HPE Juniper Networking

HPE Juniper Networking vSRX Virtual Firewall

FIPS 140-3 Non-Proprietary Security Policy

Version: Junos OS 24.4R1-S3.7

Prepared for:

HPE Juniper Networking,
1133 Innovation Way
Sunnyvale, California 94089
USA
408.745.2000
1.888 JUNIPER
www.juniper.net

Prepared by:

www.teronlabs.com

Table of Contents

1 General 6

1.1 Overview 6

1.2 Security Levels 6

1.3 Additional Information 6

2 Cryptographic Module Specification 7

2.1 Description 7

2.2 Tested and Vendor Affirmed Module Version and Identification 9

2.3 Excluded Components 9

2.4 Modes of Operation 10

2.5 Algorithms 11

2.6 Security Function Implementations 15

2.7 Algorithm Specific Information 21

2.8 RBG and Entropy 22

2.9 Key Generation 22

2.10 Key Establishment 22

2.11 Industry Protocols 23

3 Cryptographic Module Interfaces 24

3.1 Ports and Interfaces 24

4 Roles, Services, and Authentication 25

4.1 Authentication Methods 25

4.2 Roles 26

4.3 Approved Services 27

4.4 Non-Approved Services 33

4.5 External Software/Firmware Loaded 33

5 Software/Firmware Security 34

5.1 Integrity Techniques 34

5.2 Initiate on Demand 34

6 Operational Environment 34

6.1 Operational Environment Type and Requirements 34
6.2 Configuration Settings and Restrictions 34
7 Physical Security 35
8 Non-Invasive Security 35
9 Sensitive Security Parameters Management 36
9.1 Storage Areas 36
9.2 SSP Input-Output Methods 36
9.3 SSP Zeroization Methods 37
9.4 SSPs 39
9.5 Transitions 49
10 Self-Tests 50
10.1 Pre-Operational Self-Tests 50
10.2 Conditional Self-Tests 51
10.3 Periodic Self-Test Information 55
10.4 Error States 58
10.5 Operator Initiation of Self-Tests 58
11 Life-Cycle Assurance 59
11.1 Installation, Initialization, and Startup Procedures 59
11.1.1 Deploying the Module 59
11.1.2 Setting the Root Password 59
11.1.3 Installing the License 60
11.1.4 Enabling the Approved Mode 60
11.2 Administrator Guidance 61
11.3 Non-Administrator Guidance 62
11.4 Design and Rules 62
11.4.1 Module Design Rules 62
11.4.2 Module Operation Rules 62
11.5 Maintenance Requirements 62
11.6 End of Life 62
12 Mitigation of Other Attacks 63

Glossary and Abbreviations 64
References 65

List of Tables

Table 1: Security Levels 6
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 9
Table 3: Tested Operational Environments - Software, Firmware, Hybrid 9
Table 4: Modes List and Description 10
Table 5: Approved Algorithms - OpenSSL 1.1.1 Cryptographic Library 12
Table 6: Approved Algorithms - QuickSec 7 Cryptographic Library 13
Table 7: Approved Algorithms - Physical Entropy Source 13
Table 8: Approved Algorithms - Kernel Cryptographic Library 13
Table 9: Vendor-Affirmed Algorithms 13
Table 10: Security Function Implementations 20
Table 11: Entropy Certificates 22
Table 12: Entropy Sources 22
Table 13: Supported industry protocols. 23
Table 14: Ports and Interfaces 24
Table 15: Roles 26
Table 16: Approved Services 32
Table 17: Storage Areas 36
Table 18: SSP Input-Output Methods 36
Table 19: SSP Zeroization Methods 37
Table 20: SSP Table 1 44
Table 21: SSP Table 2 48
Table 22: Pre-Operational Self-Tests 50
Table 23: Conditional Self-Tests 54

Table 24: Pre-Operational Periodic Information 55
Table 25: Conditional Periodic Information 57
Table 26: Error States 58

List of Figures

Figure 1: Block diagram. 8

1 General

1.1 Overview

This is the non-proprietary Cryptographic Module Security Policy for the HPE Juniper Networking vSRX Virtual Firewall running Junos OS 24.4R1-S3.7, hereafter referred to as the module.

1.2 Security Levels

The module meets requirements applicable to Level 1 of FIPS 140-3. The table below shows the security levels claimed for each section of the security requirements.

Section	Title	Security Level
1	General	1
2	Cryptographic module specification	1
3	Cryptographic module interfaces	1
4	Roles, services, and authentication	1
5	Software/Firmware security	1
6	Operational environment	1
7	Physical security	N/A
8	Non-invasive security	N/A
9	Sensitive security parameter management	1
10	Self-tests	1
11	Life-cycle assurance	1
12	Mitigation of other attacks	N/A
	Overall Level	1

Table 1: Security Levels

1.3 Additional Information

The module offers the same functionality as Juniper's hardware SRX Series Services Gateways in a virtual machine (VM) format.

MNHA (multi-node high availability) mode is supported by the module when the `junos-ike` package is installed. This mode operates using the IKE and IPsec industry protocols using algorithms tested and validated in this evaluation.

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The module is a virtual firewall designed to provide secure connectivity and advanced security services in cloud, virtualized, and hybrid environments. It delivers comprehensive, scalable, and automated security and routing functions, protecting workloads and ensuring secure connectivity across diverse environments.

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary:

The cryptographic boundary of the module is defined as the Junos OS 24.4R1-S3.7 software running in the Approved mode and its allocated virtual resources within the VM, as shown in Figure 1. All cryptographic operations occur within this boundary.

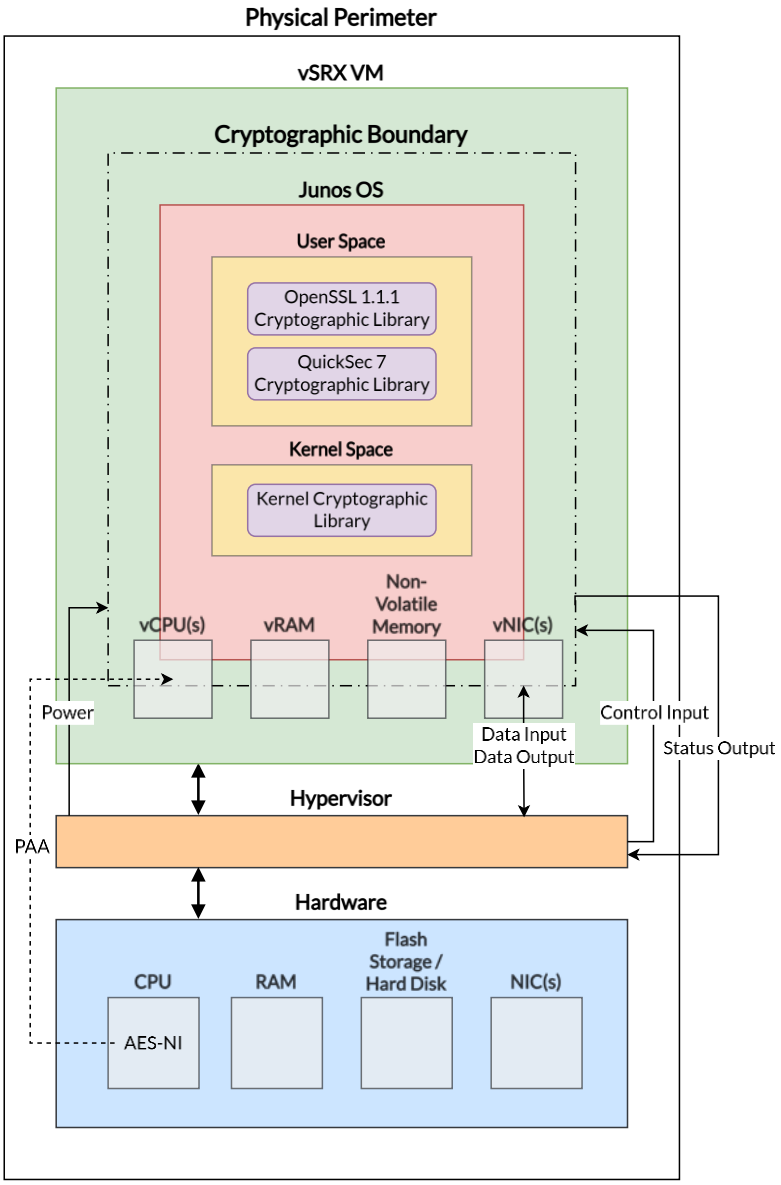

Figure 1: Block diagram.

