 1 start-time 2018-03-20.20:37
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 2 key-name 2345678922334455667788992223334445556667778889992222333344445553
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 2 start-time 2018-03-20.20:39
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 3 key-name 2345678922334455667788992223334445556667778889992222333344445554
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 3 start-time 2018-03-20.20:41
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 4 key-name 2345678922334455667788992223334445556667778889992222333344445555
   security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 key 4 start-time 2018-03-20.20:43
   ```

   Use the `prompt` command to enter a secret key value. For example, the secret key value is *2345678922334455667788992223334123456789223344556677889922233341*.

   You can configure upto 64 secret keys. For example, you can refer the following secret keys:

   ```
   [edit]
   security-administrator@host:fips#  prompt security authentication-key-chains key-chain
   ```

```
macsec-kc1 key 0 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 1 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 2 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 3 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 4 secret
New secret (secret):
Retype new secret (secret):
```

3. Associate the preshared keychain name with the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 pre-
shared-key-chain macsec-kc1
security-administrator@host:fips# set security macsec connectivity-association CA1 offset 50
security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-
suite gcm-aes-256
```

4. **NOTE**:

   - You can use the non-XPN ciphers `AES-GCM-128` and `AES-GCM-256` for 10G/xe interfaces macsec configuration only.

   - You can use the XPN ciphers `AES-GCM-XPN-128` and `AES-GCM-XPN-256` for 40G and 100G rates macsec configuration. You can also use the XPN ciphers `AES-GCM-XPN-128` and `AES-GCM-XPN-256` for 10G/xe interfaces macsec configuration, if it supports.

5. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

6. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions flag all
```

7. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-
mode static-cak
```

8. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-
server-priority 1
```

9. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka
transmit-interval 3000
```

10. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

11. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
connectivity-association CA1
security-administrator@host:fips# set interfaces interface-name unit 0 family inet address
10.1.1.2/24
```

Configuring Static MACsec for Layer 2 Traffic

To configure static MACsec for Layer 2 traffic between device R0 and device R1:

In R0:

1. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-server-priority 1
```

2. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

```
[edit]
security-administrator@host:fips# prompt security authentication-key-chains key-chain
macsec-kc1 key 0 secret
New secret (secret):
Retype new secret (secret):
```

For example, the secret key value is *2345678922334455667788992223334123456789223344556677889922233341*.

3. Associate the preshared keychain name with the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 pre-shared-key-chain macsec-kc1
security-administrator@host:fips# set security macsec connectivity-association CA1 offset 50
security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-suite gcm-aes-256
```

4. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

5. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name traceoptions file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name traceoptions flag all
```

6. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-mode static-cak
```

7. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-server-priority 1
```

8. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka
transmit-interval 3000
```

9. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-
sci
```

10. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
connectivity-association CA1
```

11. Configure VLAN tagging.

```
[edit]
security-administrator@host:fips# set interfaces interface-name1 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name1 encapsulation flexible-
ethernet-services
security-administrator@host:fips# set interfaces interface-name1 unit 100 encapsulation
vlan-bridge
security-administrator@host:fips# set interfaces interface-name1 unit 100 vlan-id 100
security-administrator@host:fips# set interfaces interface-name2 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name2 encapsulation flexible-
ethernet-services
security-administrator@host:fips# set interfaces interface-name2 unit 100 encapsulation
vlan-bridge
security-administrator@host:fips# set interfaces interface-name2 unit 100 vlan-id 100
```

In R1:

1. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

   ```
   [edit]
   security-administrator@host:fips#  prompt security authentication-key-chains key-chain
   macsec-kc1 key 0 secret
   New secret (secret):
   Retype new secret (secret):
   ```

   For example, the secret key value is 2345678922334455667788992223334123456789223344556677889922233341.

2. Associate the preshared keychain name with the connectivity association.

   ```
   [edit]
   security-administrator@host:fips# set security macsec connectivity-association CA1 pre-
   shared-key-chain macsec-kc1
   security-administrator@host:fips# set security macsec connectivity-association CA1 offset 50
   security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-
   suite gcm-aes-256
   ```

3. Set the trace option values.

   ```
   [edit]
   security-administrator@host:fips# set security macsec traceoptions file MACsec.log
   security-administrator@host:fips# set security macsec traceoptions file size 4000000000
   security-administrator@host:fips# set security macsec traceoptions flag all
   ```

4. Assign the trace to an interface.

   ```
   [edit]
   security-administrator@host:fips# set security macsec interfaces interface-name
   traceoptions file mka_xe size 1g
   security-administrator@host:fips# set security macsec interfaces interface-name
   traceoptions flag all
   ```

5. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-
mode static-cak
```

6. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-
server-priority 1
```

7. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka
transmit-interval 3000
```

8. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-
sci
```

9. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
connectivity-association CA1
```