Tested Operational Environment's Physical Perimeter (TOEPP):

The physical boundary of the module is defined as the HPE ProLiant DL380 Gen10 Plus Smart Choice server with an Intel Xeon Gold 6326 CPU running Ubuntu Server 22.04.3 LTS. This server hosts the KVM (Kernel-based VM) hypervisor, which runs the VM containing the module.

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets):

Package or File Name	Software/ Firmware Version	Features	Integrity Test
junos-vsrx3-x86-64-24.4R1-S3.7.qcow2	Junos vSRX 3, version 24.4R1-S3.7 (x86-64)	QCOW2 vSRX VM image downloaded from the Juniper software download site, containing Junos OS 24.4R1-S3.7 software. The Approved mode is enabled at runtime inside the VM.	Junos software integrity check executed at VM boot using ECDSA P-256 with SHA-256 (OpenSSL).

Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)

Tested Operational Environments - Software, Firmware, Hybrid:

The two tested OEs are Ubuntu Server 22.04.3 LTS on an HPE ProLiant DL380 Gen10 Plus Smart Choice server with an Intel Xeon Gold 6326 CPU with and without AES-NI. Details are provided below.

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Junos OS 24.4R1-S3.7	HPE ProLiant DL380 Gen10 Plus Smart Choice	Intel Xeon Gold 6326 CPU @ 2.90GHz	Yes	Ubuntu Server 22.04.3	Junos vSRX 3, version 24.4R1-S3.7 (x86-64)
Junos OS 24.4R1-S3.7	HPE ProLiant DL380 Gen10 Plus Smart Choice	Intel Xeon Gold 6326 CPU @ 2.90GHz	No	Ubuntu Server 22.04.3	Junos vSRX 3, version 24.4R1-S3.7 (x86-64)

Table 3: Tested Operational Environments - Software, Firmware, Hybrid

CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

2.3 Excluded Components

No components are excluded from the requirements of FIPS 140-3.

2.4 Modes of Operation

Modes List and Description:

The module is compliant with the FIPS 140-3 requirements when configured in the Approved mode according to the secure initialization instructions in Section 11.1.

Mode Name	Description	Type	Status Indicator
Approved	Approved mode enabled by following the configuration commands in Section 11.1	Approved	Presence of ':fips' suffix string in CLI prompt

Table 4: Modes List and Description

Failure to follow the secure initialization instructions results in the module being in a non-Approved mode, which is out of scope of the validation.

Mode Change Instructions and Status:

Not applicable.

Degraded Mode Description:

The module does not support a degraded mode.

2.5 Algorithms

Approved Algorithms:

Although the module may have been tested for additional algorithms or modes, only those listed below are utilized by the module.

OpenSSL 1.1.1 Cryptographic Library

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A7322	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CTR	A7322	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A7322	Direction - Decrypt, Encrypt IV Generation - External IV Generation Mode - 8.2.1 Key Length - 256	SP 800-38D
ECDSA KeyGen (FIPS186-5)	A7322	Curve - P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A7322	Curve - P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A7322	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512 Component - No	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A7322	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	FIPS 186-5
HMAC-SHA-1	A7322	Key Length - Key Length: 112-1024 Increment 8	FIPS 198-1
HMAC-SHA2-256	A7322	Key Length - Key Length: 112-1024 Increment 8	FIPS 198-1
HMAC-SHA2-512	A7322	Key Length - Key Length: 256-2048 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A7322	Domain Parameter Generation Methods - P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A7322	Domain Parameter Generation Methods - FC, MODP-2048 Scheme - dhEphem - KAS Role - initiator	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
KDF SSH (CVL)	A7322	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
RSA KeyGen (FIPS186-5)	A7322	Key Generation Mode - probable Modulo - 2048, 3072, 4096 Primality Tests - 2powSecStr Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A7322	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5	FIPS 186-5
RSA SigVer (FIPS186-5)	A7322	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5	FIPS 186-5
Safe Primes Key Generation	A7322	Safe Prime Groups - MODP-2048, MODP-3072, MODP-4096	SP 800-56A Rev. 3
Safe Primes Key Verification	A7322	Safe Prime Groups - MODP-2048, MODP-3072, MODP-4096	SP 800-56A Rev. 3
SHA-1	A7322	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A7322	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A7322	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A7322	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4

Table 5: Approved Algorithms - OpenSSL 1.1.1 Cryptographic Library

QuickSec 7 Cryptographic Library

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A7673	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A7673	Direction - Decrypt, Encrypt IV Generation - External IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
HMAC-SHA2-256	A7673	Key Length - Key Length: 112-1024 Increment 8	FIPS 198-1
HMAC-SHA2-384	A7673	Key Length - Key Length: 192-2048 Increment 8	FIPS 198-1
KDF IKEv1 (CVL)	A7673	Authentication Method - Digital Signature, Pre-shared Key Preshared Key Length - Preshared Key Length: 8-256 Increment 8 Diffie-Hellman Shared Secret Length - Diffie-	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
		Hellman Shared Secret Length: 256, 384, 2048 Hash Algorithm - SHA2-256, SHA2-384	
KDF IKEv2 (CVL)	A7673	Diffie-Hellman Shared Secret Length - Diffie-Hellman Shared Secret Length: 256, 384, 2048 Derived Keying Material Length - Derived Keying Material Length: 1136-2184 Increment 8 Hash Algorithm - SHA2-256, SHA2-384	SP 800-135 Rev. 1
SHA2-256	A7673	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4
SHA2-384	A7673	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4

Table 6: Approved Algorithms - QuickSec 7 Cryptographic Library

Physical Entropy Source

Algorithm	CAVP Cert	Properties	Reference
Conditioning Component AES-CBC-MAC SP800-90B	A2518	Key Length - 128	SP 800-90B

Table 7: Approved Algorithms - Physical Entropy Source

Kernel Cryptographic Library

Algorithm	CAVP Cert	Properties	Reference
HMAC DRBG	A7179	Prediction Resistance - No Mode - SHA2-512	SP 800-90A Rev. 1
HMAC-SHA2-512	A7179	Key Length - Key Length: 8-1024 Increment 8	FIPS 198-1
SHA2-512	A7179	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4

Table 8: Approved Algorithms - Kernel Cryptographic Library

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
CKG	Key type:Symmetric and Asymmetric	N/A	SP 800-133 Rev. 2 Section 4, Example 1: approved DRBG output to generate symmetric keys and seeds used for asymmetric key pair generation; Section 5.1: DRBG output used to generate private key values for digital signature key pairs in accordance with FIPS 186-5; and Section 6.2.1 and SP 800-56A: derivation of symmetric keys established via key-agreement schemes

Table 9: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

The module does not support any non-approved, allowed algorithms when the module is operating in the Approved mode.

Non-Approved, Allowed Algorithms with No Security Claimed:

The module does not support any non-approved, allowed algorithms with no security claimed when the module is operating in the Approved mode.

Non-Approved, Not Allowed Algorithms:

The module does not support any non-approved, not allowed algorithms when the module is operating in the Approved mode.

2.6 Security Function Implementations

The module implements the security functions listed in the following table.

Name	Type	Description	Properties	Algorithms
Enc/Dec (SSH)	BC-UnAuth	Unauthenticated encryption/decryption for SSH		AES-CTR: (A7322)
KAS-SSC (SSH)	KAS-SSC	Key agreement scheme shared secret computation for SSH		KAS-ECC-SSC Sp800-56Ar3: (A7322) KAS-FFC-SSC Sp800-56Ar3: (A7322)
ECDSA SigGen (SSH)	DigSig-SigGen	ECDSA signature generation for peer authentication in SSH		ECDSA SigGen (FIPS186-5): (A7322) SHA2-256: (A7322) SHA2-384: (A7322) SHA2-512: (A7322) HMAC DRBG: (A7179)
ECDSA SigVer (SSH)	DigSig-SigVer	ECDSA signature verification for peer authentication in SSH		ECDSA SigVer (FIPS186-5): (A7322) SHA2-256: (A7322) SHA2-384: (A7322) SHA2-512: (A7322)
MAC (SSH)	MAC	Message authentication for SSH		HMAC-SHA-1: (A7322) HMAC-SHA2-256: (A7322) HMAC-SHA2-512: (A7322)
KDF (SSH)	KAS-135KDF CKG	Key derivation function for SSH		CKG: () KDF SSH: (A7322) SHA-1: (A7322) SHA2-256:

Name	Type	Description	Properties	Algorithms
				(A7322) SHA2-384: (A7322) SHA2-512: (A7322)
DRBG (Kernel)	DRBG	Deterministic random bit generation within the kernel		HMAC DRBG: (A7179) HMAC-SHA2-512: (A7179) SHA2-512: (A7179)
ENT	ENT-ESV	Entropy source		Conditioning Component AES-CBC-MAC SP800-90B: (A2518)
ECDSA KeyGen (PKID)	AsymKeyPair-KeyGen CKG	Long-term ECDSA key generation used for SSH and IKE when authentication keys are internally generated		ECDSA KeyGen (FIPS186-5): (A7322) CKG: () HMAC DRBG: (A7179)
RSA KeyGen (PKID)	AsymKeyPair-KeyGen CKG	Long-term RSA key generation used for SSH and IKE when authentication keys are internally generated		RSA KeyGen (FIPS186-5): (A7322) CKG: () HMAC DRBG: (A7179)
RSA SigGen (SSH)	DigSig-SigGen	RSA signature generation for SSH		RSA SigGen (FIPS186-5): (A7322) SHA2-256: (A7322)
RSA SigVer (SSH)	DigSig-SigVer	RSA signature verification for SSH		RSA SigVer (FIPS186-5): (A7322) SHA2-256: (A7322)
Verify image	DigSig-SigVer	Verification of software image		ECDSA SigGen (FIPS186-5): (A7322) SHA2-256: (A7322)

Name	Type	Description	Properties	Algorithms
Full KAS (SSH)	AsymKeyPair-KeyGen CKG AsymKeyPair-KeyVer KAS-Full	Full key agreement for SSH	IG:IG D.F Scenario 2, path 2, split Key Confirmation:No Key derivation:KDA (separately tested) Caveat:Key establishment methodology provides between 128 and 256 bits of security strength	ECDSA KeyGen (FIPS186-5): (A7322) CKG: () HMAC DRBG: (A7179) ECDSA KeyVer (FIPS186-5): (A7322) KAS-ECC-SSC Sp800-56Ar3: (A7322) KDF SSH: (A7322) SHA-1: (A7322) SHA2-256: (A7322) SHA2-384: (A7322) SHA2-512: (A7322)
KAS-ECC KeyGen (SSH)	AsymKeyPair-KeyGen CKG AsymKeyPair-KeyVer	KAS-ECC key pair generation SSH		KAS-ECC-SSC Sp800-56Ar3: (A7322) CKG: () HMAC DRBG: (A7179)
KAS-FFC KeyGen (SSH)	AsymKeyPair-KeyGen CKG AsymKeyPair-KeyVer	KAS-FFC key pair generation for SSH		KAS-FFC-SSC Sp800-56Ar3: (A7322) CKG: () HMAC DRBG: (A7179) Safe Primes Key Generation: (A7322) Safe Primes Key Verification: (A7322)

Name	Type	Description	Properties	Algorithms
KTS (SSH)	KTS-Wrap	Key transport using SSH as per IG D.G provisions	Standard:SP 800-38F IG D.G:Key wrapping key Caveat:Key establishment methodology provides between 128 and 256 bits of encryption strength	AES-CBC: (A7322) AES-CTR: (A7322) HMAC-SHA-1: (A7322) HMAC-SHA2-256: (A7322) HMAC-SHA2-512: (A7322)
Enc/Dec (IPsec)	BC-Auth BC-UnAuth	Encryption/decryption of IP payloads in ESP		AES-GCM: (A7322) AES-CBC: (A7322)
MAC (IPsec)	MAC	Message authentication for ESP		HMAC-SHA-1: (A7322) HMAC-SHA2-256: (A7322)
KAS-SSC (IKE)	KAS-SSC	Shared secret computation for IKE key agreement		KAS-ECC-SSC Sp800-56Ar3: (A7322) KAS-FFC-SSC Sp800-56Ar3: (A7322)
KDF (IKE)	KAS-135KDF CKG	Keying material derivation for IKE key agreement		CKG: () KDF IKEv1: (A7673) KDF IKEv2: (A7673) SHA2-256: (A7673) SHA2-384: (A7673)
Enc/Dec (IKE)	BC-Auth BC-UnAuth	Encryption/decryption of IKE protocol messages within the IKE SA		AES-GCM: (A7673) AES-CBC: (A7673)
MAC (IKE)	MAC	Message authentication for IKE protocol messages		HMAC-SHA2-256: (A7673) HMAC-SHA2-384: (A7673)
Full KAS (IKE)	KAS-Full CKG	Full key agreement for IKE	IG:IG D.F Scenario 2, path	CKG: () HMAC DRBG:

Name	Type	Description	Properties	Algorithms
	AsymKeyPair-KeyVer KAS-Full		2, split Key Confirmation:No Key derivation:KDA (separately tested) Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	(A7179) KAS-ECC-SSC Sp800-56Ar3: (A7322) KAS-FFC-SSC Sp800-56Ar3: (A7322) KDF IKEv1: (A7673) KDF IKEv2: (A7673) SHA2-256: (A7673) SHA2-384: (A7673)
KAS-ECC KeyGen (IKE)	AsymKeyPair-KeyGen CKG	Generation of ECC ephemeral key pairs for IKE key agreement		CKG: () HMAC DRBG: (A7179) KAS-ECC-SSC Sp800-56Ar3: (A7322)
KAS-FFC KeyGen (IKE)	AsymKeyPair-KeyGen CKG	Generation of FFC ephemeral key pairs for IKE key agreement		Safe Primes Key Generation: (A7322) Safe Primes Key Verification: (A7322) CKG: () HMAC DRBG: (A7179) KAS-FFC-SSC Sp800-56Ar3: (A7322)
ECDSA SigGen (IKE)	DigSig-SigGen	ECDSA signature generation for IKE authentication		ECDSA SigGen (FIPS186-5): (A7322) SHA2-256: (A7322) SHA2-384: (A7322) HMAC DRBG: (A7179)

Name	Type	Description	Properties	Algorithms
ECDSA SigVer (IKE)	DigSig-SigVer	ECDSA signature verification for IKE authentication		ECDSA SigVer (FIPS186-5): (A7322) SHA2-256: (A7322) SHA2-384: (A7322)
RSA SigGen (IKE)	DigSig-SigGen	RSA signature generation for IKE authentication		RSA SigGen (FIPS186-5): (A7322) SHA2-256: (A7322)
RSA SigVer (IKE)	DigSig-SigVer	RSA signature verification for IKE authentication		RSA SigVer (FIPS186-5): (A7322) SHA2-256: (A7322)

Table 10: Security Function Implementations

2.7 Algorithm Specific Information

The module includes approved KDF algorithms for the SSH and IKE protocols. No parts of these protocols, other than the approved cryptographic algorithms and the KDFs, have been tested by the CAVP and CMVP.

The module utilizes AES-GCM for IKE (validated under QuickSec certificate A7673) and IPsec (validated under OpenSSL 1.1.1 certificate A7322) in compliance with IG C.H, Scenario 1 (IPsec-v3 protocol IV generation), per RFCs 4106, 5282, and 7296. Both AES-GCM implementations are validated using external IV generation with a deterministic construction as per SP 800-38D Section 8.2.1.

Two truncated HMAC algorithms are supported by the module for IPsec: HMAC-SHA-256-128 and HMAC-SHA1-96. As per IG C.D, the truncated forms of these HMACs are approved algorithms that can be used in the module's Approved mode because they meet the requirement specified in SP 800-107 Rev. 1 that the truncated output contains at least 32 leftmost bits. Both HMAC-SHA-256-128 (128 bits) and HMAC-SHA1-96 (96 bits) exceed this minimum requirement, with the underlying HMACs validated under OpenSSL 1.1.1 certificate A7322.

The module implements Diffie–Hellman key agreement schemes (KAS) in accordance with SP 800-56B Rev. 3 for the SSH and IKEv1/IKEv2 protocols, supporting both ECC and FFC methods. The implementations follow IG D.F Scenario 2, path (2) with split testing, where KAS components are tested separately. For SSH, shared secret computation (the algorithms KAS-ECC-SSC and KAS-FFC-SSC) and the KDF (the algorithm KDF SSH) are validated under OpenSSL certificate A7322. For IKEv1/IKEv2, the shared secret computation component (the algorithms KAS-ECC-SSC and KAS-FFC-SSC) is validated under A7322, while the KDFs (the algorithms KDF IKEv1 and KDF IKEv2) are validated under QuickSec certificate A7673. The module performs full key agreement and obtains all assurances required by Section 5.6.2 of SP 800-56A Rev. 3.