10. Configure VLAN tagging.

```
[edit]
security-administrator@host:fips# set interfaces interface-name1 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name1 encapsulation flexible-
ethernet-services
security-administrator@host:fips# set interfaces interface-name1 unit 100 encapsulation
```

```
vlan-bridge
security-administrator@host:fips# set interfaces interface-name1 unit 100 vlan-id 100
security-administrator@host:fips# set interfaces interface-name2 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name2 encapsulation flexible-
ethernet-services
security-administrator@host:fips# set interfaces interface-name2 unit 100 encapsulation
vlan-bridge
security-administrator@host:fips# set interfaces interface-name2 unit 100 vlan-id 100
```

Configuring MACsec with keychain for Layer 2 Traffic

Synchronize both macsec endpoint devices to NTP as both device's time should be the same for key start time triggers. To configure MACsec with keychain for Layer 3 traffic between device R0 and device R1:

In R0:

1. Assign a tolerance value to the authentication key chain.

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 tolerance 20
```

2. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

   You can configure upto 64 keys. For example, you can refer the following keys:

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 key-name 23456789223344556677889922233344455566677788889992222333344445551
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 start-time 2018-03-20.20:35
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 key-name 23456789223344556677889922233344455566677788889992222333344445552
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 start-time 2018-03-20.20:37
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 key-name 23456789223344556677889922233344455566677788889992222333344445553
```

```
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 start-time 2018-03-20.20:39
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 key-name 234567892233445566778899222333444455566677788899922223333444445554
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 start-time 2018-03-20.20:41
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 key-name 234567892233445566778899222333444455566677788899922223333444445555
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 start-time 2018-03-20.20:43
```

Use the `prompt` command to enter a secret key value. For example, the secret key value is *2345678922334455667788992223334123456789223344556677889922233341*.

You can configure upto 64 secret keys. For example, you can refer the following secret keys:

```
[edit]
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 0 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 1 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 2 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 3 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 4 secret
New secret (secret):
Retype new secret (secret):
```

3. Associate the preshared keychain name with the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 pre-
```

```
shared-key-chain macsec-kc1
security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-
suite gcm-aes-256
```

4. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

5. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions flag all
```

6. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-
mode static-cak
```

7. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-
server-priority 1
```

8. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka
transmit-interval 3000
```

9. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

10. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-association CA1
```

11. Configure VLAN tagging.

```
[edit]
security-administrator@host:fips# set interfaces interface-name1 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name1 encapsulation flexible-ethernet-services
security-administrator@host:fips# set interfaces interface-name1 unit 100 encapsulation vlan-bridge
security-administrator@host:fips# set interfaces interface-name1 unit 100 vlan-id 100
security-administrator@host:fips# set interfaces interface-name2 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name2 encapsulation flexible-ethernet-services
security-administrator@host:fips# set interfaces interface-name2 unit 100 encapsulation vlan-bridge
security-administrator@host:fips# set interfaces interface-name2 unit 100 vlan-id 100
```

In R1:

1. Assign a tolerance value to the authentication key chain.

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-kc1 tolerance 20
```

2. Create the secret password to use. It is a string of hexadecimal digits up to 64 characters long. The password can include spaces if the character string is enclosed in quotation marks. The keychain's secret-data is used as a CAK.

```
[edit]
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 key-name 2345678922334455667788992223334445556667778889992222333344445551
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 0 start-time 2018-03-20.20:35
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 key-name 2345678922334455667788992223334445556667778889992222333344445552
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 1 start-time 2018-03-20.20:37
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 key-name 2345678922334455667788992223334445556667778889992222333344445553
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 2 start-time 2018-03-20.20:39
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 key-name 2345678922334455667788992223334445556667778889992222333344445554
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 3 start-time 2018-03-20.20:41
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 key-name 2345678922334455667788992223334445556667778889992222333344445555
security-administrator@host:fips#  set security authentication-key-chains key-chain macsec-
kc1 key 4 start-time 2018-03-20.20:43
```

Use the `prompt` command to enter a secret key value. For example, the secret key value is *2345678922334455667788992223334123456789223344556677889922233341*.

You can configure upto 64 secret keys. For example, you can refer the following secret keys:

```
[edit]
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 0 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 1 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key 2 secret
```

```
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key  3 secret
New secret (secret):
Retype new secret (secret):
security-administrator@host:fips#  prompt security authentication-key-chains key-chain
macsec-kc1 key  4 secret
New secret (secret):
Retype new secret (secret):
```

3. Associate the preshared keychain name with the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 pre-
shared-key-chain macsec-kc1
security-administrator@host:fips# set security macsec connectivity-association CA1 cipher-
suite gcm-aes-256
```

4. Set the trace option values.

```
[edit]
security-administrator@host:fips# set security macsec traceoptions file MACsec.log
security-administrator@host:fips# set security macsec traceoptions file size 4000000000
security-administrator@host:fips# set security macsec traceoptions flag all
```

5. Assign the trace to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions file mka_xe size 1g
security-administrator@host:fips# set security macsec interfaces interface-name
traceoptions flag all
```

6. Configure the MACsec security mode as static-cak for the connectivity association.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 security-mode static-cak
```