2.8 RBG and Entropy

Cert Number	Vendor Name
E305	Juniper Networks

Table 11: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Junos OS Physical Entropy Source – Intel Xeon Silver, Gold, W Series (Ice Lake-28) with FCLGA4189 Package	Physical	Intel Xeon Silver, Gold, W Series (Ice Lake-28) with FCLGA4189 Package	128 bits	128 bits	A2518 (AES-CBC-MAC)

Table 12: Entropy Sources

The module generates sufficient entropy for DRBG seeding and key generation. The entropy source generates 128 bits of entropy per 128-bit output block (full entropy). To meet the 384-bit entropy input required by SP 800-90A Rev. 1 for a 256-bit security level DRBG, the module collects 72 bytes from the Intel RDSEED entropy source ($72 \text{ bytes} \times 8 \text{ bits / byte} = 576 \text{ bits}$), corresponding to 4.5 blocks ($576 / 128 = 4.5$) of 128-bit entropy.

There are no initialization procedures required by the users of the module to operate the entropy source in a compliant manner. The module complies to the ESV Public Use document of the validated entropy certificate E305, which lists the Intel Xeon Gold 6326 CPU as an equivalent platform to the tested platform in the table above.

2.9 Key Generation

The module implements the key generation methods listed above in the Security Function Implementations table.

2.10 Key Establishment

The module implements the Diffie–Hellman key establishment methods listed above in the Security Function Implementations table.

2.11 Industry Protocols

The module supports the protocols listed below. No part of these protocols, other than the approved cryptographic algorithms and the KDFs, have been tested by the CAVP and CMVP. The SSH, IKE, and IPsec algorithms allow independent selection of key exchange, authentication, cipher, and integrity. In reference to the supported protocols table below, each column of options for a given protocol is independent and may be used in any viable combination.

Table 13: Supported industry protocols.

Protocol	Key Exchange	Auth	Cipher	Integrity
SSHv2	KAS-ECC (P-256, P-384, P-521) KAS-DH (MODP Group Exchange)	RSA-2048 ECDSA P-256 / P-384 / P-521	AES CTR 128/192/256	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-512
IKEv1 / IKEv2	KAS-DH (MODP 2048) KAS-ECC (P-256, P-384)	RSA-2048 / SHA-256 ECDSA P-256 / SHA-256 ECDSA P-384 / SHA-384	AES CBC 128/192/256 AES GCM 128/256	SHA-256 SHA-384
IPsec ESP	N/A	N/A	AES CBC 128/192/256 AES GCM 128/192/256	HMAC-SHA1-96 HMAC-SHA-256-128

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

The following table lists the logical interfaces implemented within the module as defined in the FIPS 140-3 standard. As the module is a virtual appliance, it does not include physical interfaces. It is designed to be deployed as a VM on a hypervisor and requires physical ports on the host. The module's logical interfaces are associated with these physical ports. The module does not implement a control output interface.

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input Data Output Control Input Status Output	Network traffic (e.g., IKE, IPsec) and management traffic (e.g., SSH) over vNICs.
N/A	Control Input Status Output	Management traffic (e.g., console input, command output) over virtual serial console and the out-of-band management network interface with interface name fxp0.
N/A	Power	No data passes. The VM is powered by the hosting server under control of the hypervisor.

Table 14: Ports and Interfaces

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module does not require or enforce any specific authentication method for accessing the Cryptographic Officer (CO) or User roles.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
User	Role	User	
Cryptographic Officer	Role	CO	

Table 15: Roles

The module supports two roles: CO and User. The module supports concurrent operators but does not support a maintenance role and/or bypass capability.

The CO role has permission to view and edit configuration and secrets within the module.

The User role only has permission to view the configuration.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Configure Security	Security-relevant configuration.	':fips' suffix in CLI prompt	CLI commands Configuration files	Success or error indication	ENT DRBG (Kernel) ECDSA KeyGen (PKID) RSA KeyGen (PKID)	Cryptographic Officer - HMAC DRBG V Value: E,Z - HMAC DRBG Key Value: E,Z - HMAC DRBG Entropy Input: E,Z - HMAC DRBG Seed: E,Z - SSH-Priv: G - SSH-Pub: G - Auth-User-Pub: W - Auth-CO-Pub: W - IKE-Priv: G - IKE-PSK: W
Configure	Non-security-relevant configuration.	N/A	CLI commands Configuration files	Success or error indication	None	Cryptographic Officer
Show Status	Displays the module's operational and security status.	N/A	CLI command	Operational and security status information	None	Cryptographic Officer User
Zeroize	Zeroize / destroy all CSPs.	N/A	CLI command	None (completion indicator is implicitly provided by the module restarting)	None	Cryptographic Officer - SSH-Priv: Z - SSH-Pub: Z - Auth-User-Pub: Z - Auth-CO-Pub: Z - Root-CA: Z - Package-CA:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Z - IKE-PSK: Z - IKE-Priv: Z - IKE-Pub: Z
SSH Connect	Initiates SSH connection for remote CLI monitoring and control.	':fips' suffix in CLI prompt	CLI command Authentication data	SSH session Success or error indication	ENT DRBG (Kernel) KAS-SSC (SSH) MAC (SSH) KDF (SSH) ECDSA SigGen (SSH) RSA SigGen (SSH) ECDSA SigVer (SSH) RSA SigVer (SSH) Full KAS (SSH) KAS-ECC KeyGen (SSH) KAS-FFC KeyGen (SSH) Enc/Dec (SSH) KTS (SSH)	Cryptographic Officer - HMAC DRBG V Value: E,Z - HMAC DRBG Key Value: E,Z - HMAC DRBG Entropy Input: E,Z - HMAC DRBG Seed: E,Z - SSH-DH-Shared-Secret: G,E,Z - SSH-DH-Priv: G,E,Z - SSH-SEKs: G,E,Z - Auth-CO-Pub: E - SSH-Priv: E - SSH-DH-Pub (self): G,E,Z - SSH-DH-Pub (peer): E,Z User - HMAC DRBG V Value: E,Z - HMAC DRBG Key Value: E,Z - HMAC DRBG Entropy Input: E,Z - HMAC DRBG Seed:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						E,Z - SSH-DH-Shared-Secret: G,E,Z - SSH-DH-Priv: G,E,Z - SSH-SEKs: G,E,Z - Auth-User-Pub: E - SSH-Priv: E - SSH-DH-Pub (self): G,E,Z - SSH-DH-Pub (peer): E,Z
IPsec Connect	Initiates and negotiates IKE SAs and CHILD SAs and encrypts/decrypts IPsec ESP-protected traffic.	':fips' suffix in CLI prompt	CLI commands Authentication data	IPsec session Success or error indication	ENT DRBG (Kernel) KAS-SSC (IKE) KAS-ECC KeyGen (IKE) KAS-FFC KeyGen (IKE) ECDSA KeyGen (PKID) RSA KeyGen (PKID) KDF (IKE) Enc/Dec (IKE) MAC (IKE) ECDSA SigGen (IKE)	Cryptographic Officer - HMAC DRBG V Value: E,Z - HMAC DRBG Key Value: E,Z - HMAC DRBG Entropy Input: E,Z - HMAC DRBG Seed: E,Z - AES-GCM IV: G,E,Z - IKE-Priv: E - IKE-Pub: G,E - IKE-DH-Priv: G,E,Z - IKE-DH-Shared-Secret: G,E,Z - IKE-SKEYSEED: G,E,Z - IKE-SEKs: G,E,Z - ESP-SEKs:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					RSA SigGen (IKE) ECDSA SigVer (IKE) RSA SigVer (IKE) Full KAS (IKE) Enc/Dec (IPsec) MAC (IPsec)	G,E,Z - IKE-PSK: W,E - IKE-DH-Pub (self): G,E,Z - IKE-DH-Pub (peer): W,E User - HMAC DRBG V Value: E,Z - HMAC DRBG Key Value: E,Z - HMAC DRBG Entropy Input: E,Z - HMAC DRBG Seed: E,Z - AES-GCM IV: G,E,Z - IKE-Priv: E - IKE-Pub: G,E - IKE-DH-Priv: G,E,Z - IKE-DH-Shared-Secret: G,E,Z - IKE-SKEYSEED: G,E,Z - IKE-SEKs: G,E,Z - ESP-SEKs: G,E,Z - IKE-PSK: E - IKE-DH-Pub (self): G,E,Z - IKE-DH-Pub (peer): W,E
Console Access	Access the CLI via virtual console for	None	CLI commands	Interactive console session	None	

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	monitoring and control.		Authentication data	Success or error indication		
Module Reset	Restart the module via Junos CLI command, clearing volatile state.	None	CLI command	System restart initiated Success or error indication	None	Cryptographic Officer - HMAC DRBG V Value: Z - HMAC DRBG Key Value: Z - HMAC DRBG Entropy Input: Z - HMAC DRBG Seed: Z - SSH-DH-Shared-Secret: Z - SSH-DH-Priv: Z - SSH-SEKs: Z - SSH-DH-Pub (self): Z - SSH-DH-Pub (peer): Z
VM Reset	Restart the module from the host, clearing volatile state.	None	Host or hypervisor request to restart the VM.	Virtual machine restart initiated Success or error indication	None	Unauthenticated - HMAC DRBG V Value: Z - HMAC DRBG Key Value: Z - HMAC DRBG Entropy Input: Z - HMAC DRBG Seed: Z - SSH-DH-Shared-Secret: Z - SSH-SEKs: Z - SSH-DH-Pub

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						(self): Z - SSH-DH-Pub (peer): Z
Load Image	Load a Junos firmware image into the module.	':fips' suffix in CLI prompt	CLI command Firmware image file	Firmware image verified and loaded Success or error indication	Verify image	Cryptographic Officer - Root-CA: E - Package-CA: E
Perform Self-Tests	Run module self-tests on demand via CLI command or restart.	None	CLI command Module or VM reset	Self-test results Success or error indicator		Cryptographic Officer User Unauthenticated
Show Version	Display the firmware version.	None	CLI command	Software and package version information	None	Cryptographic Officer User

Table 16: Approved Services

4.4 Non-Approved Services

There are no non-approved services when the module is operating in the Approved mode.