7. Set the MKA key server priority.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka key-server-priority 1
```

8. Set the MKA transmit interval.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 mka transmit-interval 3000
```

9. Enable the MKA secure.

```
[edit]
security-administrator@host:fips# set security macsec connectivity-association CA1 include-sci
```

10. Assign the connectivity association to an interface.

```
[edit]
security-administrator@host:fips# set security macsec interfaces interface-name connectivity-association CA1
```

11. Configure VLAN tagging.

```
[edit]
security-administrator@host:fips# set interfaces interface-name1 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name1 encapsulation flexible-ethernet-services
security-administrator@host:fips# set interfaces interface-name1 unit 100 encapsulation
```

```
vlan-bridge
security-administrator@host:fips# set interfaces interface-name1 unit 100 vlan-id 100
security-administrator@host:fips# set interfaces interface-name2 flexible-vlan-tagging
security-administrator@host:fips# set interfaces interface-name2 encapsulation flexible-
ethernet-services
security-administrator@host:fips# set interfaces interface-name2 unit 100 encapsulation
vlan-bridge
security-administrator@host:fips# set interfaces interface-name2 unit 100 vlan-id 100
```

Disable and Restart MACsec Sessions

To disable and restart the MACsec sessions use the following configurations:

- To disable the MACsec session:

  ```
  user@host# deactivate security macsec
  ```

- To restart the MACsec session:

  ```
  user@host# run restart dot1x-protocol
  ```

  or

  ```
  user@host# activate security macsec
  ```

Performing Self-Test

The cryptographic module enforces security rules to ensure that the Juniper Networks Junos OS Evolved in FIPS mode meets the security requirements of FIPS 140-3 Level 1. To validate the output of cryptographic algorithms approved for FIPS and test the integrity of some system modules, the device performs series of known answer test (KAT) self-tests. If a failure occurs, the system logs an error in the syslog messages and results in a FIPS error status. A device reboot may be required to recover the device.

You can view the self test details for MACsec library from syslog messages using the `file show /var/log/filename | match FIPS` command:

```
root@user:fips> file show /var/log/filename | match FIPS
Sep 19 11:39:04  host dot1xd[25429]: FIPS_KNOWN_ANSWER_TEST: root :AES128-CMAC Known Answer
Test: Passed
Sep 19 11:39:04  host dot1xd[25429]: FIPS_KNOWN_ANSWER_TEST: root :AES256-CMAC Known Answer
Test: Passed
Sep 19 11:39:04  host dot1xd[25429]: FIPS_KNOWN_ANSWER_TEST: root :AES-ECB Known Answer Test:
Passed
Sep 19 11:39:04  host dot1xd[25429]: FIPS_KNOWN_ANSWER_TEST: root :AES-KEYWRAP Known Answer
Test: Passed
Sep 19 11:39:04  host dot1xd[25429]: FIPS_KNOWN_ANSWER_TEST: root :KBKDF Known Answer Test:
Passed
Sep 19 11:39:04  host dot1xd[25429]: FIPS Known Answer Tests passed
```

You can view the self test details for MACsec chip using the `show trace application securityd | match KAT` command:

```
2024-09-19 10:51:43.018130572 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 0 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.018480633 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 0 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.018829898 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 1 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.019178702 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 1 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.019526878 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 2 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.019863898 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 2 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.020212185 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 3 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.020560356 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 3 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.020857951 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 4 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.021196366 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 4 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.021544409 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 5 AES-256-GCM KAT encryption passed."
```

```
2024-09-19 10:51:43.021863317 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 5 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.022211587 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 6 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.022566737 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 6 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.022915235 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 7 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.023263471 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 7 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.023611761 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 8 AES-256-GCM KAT encryption passed."
2024-09-19 10:51:43.023960636 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_pg_aes_kats_test: ASIC 0 PG 8 AES-256-GCM KAT decryption passed."
2024-09-19 10:51:43.023961748 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_aes_kats_test: ASIC 0 KAT test is complete, validation successful"
2024-09-19 10:51:43.023962866 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_fips_run_kat - KAT passed for asic 0"
2024-09-19 10:51:43.024006290 re0:securityd:11989:TRACE_INFO lltp_info  message = "MACSECV2_DRV:
macsecv2_drv_init - called macsecv2_drv_fips_run_kat, status 0"
2024-09-19 10:51:43.024020752 re0:securityd:11989:TRACE_INFO lltp_info  message = "virtual void
SecdPicPdBt::initializeHardwareBlock(uint32_t) - Checking KAT status on 0/0, fpc i2c d24, pic
i2c d21, block_num 0"
2024-09-19 10:51:43.024024112 re0:securityd:11989:TRACE_INFO lltp_info  message = "void
SecdPicPdBt::checkFIPSKAT(u_int8_t) - KAT passed for asic 0"
2024-09-19 10:51:43.024052157 re0:securityd:11989:TRACE_NOTICE lltp_notice  message =
"SECD_MACSEC_KAT_STATUS: /Chassis[0]/Fpc[0]/Pic[0], asic 0: succeeded"
```