4.5 External Software/Firmware Loaded

The module includes a firmware load service to support necessary updates. Only the CO can install the new image using the CLI as described in Section 11.1. The loaded firmware is a complete image replacement and constitutes an entirely new module and version of Junos OS which would require a separate FIPS 140-3 validation.

5 Software/Firmware Security

5.1 Integrity Techniques

The module implements an approved firmware integrity self-test that uses ECDSA P-256 with SHA-256 to ensure the integrity of all Junos OS firmware components. The self-test is automatically run on initialization of the module. When the integrity check fails, the module enters an error state (kernel panic) which can only be exited by restarting the module.

5.2 Initiate on Demand

The module operator can run the test on demand by restarting the module.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The module is software-only and runs in a virtualized, modifiable OE deployed as a VM from a QCOW2 hypervisor base image. Specifically, the OE is a KVM hypervisor running on Ubuntu Server 22.04.3 LTS. The KVM hypervisor provides virtualization of CPU, memory, and I/O resources.

The module has full control over its own SSPs, which are stored and processed exclusively within the VM. The KVM hypervisor uses SELinux and sVirt to enforce process isolation and resource separation between VMs. This prevents access to SSPs by other processes or VMs running in the OE.

Each Junos OS image installed on the module is a complete software replacement. A new Junos OS release would therefore represent a new version of the module and require separate FIPS 140-3 validation.

6.2 Configuration Settings and Restrictions

There are no security rules, settings, or restrictions to the configuration of the operational environment beyond the initialization instructions required to set the module in the Approved mode.

7 Physical Security

This section is not applicable, as the module is software only.

8 Non-Invasive Security

This section is not applicable, as there are currently no approved non-invasive mitigation techniques specified in ISO/IEC 19790:2012.

9 Sensitive Security Parameters Management

9.1 Storage Areas

The table below lists the areas within the module's cryptographic boundary where SSPs can be stored.

Storage Area Name	Description	Persistence Type
vRAM	Volatile memory allocated to the VM by the hypervisor.	Dynamic
Non-Volatile Memory	Persistent virtual storage provided by the hypervisor. This can be either flash storage or hard disk.	Static

Table 17: Storage Areas

9.2 SSP Input-Output Methods

The table below lists the method used by the module for the input and output of SSPs.

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
Entry via SSH	Remote CO	vRAM	Encrypted	Automated	Electronic	KTS (SSH)
Entry via console	Local CO	vRAM	Plaintext	Manual	Electronic	
Output via SSH	vRAM	Remote CO	Encrypted	Automated	Electronic	KTS (SSH)
Output via console	vRAM	Local CO	Plaintext	Manual	Direct	
Entry as part of KAS (SSH)	Remote peer	vRAM	Plaintext	Automated	Electronic	KAS-SSC (SSH)
Output as part of KAS (SSH)	vRAM	Remote peer	Plaintext	Automated	Electronic	KAS-SSC (SSH)
Entry as part of KAS (IKE)	Remote peer	vRAM	Plaintext	Automated	Electronic	KAS-SSC (IKE)
Output as part of KAS (IKE)	vRAM	Remote peer	Plaintext	Automated	Electronic	KAS-SSC (IKE)
Pre-loaded	Manufacturer	Non-Volatile Memory	Plaintext	Manual	Direct	

Table 18: SSP Input-Output Methods

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Reset	Zeroization of SSPs in memory via invocation of a local or remote reset command.	RAM is volatile, so all data is lost when the VM instance is reset or powered off. Zeroization is effectively instantaneous.	Yes, both User and CO, via invocation of local or remote reset command.
Zeroize CLI command	Erases all SSPs and configuration, overwrites the virtual disk, and restores the VM to its factory-default firmware image.	This command clears all data from memory and persistent storage, and forces a restart.	Yes, CO via invocation of zeroize CLI command.
Explicit zeroize function	Zeroization of SSPs in memory when no longer needed.	Explicit zeroization function destroys SSP information immediately by overwriting the virtual memory area with zeroes.	No. The operator cannot directly initiate this method.

Table 19: SSP Zeroization Methods

The CO can run the following command from the Junos CLI to zeroize SSPs on the module:

```
crypto-officer:fips> request system zeroize
```

This wipes all SSPs, configuration, and user data, resets the configuration to the factory defaults, and restarts the module. The CO must follow the instructions in Section 11.1 to reenable the Approved mode.

Use of the zeroize command is restricted to the CO. The CO shall perform zeroization in the following situations:

1. **Before entering the Approved mode**: The module must be prepared by erasing all SSPs and other user-created data before operation as a FIPS cryptographic module in the Approved mode.
2. **Before exiting the Approved mode**: All SSPs and other user-created data must be erased.

The CO must retain control of the module while the zeroization is in process.

Zeroization and restarting the module using the command above is initiated by the operator. The Junos OS automatically zeroizes SSPs when no longer required using explicit delete commands as part of its memory management. Session termination is initiated by the operator or by environmental errors.

The completion of zeroization is indicated implicitly. If the zeroization is initiated using an explicit zeroize function invoked via a command, then completion of the command indicates that

zeroization was successful. If zeroization is initiated by executing the `request system zeroize` zeroization command and restarting the module, then successful reinitialization of the module indicates that zeroization was successful. In the case of zeroization initiated by session termination, the SSPs are zeroized using an explicit zeroize function when the session terminates, and session termination is indicated in the system logs.

9.4 SSPs

All SSPs used by the module are described in this section.

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
HMAC DRBG V Value	A critical value of the internal state of DRBG per IG D.L	256 - 256	DRBG internal state - CSP	DRBG (Kernel)		DRBG (Kernel)
HMAC DRBG Key Value	A critical value of the internal state of DRBG per IG D.L	256 - 256	DRBG internal state - CSP	DRBG (Kernel)		DRBG (Kernel)
HMAC DRBG Entropy Input	A critical value of the internal state of DRBG provided by entropy source	256 - 256	Entropy source output - CSP	ENT		DRBG (Kernel)
HMAC DRBG Seed	Seed material used to seed or reseed the HMAC DRBG	256 - 256	DRBG internal state - CSP	ENT		DRBG (Kernel)
SSH-DH-Shared-Secret	Shared DH value computed from the ephemeral DH key-pairs as part of SSH and used to derive session keys. P-256, P-384 and P-521	256, 384, 521 - 128, 192, 256	DH shared value - CSP		KAS-SSC (SSH)	KDF (SSH)
SSH-Priv	SSH host authentication key (ECDSA or RSA)	2048, 256, 4096, 384, 521 - 112, 128, 152, 192, 256	Asymmetric private key - CSP	ECDSA KeyGen (PKID) RSA KeyGen (PKID)		ECDSA SigGen (SSH) RSA SigGen (SSH)
SSH-DH-Priv	SSH DH private key used in SSH. P-256, P-384 and P-521	256, 384, 521 - 128, 192, 256	Asymmetric private key - CSP	KAS-ECC KeyGen (SSH)		KAS-SSC (SSH)
SSH-SEKs	Session encryption/decryption and integrity keys used with SSHv2.	128, 192, 256 bits (AES-CTR) 160, 256, 512 bits (HMAC) - 128, 192,	Symmetric Key - CSP		KDF (SSH)	Enc/Dec (SSH) KTS (SSH) MAC (SSH)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		256 bits (AES-CTR) 128, 256, 256 bits (HMAC)				
SSH-Pub	SSH Public Host Key	2048, 256, 4096, 384, 521 - 112,128, 152, 192, 256	Asymmetric Key - PSP	ECDSA KeyGen (PKID) RSA KeyGen (PKID)		ECDSA SigVer (SSH) RSA SigVer (SSH)
Auth-CO-Pub	SSH CO authentication public key.	2048, 256, 4096, 384, 521 - 112, 128, 152, 192, 256	Asymmetric Key - PSP			ECDSA SigVer (SSH) RSA SigVer (SSH)
Auth-User-Pub	SSH User authentication public key	2048, 256, 4096, 384, 521 - 112, 128, 152, 192, 256	Asymmetric Key - PSP			ECDSA SigVer (SSH) RSA SigVer (SSH)
Root-CA	JuniperRootCA. Used to verify the validity of the PackageCA	256 - 128	Asymmetric Key - PSP			Verify image
Package-CA	Certificate that holds the public key of the signing key that was used to generate all the signatures used on the packages and signatures lists.	256 - 128	Asymmetric Key - PSP			Verify image
SSH-DH-Pub (self)	SSH DH public key used for key establishment. P-256, P-384 and P-521	256, 384, 521 - 128, 192, 256	Asymmetric Key - PSP	KAS-ECC KeyGen (SSH)		KAS-SSC (SSH)
SSH-DH-Pub (peer)	SSH DH public keys provided by protocol peer device and used	256, 384, 521 - 128, 192, 256	Asymmetric Key - PSP			KAS-SSC (SSH)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	with SSH for key establishment. P-256, P-384 and P-521.					
IKE-PSK	Pre-shared key used for IKE authentication (if used). Configured by the CO.	Depends on PSK length and randomness. ≥128 bits of entropy recommended (e.g., 32 hex chars or 20 random ASCII chars). - Equal to entropy (≥128-bit security recommended).	Pre-Shared Key - CSP			KAS-SSC (IKE)
IKE-Priv	IKE authentication private key used in IKE_AUTH	2048 bits (RSA-2048) 4096 bits (RSA-4096) 256 bits (ECDSA P-256) 384 bits (ECDSA P-384) - 112 bits (RSA-2048) 152 bits (RSA-4096) 128 bits (ECDSA P-256) 192 bits (ECDSA P-384)	Private Key - CSP	ECDSA KeyGen (PKID) RSA KeyGen (PKID)		ECDSA SigGen (IKE) RSA SigGen (IKE)
IKE-Pub	IKE authentication public key used in IKE_AUTH.	ECDSA: 256, 384 RSA: 2048,	Public Key - PSP		ECDSA KeyGen (PKID)	

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		4096 - ECDSA: 128, 192 RSA: 112, 152			RSA KeyGen (PKID)	
IKE-SEKs	Symmetric keys protecting the IKE SA, including IKE encryption key and IKE integrity (MAC). Derived from the SKEYID/SKEYSEED using the IKE KDF.	128, 192, 256 bits (AES-CBC) 128, 256 bits (AES-GCM) 256, 384 bits (SHA) - 128, 192, 256 bits (AES-CBC) 128, 256 bits (AES-GCM) 128 bits (SHA-256) 192 bits (SHA-384)	Symmetric Key - CSP	KDF (IKE)	KDF (IKE)	Enc/Dec (IKE) MAC (IKE)
IKE-SKEYSEED	Keying material derived from IKE Diffie-Hellman/ECDH shared secret and nonces using the IKE KDF. Used to derive IKE SA keys (IKE-SEKs) and IPsec ESP session keys (ESP-SEKs).	128, 192, 256 bits (AES-CBC) 128, 256 bits (AES-GCM) 256, 384 bits (SHA) - 128, 192, 256 bits (AES-CBC) 128, 256 bits (AES-GCM) 128 bits (SHA-256) 192 bits (SHA-384)	Symmetric Key - CSP	KDF (IKE)	KDF (IKE)	KDF (IKE)
IKE-DH-Priv	Ephemeral IKE DH/ECDH private integer used to compute IKE DH/ECDH shared secret.	2048 bits (DH Group 14) 256 bits (ECDH Group 19) 384 bits (ECDH Group 20) -	Private Key - CSP	KAS-SSC (IKE)	KAS-SSC (IKE)	KAS-SSC (IKE)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		112 bits (DH Group 14) 128 bits (ECDH Group 19) 192 bits (ECDH Group 20)				
IKE-DH-Pub (self)	Local ephemeral DH/ECDH public key sent to the peer during IKE key establishment.	2048 bits (DH Group 14) 256 bits (ECDH Group 19) 384 bits (ECDH Group 20) - N/A	Public Key - PSP	KAS-ECC KeyGen (IKE) KAS-FFC KeyGen (IKE)		KAS-SSC (IKE)
IKE-DH-Pub (peer)	Peer DH/ECDH public value received during IKE key agreement.	2048 bits (DH Group 14) 256 bits (ECDH Group 19) 384 bits (ECDH Group 20) - N/A	Public Key - PSP			KAS-SSC (IKE)
IKE-DH-Shared-Secret	Shared DH/ECDH secret computed during IKE_SA_INIT.	2048 bits (DH Group 14) 256 bits (ECDH Group 19) 384 bits (ECDH Group 20) - 112 bits (DH Group 14) 128 bits (ECDH Group 19) 192 bits	DH Shared Value - CSP		KAS-SSC (IKE)	KDF (IKE)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		(ECDH Group 20)				
ESP-SEKs	SA ESP encryption and integrity keys protecting IPsec payloads. Derived from SKEYID/SKEYSEED using the IKE KDF.	128, 192, 256 bits (AES-CBC) 128, 192, 256 bits (AES-GCM) 160 bits (HMAC-SHA1-96) 256 bits (HMAC-SHA-256-128) - 128, 192, 256 bits (AES-CBC) 128, 192, 256 bits (AES-GCM) 80 bits (HMAC-SHA1-96) 128 bits (HMAC-SHA-256-128)	Symmetric Key - CSP	KDF (IKE)		Enc/Dec (IPsec) MAC (IPsec)
AES-GCM IV	Initialization Vector (IV) used for AES encryption in Galois/Counter Mode (GCM) to ensure unique ciphertexts for each encryption operation. A new IV is generated for each AES-GCM operation.	-	- PSP			Enc/Dec (SSH) Enc/Dec (IPsec) Enc/Dec (IKE)

Table 20: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
HMAC DRBG V Value		vRAM:Plaintext	Until updated by HMAC_DRBG_Update()	Reset	
HMAC DRBG Key Value		vRAM:Plaintext	Until updated by HMAC_DRBG_Update()	Reset	
HMAC DRBG Entropy Input		vRAM:Plaintext	Until HMAC_Instantiate_Update() or HMAC_DRBG_Reseed() complete	Reset Explicit zeroize function	
HMAC DRBG Seed		vRAM:Plaintext	Until HMAC_Instantiate_Update() or HMAC_DRBG_Reseed() complete	Reset Explicit zeroize function	
SSH-DH-Shared-Secret		vRAM:Plaintext	Until SSH session termination	Reset Explicit zeroize function	
SSH-Priv		vRAM:Plaintext Non-Volatile Memory:Plaintext	Until SSH session termination	Reset Zeroize CLI command Explicit zeroize function	SSH-Pub:Paired With
SSH-DH-Priv		vRAM:Plaintext	Until SSH session termination	Reset Explicit zeroize function	SSH-DH-Pub (self):Paired With
SSH-SEKs		vRAM:Plaintext	Until SSH session termination	Reset Explicit zeroize function	
SSH-Pub	Output via SSH Output via console	Non-Volatile Memory:Plaintext		Zeroize CLI command	SSH-Priv:Paired With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
	Output as part of KAS (SSH)				
Auth-CO-Pub	Entry via SSH Entry via console	Non-Volatile Memory:Plaintext		Zeroize CLI command	
Auth-User-Pub	Entry via SSH Entry via console	Non-Volatile Memory:Plaintext		Zeroize CLI command	
Root-CA	Pre-loaded	Non-Volatile Memory:Plaintext		Zeroize CLI command	
Package-CA	Pre-loaded	Non-Volatile Memory:Plaintext		Zeroize CLI command	
SSH-DH-Pub (self)	Output as part of KAS (SSH)	vRAM:Plaintext	Until SSH session termination	Reset Explicit zeroize function	SSH-DH-Priv:Paired With
SSH-DH-Pub (peer)	Entry as part of KAS (SSH)	vRAM:Plaintext	Until SSH session termination	Reset Explicit zeroize function	
IKE-PSK	Entry via SSH Entry via console	vRAM:Plaintext Non-Volatile Memory:Obfuscated	Until IKE_SA authentication completes or terminates	Reset Explicit zeroize function Zeroize CLI command	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
IKE-Priv	Entry via SSH Entry via console	Non-Volatile Memory:Plaintext		Zeroize CLI command	
IKE-Pub	Entry via SSH Entry via console	Non-Volatile Memory:Plaintext		Zeroize CLI command	
IKE-SEKs		vRAM:Plaintext	Until IKE SA lifetime ends	Reset Explicit zeroize function	
IKE-SKEYSEED		vRAM:Plaintext	Until IKE SA lifetime ends	Reset Explicit zeroize function	
IKE-DH-Priv		vRAM:Plaintext	Until DH computation completes and IKE SA is established	Reset Explicit zeroize function	
IKE-DH-Pub (self)	Output as part of KAS (IKE)	vRAM:Plaintext	Until IKE SA is established	Reset Explicit zeroize function	
IKE-DH-Pub (peer)	Entry as part of KAS (IKE)	vRAM:Plaintext	Until DH computation completes and IKE SA is established	Reset Explicit zeroize function	
IKE-DH-Shared-Secret		vRAM:Plaintext	Until the IKE SKEYSEED value is derived using the PRF	Reset Explicit zeroize function	IKE-DH-Priv:Derived From IKE-DH-Pub (peer):Derived From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					IKE-SEKs:Wraps
ESP-SEKs		vRAM:Plaintext	Until IKE SA lifetime ends or SA is rekeyed	Reset Explicit zeroize function	
AES-GCM IV		vRAM:Plaintext		Explicit zeroize function	

Table 21: SSP Table 2

9.5 Transitions

The following transitions apply to the module and will be non-approved for cryptographic protection purposes after December 31, 2030:

- All use of the SHA-1 hash algorithm
- SSPs with less than 128-bit security strength

10 Self-Tests

On startup or restart, the module performs the pre-operational self-tests and the indicated conditional cryptographic algorithm self-tests described below. All KATs (Known Answer Tests) must be completed successfully prior to the use of cryptography by the module. The algorithms utilized in the pre-operational firmware integrity test must pass their own CASTs (Conditional Algorithm Self-Tests) prior to the integrity test.

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
Firmware Integrity check	ECDSA P-256 with SHA-256	KAT	SW/FW Integrity	PASS/FAIL console output	ECDSA Verify

Table 22: Pre-Operational Self-Tests

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
Entropy Source (start-up)	N/A	APT, RCT	CAST	Console output / output of entropy source	Start-up	On VM initialization
Entropy Source (continuous)	N/A	APT, RCT	CAST	Console output / output of entropy source	Continuous	On VM initialization
AES-CBC (A7322) Encrypt	Key sizes: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Encrypt	On VM initialization
AES-CBC (A7322) Decrypt	Key sizes: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Decrypt	On VM initialization
AES-GCM (A7322) Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Encrypt	On VM initialization
AES-GCM (A7322) Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Decrypt	On VM initialization
HMAC DRBG (A7179)	SHA-512	KAT	CAST	PASS/FAIL console output	Instantiate, reseed, and generate	On VM initialization
KAS-ECC-SSC Sp800-56Ar3 (A7322)	P-256, P-384, P-521	KAT	CAST	PASS/FAIL console output	Derivation of the expected shared secret	On VM initialization
KAS-FFC-SSC Sp800-56Ar3 (A7322)	MODP-2048	KAT	CAST	PASS/FAIL console output	Derivation of the expected shared secret	On VM initialization
ECDSA SigGen (FIPS186-5) (A7322)	P-256, P-384, P-521	KAT	CAST	PASS/FAIL console output	Sign	On VM initialization
ECDSA SigVer	P-256, P-384, P-521	KAT	CAST	PASS/FAIL console output	Verify	On VM initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
(FIPS186-5) (A7322)						
HMAC-SHA-1 (A7322)	MAC length: 160 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
HMAC-SHA2-256 (A7322)	MAC length: 256 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
HMAC-SHA2-384 (A7322)	MAC length: 384 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
HMAC-SHA2-256 (A7673)	MAC length: 256 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
HMAC-SHA2-384 (A7673)	MAC length: 384 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
HMAC-SHA2-512 (A7322)	MAC length: 512 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
RSA SigGen (FIPS186-5) (A7322)	N/A	KAT	CAST	PASS/FAIL console output	Sign	On VM initialization
RSA SigVer (FIPS186-5) (A7322)	N/A	KAT	CAST	PASS/FAIL console output	Verify	On VM initialization
KDF SSH (A7322)	SHA-256	KAT	CAST	PASS/FAIL console output	Key derivation	On VM initialization
Conditioning Component AES-CBC-MAC SP800-90B (A2518)	Key size: 128 bits	KAT	CAST	PASS/FAIL console output	Encrypt only (used for conditioning the entropy source)	On VM initialization
SHA-1 (A7179)	N/A	KAT	CAST	PASS/FAIL console output	Hash	On VM initialization
SHA2-384 (A7179)	N/A	KAT	CAST	PASS/FAIL console output	Hash	On VM initialization
SHA2-512 (A7179)	N/A	KAT	CAST	PASS/FAIL console output	Hash	On VM initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC-SHA2-256 (A7179)	MAC length: 256 bits	KAT	CAST	PASS/FAIL console output	MAC	On VM initialization
ECDSA KeyGen (FIPS186-5) (A7322)	P-256, P-384, P-521	PCT	PCT	Returned key/transition soft error state	Generation and Verification of ECDSA signature	On key generation
RSA KeyGen (FIPS186-5) (A7322)	RSA 2048, RSA 3072, RSA 4096	PCT	PCT	Returned key/transition soft error state	Generation and Verification of signature	On key generation
Safe Primes Key Generation	Safe Prime Groups: MODP-2048, MODP-3072, MODP-4096	PCT	PCT	Returned key/transition soft error state	Recompute and compare	On key generation
FW load	ECDSA P-256 with SHA-256	KAT	SW/FW Load	PASS/FAIL console output	Verification of ECDSA signature on FW	On FW load
ECDSA KeyVer (FIPS186-5) (A7322)	P-256, P-384, P-521	PCT	PCT	Returned key/transition critical error state	Generation and Verification of ECDSA signature	On key generation
AES-GCM (A7673) Encrypt	Key size: 128, 256 bits	KAT	CAST	PASS/FAIL console output	Encrypt	On VM initialization
AES-GCM (A7673) Decrypt	Key size: 128, 256 bits	KAT	CAST	PASS/FAIL console output	Decrypt	On VM initialization
AES-CBC (A7673) Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Encrypt	On VM initialization
AES-CBC (A7673) Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	PASS/FAIL console output	Decrypt	On VM initialization
KDF IKEv1 (A7673)	SHA-256, SHA-384	KAT	CAST	PASS/FAIL console output	Key derivation	On VM initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF IKEv2 (A7673)	SHA-256, SHA-384	KAT	CAST	PASS/FAIL console output	Key derivation	On VM initialization

Table 23: Conditional Self-Tests

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
Firmware Integrity check	KAT	SW/FW Integrity	On demand	Manually

Table 24: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
Entropy Source (start-up)	APT, RCT	CAST	On demand	Manually
Entropy Source (continuous)	APT, RCT	CAST	On VM initialization	Manually
AES-CBC (A7322) Encrypt	KAT	CAST	On demand	Manually
AES-CBC (A7322) Decrypt	KAT	CAST	On demand	Manually
AES-GCM (A7322) Encrypt	KAT	CAST	On demand	Manually
AES-GCM (A7322) Decrypt	KAT	CAST	On demand	Manually
HMAC DRBG (A7179)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7322)	KAT	CAST	On demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A7322)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A7322)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A7322)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A7322)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A7322)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A7322)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A7673)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A7673)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-512 (A7322)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A7322)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A7322)	KAT	CAST	On demand	Manually
KDF SSH (A7322)	KAT	CAST	On demand	Manually
Conditioning Component AES-CBC-MAC SP800-90B (A2518)	KAT	CAST	On VM initialization	Manually
SHA-1 (A7179)	KAT	CAST	On VM initialization	Manually
SHA2-384 (A7179)	KAT	CAST	On VM initialization	Manually
SHA2-512 (A7179)	KAT	CAST	On VM initialization	Manually
HMAC-SHA2-256 (A7179)	KAT	CAST	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A7322)	PCT	PCT	On condition trigger	Automatic
RSA KeyGen (FIPS186-5) (A7322)	PCT	PCT	On condition trigger	Automatic
Safe Primes Key Generation	PCT	PCT	On condition trigger	Automatic
FW load	KAT	SW/FW Load	On FW load request	Automatic
ECDSA KeyVer (FIPS186-5) (A7322)	PCT	PCT	On condition trigger	Automatic
AES-GCM (A7673) Encrypt	KAT	CAST	On VM initialization	Manually
AES-GCM (A7673) Decrypt	KAT	CAST	On VM initialization	Manually
AES-CBC (A7673) Encrypt	KAT	CAST	On VM initialization	Manually
AES-CBC (A7673) Decrypt	KAT	CAST	On VM initialization	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF IKEv1 (A7673)	KAT	CAST	On demand	Manually
KDF IKEv2 (A7673)	KAT	CAST	On demand	Manually

Table 25: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Critical Failure State	The cryptographic module ceases to perform cryptographic operations, inhibits all data output, and provides status of the error via syslog messages and console status output	On self-test error	VM reset	Console status output
Soft Error State	A non-critical self-test failure occurs, causing a failure of the triggering operation	Firmware load test, continuous entropy health test failure, PCT error	The module processes the error, and resumes normal operation	Console displays error

Table 26: Error States

The module enters the critical error state on failure of any self-test, causing the kernel to panic and execution to halt. The only way to exit from this state is to restart the module, which causes the module to run the self-tests again. The module then re-enters the critical error state if the self-tests do not pass. If all self-tests pass, the module leaves the critical error state and returns to normal operation.

10.5 Operator Initiation of Self-Tests

Self-tests that are performed at module initialization are available on demand by restarting the module.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The module must be correctly installed and configured following the instructions in this section to enter a FIPS-compliant state and operate in the Approved mode.

Before installing a new instance of the module, the CO must first zeroize the SSPs on any previously deployed instances that are no longer in operation by following the instructions in Section 9.3.

11.1.1 Deploying the Module

Once zeroization is complete, the CO shall proceed with the following steps to deploy the module.

1. Download the appropriate vSRX VM image from the Juniper software download page.
2. Using a USB drive, copy the VM image to the hypervisor.
3. Open the VM management interface and create a new VM by importing the existing VM image.
4. Configure the desired amount of vRAM and number of vCPUs.
5. Configure a virtual disk as device type SCSI and select the desired amount of disk space.
6. Assign vNICs as follows:
    a. Configure one vNIC as the out-of-band management network interface (interface name `fxp0` in Junos). Assign it to the appropriate bridge network in the hypervisor.
    b. Configure additional vNICs as required for the deployment. Assign each additional vNIC to the appropriate bridge network in the hypervisor.
7. Remove or disable all virtual hardware not required for the deployment, such as virtual devices for video, audio, and remote console access (e.g., SPICE, VNC).
8. Complete the VM creation process and power on the VM from the hypervisor.

11.1.2 Setting the Root Password

Once the VM has finished initializing, the CO shall proceed as follows to set the root account password:

1. Log in to the `root` account. It should not require authentication.
2. Enter the Junos CLI configuration mode.
3. Set the root password.

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

4. Commit and exit configuration mode.

```
[edit]
root# commit
[edit]
root# exit
```

11.1.3 Installing the License

Next, the CO shall install the vSRX software license:

1. Enter the add license command.

   ```
   root> request system license add terminal
   ```

2. Paste the license key at the prompt and press **Ctrl+D**.

   ```
   [Type ^D at a new line to end input,
    enter blank line between each license key]
   ```

3. Verify that the license is installed.

   ```
   root> show system license
   ```

11.1.4 Enabling the Approved Mode

Lastly, the CO shall enable the Approved mode:

1. Enter configuration mode.

   ```
   root> set system fips level 1
   ```

2. Enable the Approved mode.

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

3. Commit and restart the module.

   ```
   [edit]
   root# commit
   [edit]
   root# exit
   root> request system reboot
   ```

After the module has restarted and the integrity and self-tests have run successfully on startup, the module is operating in the Approved mode. The CO must then create a backup image to ensure the module can be restored to this state by issuing the `request system snapshot` command.

The `show version` command will display the version of the Junos OS on the module so that the CO can confirm it is the FIPS-validated version. The CO must also verify the presence of the suffix string ":fips" in the CLI prompt, indicating the module is operating in the Approved mode.

11.2 Administrator Guidance

The CO is the person responsible for enabling, configuring, monitoring, and maintaining the module in the Approved mode. After securely initializing the module according to Section 11.1, the CO uses the root account to create the necessary user accounts and establishes keys and passwords.

To create the CO-role `crypto-officer` account, the CO enters the following commands:

```
[edit]
root:fips# set system login class security-admin permissions all
root:fips# set system login user crypto-officer class security-admin
root:fips# set system login user crypto-officer authentication plain-text-
password
root:fips# set system login user crypto-officer authentication ecdsa-sha2-
nistp256 "ecdsa-sha2-nistp256 <public_key_crypto_officer>"
```

To create the User-role `user` account, the CO enters the following commands:

```
[edit]
root:fips# set system login class monitor permissions trace
root:fips# set system login user user class monitor
root:fips# set system login user user authentication plain-text-password
root:fips# set system login user user authentication ecdsa-sha2-nistp256
"ecdsa-sha2-nistp256 <public_key_user>"
root:fips# commit
root:fips# exit
```

Next, the CO installs the IKE software package required to support IPsec VPN key management using IKE:

```
root:fips> request system software add optional://junos-ike.tgz
```

After creating the user accounts and installing the IKE package, the CO shall configure the module by setting a hostname, assigning interfaces to the appropriate security zones, configuring IP addresses, enabling approved services, and performing other security- and non-security-relevant configuration as needed.

The CO can configure and monitor the module via the virtualization platform's serial console interface (for example, using the `virsh console <domain>` command on a KVM hypervisor) or a remote SSH connection.

11.3 Non-Administrator Guidance

No specific non-administrator guidance is required to operate the module. Links to the module's documentation are provided in the References section.

11.4 Design and Rules

11.4.1 Module Design Rules

The module design corresponds to the security rules below.

1. The module clears previous authentications on restart.
2. The module does not require any operator action to perform self-tests on initialization.
3. The module inhibits data output during key generation, self-tests, zeroization, and error states.
4. The module's status information does not contain CSPs or sensitive data that, if misused, could lead to a compromise of the module.
5. The module does not restrict which SSPs are zeroized by the zeroization service.
6. The module does not support a maintenance interface or role.
7. The module does not output intermediate key values.
8. The module requires two independent internal actions to be performed prior to outputting plaintext CSPs.

11.4.2 Module Operation Rules

The statements below indicate a module operation security rule. The term *must* in this context specifically refers to a requirement for correct usage of the module in the Approved mode.

1. The CO must determine whether firmware being loaded on the module is a legacy use of the firmware load service.
2. The CO must retain control of the module while zeroization is in process.
3. The CO must not disable the Approved mode.

11.5 Maintenance Requirements

No special maintenance requirements apply.

11.6 End of Life

Before deleting an instance of the module, the CO shall execute the zeroize command described in Section 9.3 to securely remove all SSPs and configuration data from the module's virtual storage.

12 Mitigation of Other Attacks

The module does not implement mechanisms to mitigate other attacks beyond what is described in this security policy.

Glossary and Abbreviations

AES-NI	Intel Advanced Encryption Standard New Instructions
CAST	Conditional Algorithm Self-Test
CLI	Command Line Interface
DRBG	Deterministic Random Bit Generator
ECDSA	Elliptic Curve Digital Signature Algorithm
KAT	Known-Answer Test
KDF	Key Derivation Function
KVM	Kernel-based Virtual Machine
LTS	Long-Term Support
PCT	Pairwise Consistency Test
QCOW2	QEMU Copy-On-Write version 2
VA	Vendor Affirmed
vCPU	Virtual CPU
VM	Virtual Machine
vNIC	Virtual NIC
vRAM	Virtual RAM

References

Juniper Networks, “vSRX Documentation,” Juniper Networks, 2025. Available: https://www.juniper.net/documentation/product/us/en/vsrx/

Juniper Networks, “vSRX Virtual Firewall Datasheet,” Juniper Networks, October 2024. Available: https://www.juniper.net/us/en/products/security/srx-series/vsrx-virtual-firewall-datasheet.html

NIST, *FIPS 140-3 Implementation Guidance*, NIST, 2 September 2025. Available: https://csrc.nist.gov/csrc/media/Projects/cryptographic-module-validation-program/documents/fips 140-3/FIPS 140-3 IG.pdf

NIST, *FIPS 140-3*, *Security Requirements for Cryptographic Modules*, NIST, 22 March 2019. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf

NIST, *NIST Special Publication 800-133 Revision 2*, *Recommendation for Cryptographic Key Generation*, NIST, June 2020. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf

NIST, *NIST Special Publication 800-38D*, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, NIST, November 2007. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf

NIST, *NIST Special Publication 800-56A Revision 3*, *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*, NIST, April 2018. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf

NIST, *NIST Special Publication 800-56B Revision 2*, *Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography*, NIST, March 2019. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf