SUSE LLC

SUSE Linux Enterprise NSS Cryptographic Module

FIPS 140-3 Non-Proprietary Security Policy

Version 1.2

Last update: 2026-07-13

Prepared by:

atsec information security corporation

4516 Seton Center Parkway, Suite 250

Austin, TX 78759

www.atsec.com

Table of Contents

1 General ... 8
1.1 Overview ... 8
1.2 Security Levels ... 8
2 Cryptographic Module Specification ... 9
2.1 Description ... 9
2.2 Tested and Vendor Affirmed Module Version and Identification ... 10
2.3 Excluded Components ... 13
2.4 Modes of Operation ... 13
2.5 Algorithms ... 14
2.6 Security Function Implementations ... 28
2.7 Algorithm Specific Information ... 32
2.7.1 AES GCM IV ... 32
2.7.2 RSA 32
2.7.3 Legacy Use ... 32
2.7.4 Key Derivation using SP 800-132 PBKDF ... 33
2.7.5 Key Agreement ... 33
2.8 RBG and Entropy ... 33
2.9 Key Generation ... 34
2.10 Key Establishment ... 35
2.11 Industry Protocols ... 36
3 Cryptographic Module Interfaces ... 37
3.1 Ports and Interfaces ... 37
4 Roles, Services, and Authentication ... 38
4.1 Authentication Methods ... 38
4.2 Roles ... 38
4.3 Approved Services ... 38
4.4 Non-Approved Services ... 48
4.5 External Software/Firmware Loaded ... 50
5 Software/Firmware Security ... 51
5.1 Integrity Techniques ... 51
5.2 Initiate on Demand ... 51
6 Operational Environment ... 52

6.1 Operational Environment Type and Requirements 52
6.2 Configuration Settings and Restrictions 52
7 Physical Security 53
8 Non-Invasive Security 54
9 Sensitive Security Parameters Management 55
9.1 Storage Areas 55
9.2 SSP Input-Output Methods 55
9.3 SSP Zeroization Methods 56
9.4 SSPs 56
10 Self-Tests 66
10.1 Pre-Operational Self-Tests 66
10.2 Conditional Self-Tests 66
10.3 Periodic Self-Test Information 80
10.4 Error States 90
10.5 Operator Initiation of Self-Tests 90
11 Life-Cycle Assurance 91
11.1 Installation, Initialization, and Startup Procedures 91
11.1.1 Module Installation 91
11.1.2 Operating Environment Configuration 91
11.1.3 Access to Audit Data 92
11.1.4 Module Installation for Vendor Affirmed Platforms 92
11.2 Administrator Guidance 92
11.2.1 Considerations for the Approved Mode 93
11.3 Non-Administrator Guidance 94
11.4 End of Life 94
12 Mitigation of Other Attacks 95
12.1 Attack List 95
12.1.1 Blinding Against RSA Timing Attacks 95
12.1.2 Cache Invariant Modular Exponentiation 95
12.1.3 Double-Checking RSA Signatures 95
Appendix A. Glossary and Abbreviations 96
Appendix B. References 98

List of Tables

Table 1: Security Levels 8
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 11
Table 3: Tested Operational Environments - Software, Firmware, Hybrid 12
Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid 13
Table 5: Modes List and Description 13
Table 6: Approved Algorithms 25
Table 7: Vendor-Affirmed Algorithms 25
Table 8: Non-Approved, Allowed Algorithms with No Security Claimed 26
Table 9: Non-Approved, Not Allowed Algorithms 27
Table 10: Security Function Implementations 31
Table 11: Entropy Certificates 33
Table 12: Entropy Sources 34
Table 13: Ports and Interfaces 37
Table 14: Roles 38
Table 15: Approved Services 48
Table 16: Non-Approved Services 50
Table 17: Storage Areas 55
Table 18: SSP Input-Output Methods 55
Table 19: SSP Zeroization Methods 56
Table 20: SSP Table 1 61
Table 21: SSP Table 2 65
Table 22: Pre-Operational Self-Tests 66
Table 23: Conditional Self-Tests 80
Table 24: Pre-Operational Periodic Information 80
Table 25: Conditional Periodic Information 90
Table 26: Error States 90
Table 27 - Installation for Vendor Affirmed Platforms 92
Table 28 – RPM packages 93

List of Figures

Figure 1: Block Diagram.....10

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version 3.1 of the SUSE Linux Enterprise NSS Cryptographic Module. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for an overall Security Level 1 module. It has a one-to-one mapping to SP 800-140B starting with section B.2.1 named "General" which maps to Section 1 in this document and ending with section B.2.12 named "Mitigation of other attacks" which maps to Section 12 in this document.

This Non-Proprietary Security Policy may be reproduced and distributed, but only whole and intact and including this notice. Other documentation is proprietary to their authors.

1.2 Security Levels

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	1
	Overall Level	1

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The SUSE Linux Enterprise NSS Cryptographic Module (hereafter referred to as "the module") is a software library that provides a C language application program interface (API) designed to support cross-platform development of security-enabled client and server applications. Applications built with NSS can support SSLv3, TLS, IKEv2, PKCS#5, PKCS#7, PKCS#12, S/MIME, X.509 v3 certificates, and other security standards supporting FIPS 140-3 validated cryptographic algorithms.

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary:

The software block diagram below shows the cryptographic boundary of the module, and its interfaces with the operational environment.

Tested Operational Environment's Physical Perimeter (TOEPP):

The TOEPP of the module is defined as the general-purpose computer on which the module is installed.

The cryptographic boundary and TOEPP are schematically represented in Figure 1.

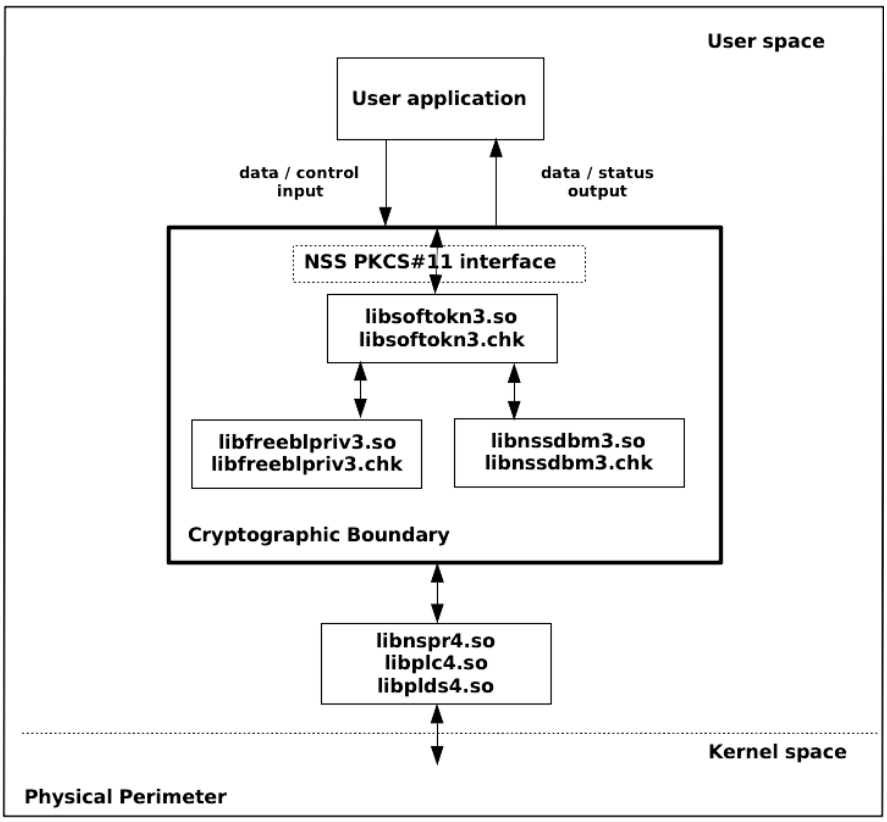

Figure 1: Block Diagram

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
libsoftokn3.so, libsoftokn3.chk, libnssdbm3.so, libnssdbm3.chk, libfreeblpriv3.so, libfreeblpriv3.chk on SUSE Linux Enterprise Server 15 SP4 and Intel® Xeon® Silver 4251R or AMD EPYC(TM) 7371	3.1	N/A	DSA SigVer
libsoftokn3.so, libsoftokn3.chk, libnssdbm3.so, libnssdbm3.chk, libfreeblpriv3.so, libfreeblpriv3.chk on SUSE Linux Enterprise Server 15 SP4 and ARM Ampere® Altra® Q80-30	3.1	N/A	DSA SigVer
libsoftokn3.so, libsoftokn3.chk, libnssdbm3.so, libnssdbm3.chk, libfreeblpriv3.so, libfreeblpriv3.chk on SUSE Linux Enterprise Server 15 SP4 and IBM z/15	3.1	N/A	DSA SigVer
libsoftokn3.so, libsoftokn3.chk, libnssdbm3.so, libnssdbm3.chk, libfreeblpriv3.so, libfreeblpriv3.chk on SUSE Linux Enterprise Server 15 SP4 and IBM	3.1	N/A	DSA SigVer

Package or File Name	Software/ Firmware Version	Features	Integrity Test
Power E1080 (9080-HEX)			

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

The table above lists the software components of the cryptographic module, which defines its cryptographic boundary.

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
SUSE Linux Enterprise Server 15 SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	Yes	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	No	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	GIGABYTE R181-Z90-00	AMD EPYC(TM) 7371	Yes	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	GIGABYTE R181-Z90-00	AMD EPYC(TM) 7371	No	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	GIGABYTE G242-P32-QZ	ARM Ampere(R) Altra(R) Q80-30	Yes	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	GIGABYTE G242-P32-QZ	ARM Ampere® Altra® Q80-30	No	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	IBM z/15	z15	Yes	N/A	3.1
SUSE Linux Enterprise Server 15 SP4	IBM z/15	z15	No	N/A	3.1
SUSE Linux Enterprise Server 15 SP4 on PowerVM (VIOS 3.1.4.00)	IBM Power E1080 (9080-HEX)	Power10	Yes	N/A	3.1

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
SUSE Linux Enterprise Server 15 SP4 on PowerVM (VIOS 3.1..4.00)	IBM Power E1080 (9080-HEX)	Power10	No	N/A	3.1

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

The module implements Processor Algorithm Implementation (PAI) for the IBM z/15 platform and Processor Algorithm Acceleration (PAA) for all other tested platforms listed above.

Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform
SUSE Linux Enterprise Server 15SP4	IBM LinuxONE III LT1 [z15]
SUSE Linux Enterprise Micro 5.3	Supermicro Super Server SYS-6019P-WTR [Intel® Xeon® Silver 4215R]
SUSE Linux Enterprise Micro 5.3	GIGABYTE R181-Z90-00 [AMD EPYC(TM) 7371]
SUSE Linux Enterprise Micro 5.3	GIGABYTE G242-P32-QZ [ARM Ampere® Altra® Q80-30]
SUSE Linux Enterprise Micro 5.3	IBM z/15 [z15]
SUSE Linux Enterprise Micro 5.3	IBM LinuxONE III LT1 [z15]
SUSE Linux Enterprise Server for SAP 15SP4	Supermicro Super Server SYS-6019P-WTR [Intel® Xeon® Silver 4215R]
SUSE Linux Enterprise Server for SAP 15SP4	GIGABYTE R181-Z90-00 [AMD EPYC(TM) 7371]
SUSE Linux Enterprise Server for SAP 15SP4	IBM Power E1080 (9080-HEX) [Power10]
SUSE Linux Enterprise Base Container Image 15SP4	Supermicro Super Server SYS-6019P-WTR [Intel® Xeon® Silver 4215R]
SUSE Linux Enterprise Base Container Image 15SP4	GIGABYTE R181-Z90-00 [AMD EPYC(TM) 7371]
SUSE Linux Enterprise Base Container Image 15SP4	GIGABYTE G242-P32-QZ [ARM Ampere® Altra® Q80-30]

Operating System	Hardware Platform
SUSE Linux Enterprise Base Container Image 15SP4	IBM z/15 [z15]
SUSE Linux Enterprise Base Container Image 15SP4	IBM LinuxONE III LT1 [z15]
SUSE Linux Enterprise Base Container Image 15SP4	IBM Power E1080 (9080-HEX) [Power10]
SUSE Linux Enterprise Desktop 15SP4	Supermicro Super Server SYS-6019P-WTR [Intel® Xeon® Silver 4215R]
SUSE Linux Enterprise Desktop 15SP4	GIGABYTE R181-Z90-00 [AMD EPYC(TM) 7371]
SUSE Linux Enterprise Real Time 15SP4	Supermicro Super Server SYS-6019P-WTR [Intel® Xeon® Silver 4215R]
SUSE Linux Enterprise Real Time 15SP4	GIGABYTE R181-Z90-00 [AMD EPYC(TM) 7371]

Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid

The SUSE Linux Enterprise Server operating system is used as the basis of other products. Compliance is maintained for SUSE products whenever the binary is found unchanged per the vendor affirmation from SUSE based on the allowance FIPS 140-3 Management Manual, Section 7.9.1, bullet 1 a i).

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

There are no components excluded from the requirements of the FIPS 140-3 standard.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved mode	Automatically entered whenever an approved service is requested	Approved	Equivalent to the indicator of the requested service (NSC_NSSGetFIPSStatus returns 1)
Non-approved mode	Automatically entered whenever a non-approved service is requested	Non-Approved	Equivalent to the indicator of the requested service (NSC_NSSGetFIPSStatus does not return 1)

Table 5: Modes List and Description

After passing all pre-operational self-tests and cryptographic algorithm self-tests executed on start-up, the module automatically transitions to the approved mode. No operator intervention is required to reach this point. The module operates in the approved mode of operation by default and can only transition into the non-approved mode by calling one of the non-approved services listed in the Non-Approved Services table of the Security Policy.

In the operational state, the module accepts service requests from calling applications through its logical interfaces. At any point in the operational state, a calling application can end its process, causing the module to end its operation.

Mode Change Instructions and Status:

The module automatically switches between the approved and non-approved modes depending on the services requested by the operator. The status indicator of the mode of operation is equivalent to the indicator of the service that was requested.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A3575	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC	A3581	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC	A3585	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A3580	Direction - decrypt, encrypt Key Length - 128, 192, 256 Payload Length - Payload Length: 128-65536 Increment 8	SP 800-38A
AES-CMAC	A3577	Direction - Generation, Verification Key Length - 128, 192, 256 MAC Length - MAC Length: 128 Message Length - Message Length: 8-524288 Increment 8	SP 800-38B
AES-CTR	A3575	Direction - Decrypt, Encrypt Key Length - 128, 192, 256 Payload Length - Payload Length: 8-128 Increment 8 Supports Counter larger than maximum value - No Incremental Counter - Yes Counter Tests Performed - Yes	SP 800-38A
AES-CTR	A3581	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
		Payload Length - Payload Length: 8-128 Increment 8 Supports Counter larger than maximum value - No Incremental Counter - Yes Counter Tests Performed - Yes	
AES-ECB	A3575	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3581	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3582	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3583	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3585	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3586	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3587	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A3575	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D
AES-GCM	A3581	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D

Algorithm	CAVP Cert	Properties	Reference
AES-GCM	A3582	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D
AES-GCM	A3583	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D
AES-GCM	A3585	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D
AES-GCM	A3586	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D
AES-GCM	A3587	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96, 128 Payload Length - Payload Length: 128, 1024, 120, 248 AAD Length - AAD Length: 0, 128, 1024, 120, 248	SP 800-38D

Algorithm	CAVP Cert	Properties	Reference
AES-KW	A3576	Direction - Decrypt, Encrypt Cipher - Cipher Key Length - 128, 192, 256 Payload Length - Payload Length: 128-4096 Increment 128	SP 800-38F
AES-KWP	A3576	Direction - Decrypt, Encrypt Cipher - Cipher Key Length - 128, 192, 256 Payload Length - Payload Length: 8-4096 Increment 8	SP 800-38F
DSA SigVer (FIPS186-4)	A3575	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
DSA SigVer (FIPS186-4)	A3584	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
DSA SigVer (FIPS186-4)	A3588	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A3575	Curve - P-256, P-384, P-521 Secret Generation Mode - Extra Bits	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A3584	Curve - P-256, P-384, P-521 Secret Generation Mode - Extra Bits	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A3588	Curve - P-256, P-384, P-521 Secret Generation Mode - Extra Bits	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3575	Curve - P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3584	Curve - P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3588	Curve - P-256, P-384, P-521	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A3575	Component - No Curve - P-256, P-384, P-521	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
		Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	
ECDSA SigGen (FIPS186-4)	A3584	Component - No Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A3588	Component - No Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A3575	Component - No Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A3584	Component - No Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A3588	Component - No Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
Hash DRBG	A3575	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
Hash DRBG	A3582	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256	SP 800-90A Rev. 1

Algorithm	CAVP Cert	Properties	Reference
		Additional Input - Additional Input: 0, 256 Returned Bits - 1024	
Hash DRBG	A3583	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
Hash DRBG	A3584	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
Hash DRBG	A3585	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
Hash DRBG	A3586	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
Hash DRBG	A3587	Prediction Resistance - No, Yes Supports Reseed - No	SP 800-90A Rev. 1

Algorithm	CAVP Cert	Properties	Reference
		Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	
Hash DRBG	A3588	Prediction Resistance - No, Yes Supports Reseed - No Mode - SHA2-256 Entropy Input - Entropy Input: 256 Nonce - Nonce: 256 Personalization String Length - Personalization String Length: 0, 256 Additional Input - Additional Input: 0, 256 Returned Bits - 1024	SP 800-90A Rev. 1
HMAC-SHA-1	A3575	MAC - MAC: 160 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA-1	A3588	MAC - MAC: 160 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A3575	MAC - MAC: 224 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A3584	MAC - MAC: 224 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A3588	MAC - MAC: 224 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A3575	MAC - MAC: 256 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A3584	MAC - MAC: 256 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A3588	MAC - MAC: 256 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A3575	MAC - MAC: 384 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-512	A3575	MAC - MAC: 512 Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A3575	Domain Parameter Generation Methods - P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A3584	Domain Parameter Generation Methods - P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A3588	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	A3575	Domain Parameter Generation Methods - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A3584	Domain Parameter Generation Methods - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A3588	Domain Parameter Generation Methods - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KDA HKDF Sp800-56Cr1	A3574	Fixed Info Pattern - uPartyInfo||vPartyInfo Fixed Info Encoding - concatenation Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-65336 Increment 8	SP 800-56C Rev. 2

Algorithm	CAVP Cert	Properties	Reference
		HMAC Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	
KDF IKEv1 (CVL)	A3579	Authentication Method - Pre-shared Key Initiator Nonce Length - Initiator Nonce Length: 128, 256, 512, 2048 Responder Nonce Length - Responder Nonce Length: 128, 256, 512, 2048 Preshared Key Length - Preshared Key Length: 8, 384, 768, 8192 Diffie-Hellman Shared Secret Length - Diffie-Hellman Shared Secret Length: 224, 2048, 8192 Hash Algorithm - SHA-1, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF IKEv2 (CVL)	A3579	Initiator Nonce Length - Initiator Nonce Length: 128, 256, 512, 2048 Responder Nonce Length - Responder Nonce Length: 128, 256, 512, 2048 Diffie-Hellman Shared Secret Length - Diffie-Hellman Shared Secret Length: 224, 2048, 8192 Derived Keying Material Length - Derived Keying Material Length: 1056, 3072 Hash Algorithm - SHA-1, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF SP800-108	A3578	KDF Mode - Counter, Double Pipeline Iteration, Feedback MAC Mode - CMAC-AES128, CMAC-AES192, CMAC-AES256, HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512 Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096 Fixed Data Order - After Fixed Data, Before Fixed Data Counter Length - 16, 24, 32, 8 Supports Empty IV - No Requires Empty IV - No Custom Key In Length - 0	SP 800-108 Rev. 1
KDF TLS (CVL)	A3575	TLS Version - v1.0/1.1	SP 800-135 Rev. 1
KDF TLS (CVL)	A3584	TLS Version - v1.0/1.1	SP 800-135 Rev. 1
KDF TLS (CVL)	A3588	TLS Version - v1.0/1.1	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
PBKDF	A3575	Iteration Count - Iteration Count: 10-1000 Increment 1 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 Password Length - Password Length: 8-128 Increment 1 Salt Length - Salt Length: 128-4096 Increment 8 Key Data Length - Key Data Length: 128-256 Increment 8	SP 800-132
PBKDF	A3584	Iteration Count - Iteration Count: 10-1000 Increment 1 HMAC Algorithm - SHA2-224, SHA2-256 Password Length - Password Length: 8-128 Increment 1 Salt Length - Salt Length: 128-4096 Increment 8 Key Data Length - Key Data Length: 128-256 Increment 8	SP 800-132
PBKDF	A3588	Iteration Count - Iteration Count: 10-1000 Increment 1 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256 Password Length - Password Length: 8-128 Increment 1 Salt Length - Salt Length: 128-4096 Increment 8 Key Data Length - Key Data Length: 128-256 Increment 8	SP 800-132
RSA KeyGen (FIPS186-4)	A3575	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.3 Info Generated By Server - No Public Exponent Mode - Fixed Fixed Public Exponent - 010001 Private Key Format - Standard	FIPS 186-4
RSA KeyGen (FIPS186-4)	A3584	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.3 Info Generated By Server - No Public Exponent Mode - Fixed Fixed Public Exponent - 010001 Private Key Format - Standard	FIPS 186-4
RSA KeyGen (FIPS186-4)	A3588	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.3 Info Generated By Server - No Public Exponent Mode - Fixed Fixed Public Exponent - 010001 Private Key Format - Standard	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
RSA SigGen (FIPS186-4)	A3575	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224	FIPS 186-4
RSA SigGen (FIPS186-4)	A3584	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224	FIPS 186-4
RSA SigGen (FIPS186-4)	A3588	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224	FIPS 186-4
RSA SigVer (FIPS186-4)	A3575	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224 Public Exponent Mode - Fixed Fixed Public Exponent - 010001	FIPS 186-4
RSA SigVer (FIPS186-4)	A3584	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224 Public Exponent Mode - Fixed Fixed Public Exponent - 010001	FIPS 186-4
RSA SigVer (FIPS186-4)	A3588	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224 Public Exponent Mode - Fixed Fixed Public Exponent - 010001	FIPS 186-4
Safe Primes Key Generation	A3575	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
Safe Primes Key Generation	A3584	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
Safe Primes Key Generation	A3588	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
SHA-1	A3575	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA-1	A3588	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-224	A3575	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-224	A3584	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-224	A3588	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A3575	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A3584	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A3588	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A3575	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A3575	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
TLS v1.2 KDF RFC7627 (CVL)	A3575	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.2 KDF RFC7627 (CVL)	A3584	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.2 KDF RFC7627 (CVL)	A3588	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1

Table 6: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
CKG	Key Type:Symmetric and Asymmetric	N/A	SP 800-133 Rev. 2, Section 4 Example 1 with V=0

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

The module does not implement any non-approved algorithms which are allowed in the approved mode of operation.

Non-Approved, Allowed Algorithms with No Security Claimed:

Name	Caveat	Use and Function
MD5	Only allowed as part of the PRF in TLSv1.0 and v1.1 per IG 2.4.A	Message digest used in TLS v1.0/v1.1 KDF only

Table 8: Non-Approved, Allowed Algorithms with No Security Claimed

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
AES in CBC-MAC and XCBC-MAC modes.	Symmetric encryption and decryption
AES in GCM with external IV.	Symmetric encryption
Camellia, CAST, CAST3, CAST5, ChaCha20, DES, DES2, Triple-DES, CDMF, IDEA, RC2, RC4, RC5, SEED	Symmetric key generation, encryption, and decryption
Poly1305	Symmetric encryption and decryption, message authentication code (MAC)
MD2, MD5	Message digest
HMAC using keys less than 112 bits of length; HMAC with non-approved message digest algorithms	Message authentication code (MAC)
DSA with any key size	Key pair generation, domain parameter generation and verification, digital signature generation
DSA with non-approved message digest algorithms	Digital signature verification
DSA with keys smaller than 1024 bits or greater than 3072 bits	Digital signature verification
RSA with pre-hashed message	Digital signature generation and verification

Name	Use and Function
RSA PSS with non-approved message digest algorithms	Digital signature generation and verification
RSA PSS with keys smaller than 2048 bits or greater than 4096 bits	Key pair generation, digital signature generation and verification
RSA PKCS#1v1.5 with non-approved message digest algorithms	Digital signature generation and verification
RSA PKCS#1v1.5 with keys smaller than 2048 bits or greater than 4096 bits	Key pair generation, digital signature generation and verification
RSA encryption and decryption with any key size.	Key encapsulation
ISO/IEC 9796 RSA	Digital signature generation and verification with and without message recovery
RSA X.509	RSA X.509 certificate generation
ECDSA with pre-hashed message	Digital signature generation and verification
ECDSA using non-approved message digest algorithms	Digital signature generation and verification
ECDSA with P-192 and P-224 curves, K curves, B curves and non-NIST curves.	Key pair generation, digital signature generation and verification
Curve25519	Key pair generation, domain parameter generation and verification, digital signature generation and verification
J-PAKE	Key agreement
HKDF (outside of the TLS 1.3 protocol), PBKDF1	Key derivation
Diffie-Hellman with keys generated with domain parameters other than safe primes	Diffie-Hellman shared secret computation
EC Diffie-Hellman with P-192 and P-224 curves, K curves, B curves and non-NIST curves	EC Diffie-Hellman shared secret computation

Table 9: Non-Approved, Not Allowed Algorithms

The table above lists all non-approved cryptographic algorithms of the module employed by the non-approved services listed in Section 4.4.

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Symmetric encryption	BC-UnAuth	Symmetric encryption		AES-CBC: (A3581, A3585, A3575) AES-CTR: (A3581, A3575) AES-ECB: (A3581, A3585, A3587, A3583, A3575, A3582, A3586) AES-CBC-CS1: (A3580)
Symmetric decryption	BC-UnAuth	Symmetric decryption		AES-CBC: (A3581, A3585, A3575) AES-CTR: (A3581, A3575) AES-ECB: (A3581, A3585, A3587, A3583, A3575, A3582, A3586) AES-CBC-CS1: (A3580)
Authenticated symmetric encryption	BC-Auth	Authenticated symmetric encryption		AES-GCM: (A3575, A3581, A3582, A3583, A3585, A3586, A3587)
Authenticated symmetric decryption	BC-Auth	Authenticated symmetric decryption		AES-GCM: (A3575, A3581, A3582, A3583, A3585, A3586, A3587)
Key wrapping (KTS)	KTS-Wrap	Key wrapping for CSP export	Standard:SP 800-38F IG D.G:approved method from IG D.G Caveat:Key establishment methodology provides between 128 and 256 bits of security strength	AES-KW: (A3576) AES-KWP: (A3576)

Name	Type	Description	Properties	Algorithms
Key unwrapping (KTS)	KTS-Unwrap	Key unwrapping for CSP import	Standard:SP 800-38F IG D.G:approved method from IG D.G Caveat:Key establishment methodology provides between 128 and 256 bits of security strength	AES-KW: (A3576) AES-KWP: (A3576)
Symmetric key generation	CKG	Key generation for AES and HMAC keys		CKG: ()
EC key pair generation	AsymKeyPair-KeyGen CKG	EC key pair generation		ECDSA KeyGen (FIPS186-4): (A3584, A3575, A3588) CKG: ()
EC public key verification	AsymKeyPair-KeyVer	EC public key verification		ECDSA KeyVer (FIPS186-4): (A3584, A3575, A3588)
Safe Primes key generation	AsymKeyPair-KeyGen CKG	Safe Primes key generation		Safe Primes Key Generation: (A3584, A3575, A3588) CKG: ()
RSA Key pair generation	AsymKeyPair-KeyGen CKG	RSA Key pair generation		RSA KeyGen (FIPS186-4): (A3584, A3575, A3588) CKG: ()
Digital signature generation	DigSig-SigGen	Digital signature generation		ECDSA SigGen (FIPS186-4): (A3584, A3575, A3588) RSA SigGen (FIPS186-4):

Name	Type	Description	Properties	Algorithms
				(A3584, A3575, A3588)
Digital signature verification	DigSig-SigVer	Digital signature verification		DSA SigVer (FIPS186-4): (A3584, A3575, A3588) L: 2048, 3072 N: 256 ECDSA SigVer (FIPS186-4): (A3584, A3575, A3588) RSA SigVer (FIPS186-4): (A3584, A3575, A3588)
Integrity test	DigSig-SigVer	Integrity test		DSA SigVer (FIPS186-4): (A3584, A3575, A3588)
Message digest	SHA	Message digest		SHA-1: (A3575, A3588) SHA2-224: (A3584, A3575, A3588) SHA2-256: (A3584, A3575, A3588) SHA2-384: (A3575) SHA2-512: (A3575)
Message authentication code (MAC)	MAC	Message authentication code (MAC)		HMAC-SHA-1: (A3575, A3588) HMAC-SHA2-224: (A3584, A3575, A3588) HMAC-SHA2-256: (A3584, A3575, A3588) HMAC-SHA2-384: (A3575) HMAC-SHA2-512: (A3575) AES-CMAC: (A3577)

Name	Type	Description	Properties	Algorithms
Deterministic random bit generation	DRBG	Deterministic random bit generation		Hash DRBG: (A3585, A3587, A3583, A3584, A3575, A3582, A3586, A3588)
EC Diffie-Hellman shared secret computation	KAS-SSC	EC Diffie-Hellman shared secret computation		KAS-ECC-SSC Sp800-56Ar3: (A3584, A3575, A3588)
Diffie-Hellman shared secret computation	KAS-SSC	Diffie-Hellman shared secret computation		KAS-FFC-SSC Sp800-56Ar3: (A3584, A3575, A3588)
Key derivation for TLS	KAS-135KDF	Key derivation for TLS v1.0, v1.1, and v1.2		KDF TLS: (A3584, A3575, A3588) TLS v1.2 KDF RFC7627: (A3584, A3575, A3588)
Key derivation for TLS (only TLS 1.3)	KAS-56CKDF	Key derivation for TLS (only TLS 1.3)		KDA HKDF Sp800-56Cr1: (A3574)
Password-based key derivation	PBKDF	Password-based key derivation		PBKDF: (A3584, A3575, A3588)
Key-based key derivation	KBKDF	Key-based key derivation		KDF SP800-108: (A3578)
Key derivation for IKEv1	KAS-135KDF	Key derivation for IKEv1		KDF IKEv1: (A3579)
Key derivation for IKEv2	KAS-135KDF	Key derivation for IKEv2		KDF IKEv2: (A3579)
DSA Digital Signature Verification (Legacy Use)	DigSig-SigVer	Digital signature verification with DSA		DSA SigVer (FIPS186-4): (A3575, A3584, A3588) L: 1024, 2048 N: 160, 224

Table 10: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES GCM IV

The module offers AES GCM IV generation in compliance with section 8.2.2 of SP 800-38D and IG C.H scenario 2, in which the GCM IV is generated internally and entirely randomly. The module uses a SP 800-90A Rev. 1-compliant DRBG for generating the IV.

The GCM IV must be at least 96 bits in length, which is enforced by the module.

When a GCM IV is used for decryption, the responsibility for the IV generation lies with the party that performs the AES GCM encryption.

The module implements AES GCM for use in the TLS v1.2 and v1.3 protocols. AES GCM IV generation is in compliance with IG C.H for both protocols as follows:

- For TLS v1.2, IV generation is in compliance with scenario 1.a of IG C.H and RFC5288. The module supports acceptable AES-GCM cipher suites from section 3.3.1 of SP 800-52 Rev. 2.
- For TLS v1.3, IV generation is in compliance with scenario 5 of IG C.H and RFC8446. The module supports acceptable AES-GCM cipher suites from section 3.3.1 of SP 800-52 Rev. 2.

Additionally, the module offers an internal deterministic IV generation mode compliant with Scenario 3 of FIPS 140-3 IG C.H. The size of the fixed (name) field used by this IV generation mode is at least 32 bits. The module then internally generates a 32 bit or longer deterministic non-repetitive counter. The module explicitly ensures that this counter is monotonically increasing at each invocation of the AES-GCM for the same encryption key, and that this counter does not exhaust all its possible values. The generated GCM IV is at least 96 bits in length.

The IV generated in both scenarios is only used within the context of the TLS protocol. The design of the TLS protocol implicitly ensures that the nonce_explicit, or counter portion of the IV will not exhaust all of its possible values.

In case the module's power is lost and then restored, the key used for the AES GCM encryption or decryption shall be redistributed.

2.7.2 RSA

In compliance with IG C.F, the module implements modulus sizes of 2048, 3072, and 4096 bits for signature generation and verification. Each algorithm was tested with all key sizes. The corresponding certificates can be found in Section 2.5.

There are no untested RSA modulus sizes used by the cryptographic module.

2.7.3 Legacy Use

Digital signature verification using DSA with L=1024 or N=224 is allowed for legacy use only.

This legacy algorithm can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M.

2.7.4 Key Derivation using SP 800-132 PBKDF

The module provides password-based key derivation (PBKDF), compliant with SP 800-132. The module supports option 1a from Section 5.4 of SP 800-132, in which the Master Key (MK) or a segment of it is used directly as the Data Protection Key (DPK).

In accordance with SP 800-132 and FIPS 140-3 IG D.N, the following requirements are met:

- Derived keys shall be used only for storage applications and shall not be used for any other purposes. The length of the MK or DPK is 112 bits or more.
- Passwords or passphrases, used as an input for PBKDF, are not used as cryptographic keys.
- The minimum length of the password or passphrase accepted by the module is 20 characters. The probability of guessing the value, assuming a worst-case scenario of all digits, is estimated to be at most 10^{-20}. Combined with the minimum iteration count as described below, this provides an acceptable trade-off between user experience and security against brute-force attacks.
- A portion of the salt, with a length of at least 128 bits, is generated randomly using the SP 800-90A Rev. 1 DRBG provided by the module.
- The iteration count shall be selected as large as possible, as long as the time required to generate the key using the entered password is acceptable for the users. The minimum value accepted by the module is 1000.

If any of these requirements are not met, the requested service is non-approved (see Section 4.4).

2.7.5 Key Agreement

The module does not establish SSPs using an approved key agreement scheme (KAS). However, it does offer some or all of the underlying KAS cryptographic functionality to be used by an external operator/application as part of an approved KAS.

2.8 RBG and Entropy

Cert Number	Vendor Name
E28	SUSE LLC
E29	SUSE LLC

Table 11: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Userspace Standalone CPU Time Jitter RNG (64-bit with internal timer)	Non-Physical	See Tested Operational Environment table	256 bits	Full entropy	SHA3-256 (A3034)
Userspace Standalone CPU Time Jitter RNG (64-bit with external timer)	Non-Physical	See Tested Operational Environment table	256 bits	Full entropy	SHA3-256 (A3034)

Table 12: Entropy Sources

The module employs a Deterministic Random Bit Generator (DRBG) based on SP 800-90A Rev. 1 for the creation of seeds for symmetric keys, asymmetric keys, RSA signature generation, and ECDSA signature generation. In addition the module provides a Random Number Generation service to calling applications.

The DRBG supports the Hash_DRBG mechanism using SHA2-256 and without prediction resistance. The module uses the SP 800-90B-compliant entropy sources specified above. These entropy sources are located within the physical perimeter, but outside the cryptographic boundary of the module. The module obtains 384 bits of entropy to instantiate the DRBG and 256 bits to reseed it, sufficient to provide a DRBG with 256 bits of security strength.

2.9 Key Generation

In accordance with FIPS 140-3 IG D.H, the cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys.

- For generating RSA and ECDSA keys, the module implements asymmetric cryptographic key generation (CKG) services compliant with [FIPS186-4], providing 112 to 149 bits of key strength for RSA and 128 to 256 bits for ECDSA.
- The public and private keys used in the EC Diffie-Hellman shared secret computation schemes are generated internally by the module using the EC key generation method compliant with [FIPS186-4] and [SP800-56Arev3], providing 128 to 256 bits of key strength.
- The public and private keys used in the Diffie-Hellman shared secret computation scheme are also compliant with [SP800-56Arev3]. The module generates keys using safe primes defined in RFC7919 and RFC3526, providing 112 to 200 bits of key strength as described in the next section.

Additionally, for AES and HMAC keys, the module provides key generation services compliant with section 4 of [SP800-133rev2], providing 128 to 256 bits of key strength for AES and 112-256 bits of key strength for HMAC.

Random values used for symmetric and asymmetric key generation are obtained directly from an approved SP 800-90A Rev. 1 DRBG that supports the required security strength requested by the caller (without any V, as described in Additional Comments 2 of IG D.H), in compliance with Section 4 of SP 800-133 Rev. 2.

The module supports the following key derivation methods:

- KDF for the TLS protocol, used as pseudo-random functions (PRF) for TLSv1.0/1.1 and TLSv1.2, compliant with SP 800-135 Rev. 1.
- KDF for the Internet Key Exchange (IKE) protocol versions 1 and 2, compliant with SP 800-135 Rev. 1.
- HKDF for the TLS protocol TLSv1.3, compliant with SP 800-56C Rev. 2.
- KBKDF, compliant with SP 800-108 Rev 1. This implementation can be used to generate secret keys from a pre-existing key-derivation-key.
- PBKDF, compliant with option 1a of SP 800-132. This implementation can only be used to derive keys for storage applications.

2.10 Key Establishment

The module provides Diffie-Hellman (dhEphem) and EC Diffie-Hellman (Ephemeral Unified Scheme) shared secret computation compliant with SP 800-56A Rev. 3, in accordance with Scenario 2 (1) of IG D.F.

For Diffie-Hellman, the module supports the use of safe primes from RFC7919 for domain parameters and key generation.

- TLS (RFC7919)
  - ffdhe2048 (ID = 256)
  - ffdhe3072 (ID = 257)
  - ffdhe4096 (ID = 258)
  - ffdhe6144 (ID = 259)
  - ffdhe8192 (ID = 260)

The module also supports the use of safe primes from RFC3526, which are part of the Modular Exponential (MODP) Diffie-Hellman groups that can be used for Internet Key Exchange (IKE). Note that the module only implements key generation and verification, and shared secret computation using safe primes, but no part of the IKE protocol.

- IKEv2 (RFC3526)
  - MODP-2048 (ID=14)
  - MODP-3072 (ID=15)
  - MODP-4096 (ID=16)
  - MODP-6144 (ID=17)
  - MODP-8192 (ID=18)

For Elliptic Curve Diffie-Hellman, the module supports the NIST-defined P-256, P-384, and P-521 curves.

According to Table 2: Comparable strengths in SP 800-57 Rev. 5, the key sizes of Diffie-Hellman and EC Diffie-Hellman provide the following security strength in the approved mode of operation:

- Diffie-Hellman shared secret computation provides between 112 and 200 bits of strength.
- EC Diffie-Hellman shared secret computation provides between 128 and 256 bits of strength.

In addition, the module provides the following methods for key transport to securely input SSPs into and output SSPs from the module:

- Key wrapping with AES KW
- Key wrapping with AES KWP

This method of key transport provides 128, 192, or 256 bits of security strength when used with 128-, 192-, and 256-bit keys respectively.

2.11 Industry Protocols

The module does not implement any industry protocols. However, the module does offer cryptographic components that can be used by a calling application to implement various industry protocols, including but not limited to those named in Section 2.1.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API input parameters for data
N/A	Data Output	API output parameters for data
N/A	Control Input	API function calls, API input parameters for control input, /proc/sys/crypto/fips_enabled control file
N/A	Status Output	API return codes, API output parameters for status output

Table 13: Ports and Interfaces

As a software-only module, the module does not have physical ports. The operator can only interact with the module through the API provided by the module. Thus, the physical ports are interpreted to be the physical ports of the hardware platform on which the module runs.

The logical interfaces are the APIs through which the applications request services. These logical interfaces are logically separated from each other by the API design. All data output via the data output interface is inhibited when the module is performing pre-operational self-tests, conditional self-tests, zeroization, or when the module is in an error state.

The module does not implement a control output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module does not support authentication methods.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	Crypto Officer	None

Table 14: Roles

The Crypto Officer is implicitly and always assumed by the operator of the module. The module does not support multiple concurrent operators.

4.3 Approved Services

The table below lists the approved services provided by the module. For each service, the table lists the associated cryptographic algorithm(s), the role to perform the service, the cryptographic keys or CSPs involved, and their access type(s). The following convention is used to specify access rights to a CSP:

- **Generate (G)**: The module generates or derives the SSP.
- **Read (R)**: The SSP is read from the module (e.g. the SSP is output).
- **Write (W)**: The SSP is updated, imported, or written to the module.
- **Execute (E)**: The module uses the SSP in performing a cryptographic operation.
- **Zeroize (Z)**: The module zeroizes the SSP.

The details of the approved cryptographic algorithms including the CAVP certificate numbers can be found in Section 2.5.

The module implements the method NSC_NSSGetFIPSStatus() to indicate whether the last service requested was approved.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Symmetric encryption	Perform AES encryption	NSC_NSSGetFIPSStatus = 1	Key, IV, Plaintext	Ciphertext	Symmetric encryption	Crypto Officer - AES key: W,E
Symmetric decryption	Perform AES decryption	NSC_NSSGetFIPSStatus = 1	Key, IV, Ciphertext	Plaintext	Symmetric decryption	Crypto Officer - AES key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Authenticated symmetric encryption	Perform authenticated AES encryption	NSC_NSSGetFIPSStatus = 1	Key, IV, Plaintext	Ciphertext	Authenticated symmetric encryption	Crypto Officer - AES key: W,E
Authenticated symmetric decryption	Perform authenticated AES decryption	NSC_NSSGetFIPSStatus = 1	Key, IV, Ciphertext	Plaintext or Fail	Authenticated symmetric decryption	Crypto Officer - AES key: W,E
Key wrapping	Perform AES-based key wrapping	NSC_NSSGetFIPSStatus = 1	Key to be wrapped, Key wrapping key	Wrapped key	Key wrapping (KTS)	Crypto Officer - AES key: W,E
Key unwrapping	Perform AES-based key unwrapping	NSC_NSSGetFIPSStatus = 1	Wrapped key, Key wrapping key	Unwrapped key	Key unwrapping (KTS)	Crypto Officer - AES key: W,E
Symmetric key generation	Generate AES or HMAC key	NSC_NSSGetFIPSStatus = 1	Key size	Module generated key	Symmetric key generation	Crypto Officer - Module generated AES key: G,R - Module generated HMAC key: G,R
Asymmetric key generation	Generate key pairs	NSC_NSSGetFIPSStatus = 1	Key type, domain parameters, key size	Module generated key pair	EC key pair generation Safe Primes key generation RSA Key pair generation	Crypto Officer - Module generated RSA private key: G,R - Module generated RSA public key: G,R - Module generate

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						d EC private key: G,R - Module generated EC public key: G,R - Module generated Diffie-Hellman private key: G,R - Module generated Diffie-Hellman public key: G,R
Digital signature generation	Generate a signature	NSC_NSSGetFIPSStatus = 1	Message, hash algorithm, private key	Digital signature	Digital signature generation	Crypto Officer - RSA private key: W,E - EC private key: W,E
Digital signature verification	Verify a signature	NSC_NSSGetFIPSStatus = 1	Signature, hash algorithm, public key	Verification result	Digital signature verification	Crypto Officer - RSA public key: W,E - EC public key: W,E - DSA public key: W,E
DSA Signature Verificatio	Verify DSA signatures generated with	NSC_NSSGetFIPSStatus = 1	Signature, hash algorithm, public key	Verification result	DSA Digital Signature	Crypto Officer - DSA

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
n (Legacy Use)	parameters L = 1024 or 2048, N= 160 or 224 prior to the sunset data provided in IG C.M				Verification (Legacy Use)	public key: W,E
Public key validation	Verify a public key	NSC_NSSGetFIPSStatus = 1	Public key	Pass or Invalid	EC public key verification	Crypto Officer - EC public key: W,E
Random number generation	Generate random bitstrings	NSC_NSSGetFIPSStatus = 1	Number of bits	Random bitstring	Deterministic random bit generation	Crypto Officer - Entropy input: W,E,Z - DRBG seed: G,E,Z - DRBG internal state: V, C: G,W,E
Message digest	Compute SHA hashes	NSC_NSSGetFIPSStatus = 1	Message	Digest of the message	Message digest	Crypto Officer
Message authentication code (MAC)	Compute a MAC tag	NSC_NSSGetFIPSStatus = 1	Message, HMAC key or AES key	Message authentication code	Message authentication code (MAC)	Crypto Officer - AES key: W,E - HMAC key: W,E
Diffie-Hellman shared secret computation	Perform DH shared secret computation	NSC_NSSGetFIPSStatus = 1	Diffie-Hellman private key (owner), Diffie-Hellman public key from peer	Diffie-Hellman shared secret	Diffie-Hellman shared secret computation	Crypto Officer - Diffie-Hellman private key: W,E - Diffie-

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Hellman public key: W,E - Diffie-Hellman shared secret: G,R
EC Diffie-Hellman shared secret computation	Perform ECDH shared secret computation	NSC_NSSGetFIPSStatus = 1	EC private key (owner), EC public key from peer	EC Diffie-Hellman shared secret	EC Diffie-Hellman shared secret computation	Crypto Officer - EC private key: W,E - EC public key: W,E - EC Diffie-Hellman shared secret: G,R
Key derivation for TLS	Perform key derivation for TLS	NSC_NSSGetFIPSStatus = 1	(EC) Diffie-Hellman shared secret	TLS derived key	Key derivation for TLS Key derivation for TLS (only TLS 1.3)	Crypto Officer - Diffie-Hellman shared secret: W,E - EC Diffie-Hellman shared secret: W,E - TLS derived key: G,R
Key derivation for IKEv1 and IKEv2	Perform key derivation for IKEv1 and IKEv2	NSC_NSSGetFIPSStatus = 1	(EC) Diffie-Hellman shared secret	IKE derived key	Key derivation for IKEv1 Key	Crypto Officer - Diffie-Hellman

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					derivation for IKEv2	shared secret: W,E - EC Diffie-Hellman shared secret: W,E - IKE derived key: G,R
Password-based key derivation	Perform key derivation from a password/passphrase	NSC_NSSGetFIPSStatus = 1	Password/Passphrase, Salt, Key size, Iteration Count	PBKDF derived key	Password-based key derivation	Crypto Officer - Password or passphrase: W,E - PBKDF derived key: G,R
Key-based key derivation	Perform key derivation from a key	NSC_NSSGetFIPSStatus = 1	Key derivation key	KBKDF derived key	Key-based key derivation	Crypto Officer - Key derivation key: W,E - KBKDF derived key: G,R
Show status	Show module status	N/A	None	Return codes and/or log messages	None	Crypto Officer
Zeroization	Zeroize CSPs	N/A	Context containing SSPs	N/A	None	Crypto Officer - Module generated AES key: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- AES key: Z - Module generated HMAC key: Z - HMAC key: Z - Module generated RSA private key: Z - Module generated RSA public key: Z - RSA private key: Z - RSA public key: Z - Module generated EC private key: Z - Module generated EC public key: Z - EC private key: Z - EC public key: Z - Module generated Diffie-Hellman private

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						key: Z - Module generated Diffie-Hellman public key: Z - Diffie-Hellman private key: Z - Diffie-Hellman public key: Z - DSA public key: Z - Intermediate key generation value: Z - Diffie-Hellman shared secret: Z - EC Diffie-Hellman shared secret: Z - Password or passphrase: Z - PBKDF derived key: Z - Entropy input: Z - DRBG seed: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- DRBG internal state: V, C: Z - TLS derived key: Z - IKE derived key: Z - Key derivation key: Z - KBKDF derived key: Z
Self-tests	Perform self-tests	N/A	Module reset	Result of self-test (pass/fail)	Symmetric encryption Symmetric decryption Authenticated symmetric encryption Authenticated symmetric decryption Key wrapping (KTS) Key unwrapping (KTS) Digital signature generation Digital signature verification Integrity test Message	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					digest Message authentication code (MAC) Deterministic random bit generation EC Diffie-Hellman shared secret computation Diffie-Hellman shared secret computation Key derivation for TLS Key derivation for TLS (only TLS 1.3) Password-based key derivation Key-based key derivation Key derivation for IKEv1 Key derivation for IKEv2	

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Module installation and configuration	Install and configure module	N/A	Configuration parameters	Return codes and/or log messages	None	Crypto Officer
Module initialization	Initialize module	N/A	None	None	None	Crypto Officer
Show module name and version	Show module name and version	N/A	None	Name and version information	None	Crypto Officer

Table 15: Approved Services

4.4 Non-Approved Services

Name	Description	Algorithms	Role
AES in CBC-MAC and XCBC-MAC modes.	Symmetric encryption and decryption	AES in CBC-MAC and XCBC-MAC modes.	CO
AES in GCM with external IV.	Symmetric encryption	AES in GCM with external IV.	CO
Camellia, CAST, CAST3, CAST5, ChaCha20, DES, DES2, Triple-DES, CDMF, IDEA, RC2, RC4, RC5, SEED	Symmetric key generation, encryption, and decryption	Camellia, CAST, CAST3, CAST5, ChaCha20, DES, DES2, Triple-DES, CDMF, IDEA, RC2, RC4, RC5, SEED	CO
Poly1305	Symmetric encryption and decryption, message authentication code (MAC)	Poly1305	CO
MD2, MD5	Message digest	MD2, MD5	CO
HMAC using keys less than 112 bits of length; HMAC with non-approved message digest algorithms	Message authentication code (MAC)	HMAC using keys less than 112 bits of length; HMAC with non-approved message digest algorithms	CO
DSA with any key size	Key pair generation, domain parameter generation and	DSA with any key size	CO

Name	Description	Algorithms	Role
	verification, digital signature generation		
DSA with non-approved message digest algorithms	Digital signature verification	DSA with non-approved message digest algorithms	CO
DSA with keys smaller than 1024 bits or greater than 3072 bits	Digital signature verification	DSA with keys smaller than 1024 bits or greater than 3072 bits	CO
RSA with pre-hashed message	Digital signature generation and verification	RSA with pre-hashed message	CO
RSA PSS with non-approved message digest algorithms	Digital signature generation and verification	RSA PSS with non-approved message digest algorithms	CO
RSA PSS with keys smaller than 2048 bits or greater than 4096 bits	Key pair generation, digital signature generation and verification	RSA PSS with keys smaller than 2048 bits or greater than 4096 bits	CO
RSA PKCS#1v1.5 with non-approved message digest algorithms	Digital signature generation and verification	RSA PKCS#1v1.5 with non-approved message digest algorithms	CO
RSA PKCS#1v1.5 with keys smaller than 2048 bits or greater than 4096 bits	Key pair generation, digital signature generation and verification	RSA PKCS#1v1.5 with keys smaller than 2048 bits or greater than 4096 bits	CO
RSA encryption and decryption with any key size.	Key encapsulation	RSA encryption and decryption with any key size.	CO
ISO/IEC 9796 RSA	Digital signature generation and verification with and without message recovery	ISO/IEC 9796 RSA	CO
RSA X.509	RSA X.509 certificate generation	RSA X.509	CO
ECDSA with pre-hashed message	Digital signature generation and verification	ECDSA with pre-hashed message	CO
ECDSA using non-approved message digest algorithms	Digital signature generation and verification	ECDSA using non-approved message digest algorithms	CO
ECDSA with P-192 and P-224 curves, K curves, B curves and non-NIST curves.	Key pair generation, digital signature generation and verification	ECDSA with P-192 and P-224 curves, K curves, B curves and non-NIST curves.	CO

Name	Description	Algorithms	Role
Curve25519	Key pair generation, domain parameter generation and verification, digital signature generation and verification	Curve25519	CO
J-PAKE	Key agreement	J-PAKE	CO
HKDF (outside of the TLS 1.3 protocol), PBKDF1	Key derivation	HKDF (outside of the TLS 1.3 protocol), PBKDF1	CO
Diffie-Hellman with keys generated with domain parameters other than safe primes	Diffie-Hellman shared secret computation	Diffie-Hellman with keys generated with domain parameters other than safe primes	CO
EC Diffie-Hellman with P-192 and P-224 curves, K curves, B curves and non-NIST curves	EC Diffie-Hellman shared secret computation	EC Diffie-Hellman with P-192 and P-224 curves, K curves, B curves and non-NIST curves	CO

Table 16: Non-Approved Services

4.5 External Software/Firmware Loaded

The module does not load external software or firmware.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by performing a DSA signature verification for each component that comprises the module. The module uses DSA signature verification with SHA2-256 using a 2048-bit key hardcoded in the module. If the DSA signature for any of the components cannot be verified, then the test fails and the module enters the error state.

5.2 Initiate on Demand

Integrity tests are performed as part of the pre-operational self-tests, which are executed when the module is initialized after the system is rebooted. The integrity tests can also be performed on demand by invoking the sftk_FIPSRepeatIntegrityCheck() function which will perform integrity tests and the cryptographic algorithm self-tests. During the execution of the on-demand self-tests, services are not available and no data output is possible.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The module operates in a modifiable operational environment per the FIPS 140-3 level 1 specifications. The SUSE Linux Enterprise Server operating system is used as the basis of other products. Compliance is maintained for SUSE products whenever the binary is found unchanged per the vendor affirmation from SUSE based on the allowance provided by the FIPS 140-3 Management Manual, Section 7.9.1 Bullet 1 a i.

If properly installed, the operating system provides process isolation and memory protection mechanisms that ensure appropriate separation for memory access among the processes on the system. Each process has control over its own data and uncontrolled access to the data of other processes is prevented.

SSPs contained within the module are protected by the process isolation and memory separation mechanisms provided by the SUSE Linux Enterprise Server operating system, and only the module has control over these SSPs.

6.2 Configuration Settings and Restrictions

The module shall be installed as stated in Section 11.1.

Instrumentation tools like the ptrace system call, gdb and strace, userspace live patching, as well as other tracing mechanisms offered by the Linux environment such as ftrace or systemtap, shall not be used in the operational environment. The use of any of these tools implies that the cryptographic module is running in a non-validated operational environment.

7 Physical Security

The module is comprised of software only and therefore this section is not applicable.

8 Non-Invasive Security

This module does not implement any non-invasive security mechanisms, and therefore this section is not applicable.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Temporary storage for SSPs used by the module as part of service execution. The module does not perform persistent storage of SSPs	Dynamic

Table 17: Storage Areas

The module does not perform persistent storage of SSPs. The SSPs are temporarily stored in the RAM in plaintext form. SSPs are stored until they are zeroized by the operator (using a zeroization call or removing power from the module) or zeroized automatically.

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API input parameters (plaintext)	Operator calling application (TOEPP)	Cryptographic module	Plaintext	Manual	Electronic	
API input parameters (encrypted)	Operator calling application (TOEPP)	Cryptographic module	Encrypted	Manual	Electronic	Key unwrapping (KTS)
API output parameters (plaintext)	Cryptographic module	Operator calling application (TOEPP)	Plaintext	Manual	Electronic	
API output parameters (encrypted)	Cryptographic module	Operator calling application (TOEPP)	Encrypted	Manual	Electronic	Key wrapping (KTS)

Table 18: SSP Input-Output Methods

CSPs (with the exception of passwords) are wrapped on export and unwrapped on import to the module. PSPs are input and output in plaintext.

The module does not support manual SSP entry or intermediate SSP generation output. SSPs are provided to the module via API input parameters and output via API output parameters within the operational environment.

This is allowed by FIPS 140-3 IG 9.5.A, according to the "CM Software to/from App via TOEPP Path" entry in the Key Establishment table.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Wipe and Free memory block allocated	Zeroizes the SSPs contained within the cipher handle.	Memory occupied by SSPs is overwritten with zeroes and then it is released, which renders the SSP values irretrievable. The completion of the zeroization routine indicates that the zeroization procedure succeeded.	By calling the cipher related zeroization API (FC_Finalize, FC_CloseSession, or FC_CloseAllSession)
Automatic	Automatically zeroized by the module when no longer needed	Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable.	N/A
Module Reset	De-allocates the volatile memory used to store SSPs	Volatile memory used by the module is overwritten within nanoseconds when power is removed.	By unloading and reloading the module

Table 19: SSP Zeroization Methods

The memory occupied by SSPs is allocated by regular memory allocation operating system calls. The application acting as the CO is responsible for calling the appropriate zeroization functions provided in the module's API. The zeroization functions overwrite the memory occupied by SSPs with zeros then deallocate the memory using memory deallocation operating system calls. The completion of a zeroization routine serves as an indicator that zeroization has succeeded.

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Module generated AES key	Module generated AES key	128, 192, 256 bits - 128, 192, 256 bits	Symmetric key - CSP	Symmetric key generation		
AES key	AES key	128, 192, 256 bits -	Symmetric key - CSP			Symmetric encryption Symmetric decryption

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		128, 192, 256 bits				Authenticated symmetric encryption Authenticated symmetric decryption Key wrapping (KTS) Key unwrapping (KTS) Message authentication code (MAC)
Module generated HMAC key	Module generated HMAC key	112-256 bits - 112-256 bits	Symmetric key - CSP	Symmetric key generation		
HMAC key	HMAC key	112-256 bits - 112-256 bits	Symmetric key - CSP			Message authentication code (MAC)
Module generated RSA private key	Module generated RSA private key	2048, 3072, 4096 bits - 112, 128, 149 bits	Private key - CSP	RSA Key pair generation		
Module generated RSA public key	Module generated RSA public key	2048, 3072, 4096 bits - 112, 128, 149 bits	Public key - PSP	RSA Key pair generation		
RSA private key	RSA private key	2048, 3072, 4096 bits - 112, 128, 149 bits	Private key - CSP			Digital signature generation
RSA public key	RSA public key	2048, 3072, 4096 bits - 112,	Public key - PSP			Digital signature verification

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		128, 149 bits				
Module generated EC private key	Module generated EC private key	P-256, P-384, P-521 bits - 128, 192, 256 bits	Private key - CSP	EC key pair generation		
Module generated EC public key	Module generated EC public key	P-256, P-384, P-521 bits - 128, 192, 256 bits	Public key - PSP	EC key pair generation		
EC private key	EC private key	P-256, P-384, P-521 bits - 128, 192, 256 bits	Private key - CSP			Digital signature generation EC Diffie-Hellman shared secret computation
EC public key	EC public key	P-256, P-384, P-521 bits - 128, 192, 256 bits	Public key - PSP			Digital signature verification EC Diffie-Hellman shared secret computation
Module generated Diffie-Hellman private key	Module generated Diffie-Hellman private key	2048, 3072, 4096, 6144, 8192 bits - 112, 128, 152, 176, 200 bits	Private key - CSP	Safe Primes key generation		
Module generated Diffie-Hellman public key	Module generated Diffie-Hellman public key	2048, 3072, 4096, 6144, 8192 bits - 112,	Public key - PSP	Safe Primes key generation		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		128, 152, 176, 200 bits				
Diffie-Hellman private key	Diffie-Hellman private key	2048, 3072, 4096, 6144, 8192 bits - 112, 128, 152, 176, 200 bits	Private key - CSP			Diffie-Hellman shared secret computation
Diffie-Hellman public key	Diffie-Hellman public key	2048, 3072, 4096, 6144, 8192 bits - 112, 128, 152, 176, 200 bits	Public key - PSP			Diffie-Hellman shared secret computation
DSA public key	DSA public key	1024, 2048, 3072 bits - 80, 112, 128 bits	Public key - PSP			Digital signature verification Integrity test DSA Digital Signature Verification (Legacy Use)
Intermediate key generation value	Intermediate key generation value	112 to 2048 bits - 112 to 256 bits	Intermediate value - CSP	EC key pair generation Safe Primes key generation RSA Key pair generation		EC key pair generation Safe Primes key generation RSA Key pair generation
Diffie-Hellman shared secret	Diffie-Hellman shared secret	2048, 3072, 4096, 6144, 8192 bits - 112, 128, 152,	Shared secret - CSP		Diffie-Hellman shared secret computation	Key derivation for TLS Key derivation for TLS (only TLS 1.3) Key derivation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		176, 200 bits				for IKEv1 Key derivation for IKEv2
EC Diffie-Hellman shared secret	EC Diffie-Hellman shared secret	P-256, P-384, P-512 - 128, 192, 256 bits	Shared secret - CSP		EC Diffie-Hellman shared secret computation	
Password or passphrase	Password or passphrase used for PBKDF	20 or more characters - N/A	Password - CSP			Password-based key derivation
PBKDF derived key	PBKDF derived key	- 112 to 256 bits	Symmetric key - CSP	Password-based key derivation		
Entropy input	Entropy input	256, 384 bits - 256, 384 bits	Entropy Input - CSP	Deterministic random bit generation		Deterministic random bit generation
DRBG seed	DRBG seed - IG D.L compliant	440 bits - 256 bits	Seed - CSP	Deterministic random bit generation		Deterministic random bit generation
DRBG internal state: V, C	DRBG internal state: V, C - IG D.L compliant	880 bits - 256 bits	Internal state - CSP	Deterministic random bit generation		Deterministic random bit generation
TLS derived key	TLS derived key	112-8192 bits - 112-256 bits	Symmetric key - CSP	Key derivation for TLS Key derivation for TLS (only TLS 1.3)		
IKE derived key	IKE derived key	112-8192 bits - 112-256 bits	Symmetric key - CSP	Key derivation for IKEv1 Key		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
				derivation for IKEv2		
Key derivation key	Key derivation key	112-8192 bits - 112-256 bits	Symmetric key - CSP			Key-based key derivation
KBKDF derived key	KBKDF derived key	112-4096 bits - 128 to 256 bits	Symmetric key - CSP	Key-based key derivation		

Table 20: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Module generated AES key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	
AES key	API input parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
Module generated HMAC key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	
HMAC key	API input parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
Module generated RSA private key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated RSA public key:Paired With Intermediate key generation value:Derived From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Module generated RSA public key	API output parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated RSA private key:Paired With Intermediate key generation value:Derived From
RSA private key	API input parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	RSA public key:Paired With
RSA public key	API input parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	RSA private key:Paired With
Module generated EC private key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated EC public key:Paired With Intermediate key generation value:Derived From
Module generated EC public key	API output parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated EC private key:Paired With Intermediate key generation value:Derived From
EC private key	API input parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC public key:Paired With EC Diffie-Hellman shared secret:Establishes
EC public key	API input parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC private key:Paired With EC Diffie-Hellman shared secret:Establishes

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Module generated Diffie-Hellman private key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated Diffie-Hellman public key:Paired With Intermediate key generation value:Derived From
Module generated Diffie-Hellman public key	API output parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated Diffie-Hellman private key:Paired With Intermediate key generation value:Derived From
Diffie-Hellman private key	API input parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Diffie-Hellman public key:Paired With Diffie-Hellman shared secret:Establishes
Diffie-Hellman public key	API input parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Diffie-Hellman private key:Paired With Diffie-Hellman shared secret:Establishes
DSA public key	API input parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
Intermediate key generation value		RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Module generated RSA private key:Derives Module generated RSA public key:Derives Module generated EC private key:Derives Module generated EC public key:Derives Module generated

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					Diffie-Hellman private key:Derives Module generated Diffie-Hellman public key:Derives
Diffie-Hellman shared secret	API input parameters (encrypted) API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Diffie-Hellman private key:Established by Diffie-Hellman public key:Established by TLS derived key:Derives IKE derived key:Derives
EC Diffie-Hellman shared secret	API input parameters (encrypted) API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	EC private key:Established by EC public key:Established by TLS derived key:Derives IKE derived key:Derives
Password or passphrase	API input parameters (plaintext)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	PBKDF derived key:Derives
PBKDF derived key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Password or passphrase:Derived From
Entropy input		RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	DRBG seed:Derives
DRBG seed		RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Entropy input:Derived From DRBG internal state: V, C:Derives

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
DRBG internal state: V, C		RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Intermediate key generation value:Derives Module generated AES key:Derives Module generated HMAC key:Derives
TLS derived key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Diffie-Hellman shared secret:Derived From EC Diffie-Hellman shared secret:Derived From
IKE derived key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completioon	Automatic Module Reset	Diffie-Hellman shared secret:Derived From EC Diffie-Hellman shared secret:Derived From
Key derivation key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	KBKDF derived key:Derives
KBKDF derived key	API output parameters (encrypted)	RAM:Plaintext	From service invocation to service completion	Automatic Module Reset	Key derivation key:Derived From

Table 21: SSP Table 2

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
DSA SigVer (FIPS186-4) (A3575)	2048-bit key, SHA2-256	Signature verification	SW/FW Integrity	Module becomes operational	N/A
DSA SigVer (FIPS186-4) (A3584)	2048-bit key, SHA2-256	Signature verification	SW/FW Integrity	Module becomes operational	N/A
DSA SigVer (FIPS186-4) (A3588)	2048-bit key, SHA2-256	Signature verification	SW/FW Integrity	Module becomes operational	N/A

Table 22: Pre-Operational Self-Tests

The pre-operational software integrity tests are performed automatically when the module is powered on before the module transitions into the operational state. The module transitions to the operational state only after the pre-operational self-tests are passed successfully. If any pre-operational self-test fails, the module immediately transitions to the error state.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A3581) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3585) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3587) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3583) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A3575) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3582) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3586) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3581) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3585) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3587) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3583) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3575) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A3582) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A3586) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3581) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3585) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3575) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3581) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3585) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-CBC (A3575) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-KW (A3576) - Encryption	128/192/256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-KW (A3576) - Decryption	128/192/256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KAS-FFC-SSC Sp800-56Ar3 (A3584)	ffdhe2048, MODP-2048	PCT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-FFC-SSC Sp800-56Ar3 (A3575)	ffdhe2048, MODP-2048	PCT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-FFC-SSC Sp800-56Ar3 (A3588)	ffdhe2048, MODP-2048	PCT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A3584)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A3575)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A3588)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
Hash DRBG (A3585)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3587)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3583)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
Hash DRBG (A3584)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3575)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3582)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3586)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
Hash DRBG (A3588)	SHA2-256, without prediction resistance	KAT	CAST	Module becomes operational	Compliant with SP 800-90Ar1	Test runs at power-on before the integrity test
DSA SigVer (FIPS186-4) (A3584)	L=2048, N=224, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
DSA SigVer (FIPS186-4) (A3575)	L=2048, N=224, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
DSA SigVer (FIPS186-4) (A3588)	L=2048, N=224, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA KeyGen (FIPS186-4) (A3584)	SHA2-224	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA KeyGen (FIPS186-4) (A3575)	SHA2-224	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-4) (A3588)	SHA2-224	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA SigGen (FIPS186-4) (A3584)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-4) (A3575)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-4) (A3588)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-4) (A3584)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-4) (A3575)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-4) (A3588)	P-256, SHA2-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
PBKDF (A3584) - HMAC-SHA2-256	HMAC-SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
PBKDF (A3575) - HMAC-SHA2-256	HMAC-SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3588) - HMAC-SHA2-256	HMAC-SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3584) - HMAC-SHA2-384	HMAC-SHA2-384	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3575) - HMAC-SHA2-384	HMAC-SHA2-384	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3588) - HMAC-SHA2-384	HMAC-SHA2-384	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
HMAC-SHA-1 (A3575)	SHA-1	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA-1 (A3588)	SHA-1	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-224 (A3584)	SHA2-224	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-224 (A3575)	SHA2-224	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC-SHA2-224 (A3588)	SHA2-224	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A3584)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A3575)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A3588)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-384 (A3575)	SHA2-384	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-512 (A3575)	SHA2-512	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
KDF IKEv1 (A3579) - HMAC-SHA-1	HMAC-SHA-1	KAT	CAST	Module becomes operational	IKEv1 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv1 (A3579) - HMAC-SHA2-256	HMAC-SHA2-256	KAT	CAST	Module becomes operational	IKEv1 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv1 (A3579) - HMAC-SHA2-384	HMAC-SHA2-384	KAT	CAST	Module becomes operational	IKEv1 Key Derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF IKEv1 (A3579) - HMAC-SHA2-512	HMAC-SHA2-512	KAT	CAST	Module becomes operational	IKEv1 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv2 (A3579) - HMAC-SHA-1	HMAC-SHA-1	KAT	CAST	Module becomes operational	IKEv2 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv2 (A3579) - HMAC-SHA2-256	HMAC-SHA2-256	KAT	CAST	Module becomes operational	IKEv2 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv2 (A3579) - HMAC-SHA2-384	HMAC-SHA2-384	KAT	CAST	Module becomes operational	IKEv2 Key Derivation	Test runs at power-on before the integrity test
KDF IKEv2 (A3579) - HMAC-SHA2-512	HMAC-SHA2-512	KAT	CAST	Module becomes operational	IKEv2 Key Derivation	Test runs at power-on before the integrity test
KDF SP800-108 (A3578)	HMAC-SHA2-256	KAT	CAST	Module becomes operational	Key based key derivation	Test runs at power-on before the integrity test
PBKDF (A3584) - HMAC-SHA-1	HMAC-SHA-1	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3575) - HMAC-SHA-1	HMAC-SHA-1	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test
PBKDF (A3588) - HMAC-SHA-1	HMAC-SHA-1	KAT	CAST	Module becomes operational	Password-based key derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA KeyGen (FIPS186-4) (A3584)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-4) (A3575)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-4) (A3588)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA SigGen (FIPS186-4) (A3584) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3575) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3588) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3584) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3575) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3588) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA SigGen (FIPS186-4) (A3584) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3575) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-4) (A3588) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3584) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3575) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3588) - SHA2-256	PKCS#1 v1.5, 2048-bit key, SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3584) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3575) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3588) - SHA2-384	PKCS#1 v1.5, 2048-bit key, SHA2-384	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA SigVer (FIPS186-4) (A3584) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3575) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4) (A3588) - SHA2-512	PKCS#1 v1.5, 2048-bit key, SHA2-512	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
Safe Primes Key Generation (A3584)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
Safe Primes Key Generation (A3575)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
Safe Primes Key Generation (A3588)		PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
SHA-1 (A3575)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA-1 (A3588)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-224 (A3584)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
SHA2-224 (A3575)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-224 (A3588)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-256 (A3584)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-256 (A3575)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-256 (A3588)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-384 (A3575)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-512 (A3575)		KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
KDF TLS (A3584)		KAT	CAST	Module becomes operational	KDF for TLS v1.0 and v1.1	Test runs at power-on before the integrity test
KDF TLS (A3575)		KAT	CAST	Module becomes operational	KDF for TLS v1.0 and v1.1	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF TLS (A3588)		KAT	CAST	Module becomes operational	KDF for TLS v1.0 and v1.1	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3584) - SHA2-256	SHA2-256	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3575) - SHA2-256	SHA2-256	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3588) - SHA2-256	SHA2-256	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3584) - SHA2-384	SHA2-384	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3575) - SHA2-384	SHA2-384	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3588) - SHA2-384	SHA2-384	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3584) - SHA2-512	SHA2-512	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A3575) - SHA2-512	SHA2-512	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
TLS v1.2 KDF RFC7627 (A3588) - SHA2-512	SHA2-512	KAT	CAST	Module becomes operational	Industry-based TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test

Table 23: Conditional Self-Tests

Data output through the data output interface is inhibited during the conditional self-tests. The module does not return control to the calling application until the tests are completed. If any of these tests fails, the module transitions to the error state.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
DSA SigVer (FIPS186-4) (A3575)	Signature verification	SW/FW Integrity	On Demand	Manually
DSA SigVer (FIPS186-4) (A3584)	Signature verification	SW/FW Integrity	On Demand	Manually
DSA SigVer (FIPS186-4) (A3588)	Signature verification	SW/FW Integrity	On Demand	Manually

Table 24: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A3581) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3585) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3587) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3583) - Encryption	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A3575) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3582) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3586) - Encryption	KAT	CAST	On Demand	Manually
AES-ECB (A3581) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3585) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3587) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3583) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3575) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3582) - Decryption	KAT	CAST	On Demand	Manually
AES-ECB (A3586) - Decryption	KAT	CAST	On Demand	Manually
AES-CBC (A3581) - Encryption	KAT	CAST	On Demand	Manually
AES-CBC (A3585) - Encryption	KAT	CAST	On Demand	Manually
AES-CBC (A3575) - Encryption	KAT	CAST	On Demand	Manually
AES-CBC (A3581) - Decryption	KAT	CAST	On Demand	Manually
AES-CBC (A3585) - Decryption	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-CBC (A3575) - Decryption	KAT	CAST	On Demand	Manually
AES-KW (A3576) - Encryption	KAT	CAST	On Demand	Manually
AES-KW (A3576) - Decryption	KAT	CAST	On Demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A3584)	PCT	CAST	On Demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A3575)	PCT	CAST	On Demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A3588)	PCT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A3584)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A3575)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A3588)	KAT	CAST	On Demand	Manually
Hash DRBG (A3585)	KAT	CAST	On Demand	Manually
Hash DRBG (A3587)	KAT	CAST	On Demand	Manually
Hash DRBG (A3583)	KAT	CAST	On Demand	Manually
Hash DRBG (A3584)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
Hash DRBG (A3575)	KAT	CAST	On Demand	Manually
Hash DRBG (A3582)	KAT	CAST	On Demand	Manually
Hash DRBG (A3586)	KAT	CAST	On Demand	Manually
Hash DRBG (A3588)	KAT	CAST	On Demand	Manually
DSA SigVer (FIPS186-4) (A3584)	KAT	CAST	On Demand	Manually
DSA SigVer (FIPS186-4) (A3575)	KAT	CAST	On Demand	Manually
DSA SigVer (FIPS186-4) (A3588)	KAT	CAST	On Demand	Manually
ECDSA KeyGen (FIPS186-4) (A3584)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-4) (A3575)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-4) (A3588)	PCT	PCT	On Demand	Manually
ECDSA SigGen (FIPS186-4) (A3584)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-4) (A3575)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA SigGen (FIPS186-4) (A3588)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-4) (A3584)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-4) (A3575)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-4) (A3588)	KAT	CAST	On Demand	Manually
PBKDF (A3584) - HMAC-SHA2-256	KAT	CAST	On Demand	Manually
PBKDF (A3575) - HMAC-SHA2-256	KAT	CAST	On Demand	Manually
PBKDF (A3588) - HMAC-SHA2-256	KAT	CAST	On Demand	Manually
PBKDF (A3584) - HMAC-SHA2-384	KAT	CAST	On Demand	Manually
PBKDF (A3575) - HMAC-SHA2-384	KAT	CAST	On Demand	Manually
PBKDF (A3588) - HMAC-SHA2-384	KAT	CAST	On Demand	Manually
HMAC-SHA-1 (A3575)	KAT	CAST	On Demand	Manually
HMAC-SHA-1 (A3588)	KAT	CAST	On Demand	Manually
HMAC-SHA2-224 (A3584)	KAT	CAST	On Demand	Manually
HMAC-SHA2-224 (A3575)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-224 (A3588)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A3584)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A3575)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A3588)	KAT	CAST	On Demand	Manually
HMAC-SHA2-384 (A3575)	KAT	CAST	On Demand	Manually
HMAC-SHA2-512 (A3575)	KAT	CAST	On Demand	Manually
KDF IKEv1 (A3579) - HMAC-SHA-1	KAT	CAST	On Demand	Manually
KDF IKEv1 (A3579) - HMAC-SHA2-256	KAT	CAST	On Demand	Manually
KDF IKEv1 (A3579) - HMAC-SHA2-384	KAT	CAST	On Demand	Manually
KDF IKEv1 (A3579) - HMAC-SHA2-512	KAT	CAST	On Demand	Manually
KDF IKEv2 (A3579) - HMAC-SHA-1	KAT	CAST	On Demand	Manually
KDF IKEv2 (A3579) - HMAC-SHA2-256	KAT	CAST	On Demand	Manually
KDF IKEv2 (A3579) - HMAC-SHA2-384	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF IKEv2 (A3579) - HMAC-SHA2-512	KAT	CAST	On Demand	Manually
KDF SP800-108 (A3578)	KAT	CAST	On Demand	Manually
PBKDF (A3584) - HMAC-SHA-1	KAT	CAST	On Demand	Manually
PBKDF (A3575) - HMAC-SHA-1	KAT	CAST	On Demand	Manually
PBKDF (A3588) - HMAC-SHA-1	KAT	CAST	On Demand	Manually
RSA KeyGen (FIPS186-4) (A3584)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-4) (A3575)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-4) (A3588)	PCT	PCT	On Demand	Manually
RSA SigGen (FIPS186-4) (A3584) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3575) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3588) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
(A3584) - SHA2-384				
RSA SigGen (FIPS186-4) (A3575) - SHA2-384	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3588) - SHA2-384	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3584) - SHA2-512	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3575) - SHA2-512	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-4) (A3588) - SHA2-512	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3584) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3575) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3588) - SHA2-256	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
(A3584) - SHA2-384				
RSA SigVer (FIPS186-4) (A3575) - SHA2-384	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3588) - SHA2-384	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3584) - SHA2-512	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3575) - SHA2-512	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4) (A3588) - SHA2-512	KAT	CAST	On Demand	Manually
Safe Primes Key Generation (A3584)	PCT	PCT	On Demand	Manually
Safe Primes Key Generation (A3575)	PCT	PCT	On Demand	Manually
Safe Primes Key Generation (A3588)	PCT	PCT	On Demand	Manually
SHA-1 (A3575)	KAT	CAST	On Demand	Manually
SHA-1 (A3588)	KAT	CAST	On Demand	Manually
SHA2-224 (A3584)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
SHA2-224 (A3575)	KAT	CAST	On Demand	Manually
SHA2-224 (A3588)	KAT	CAST	On Demand	Manually
SHA2-256 (A3584)	KAT	CAST	On Demand	Manually
SHA2-256 (A3575)	KAT	CAST	On Demand	Manually
SHA2-256 (A3588)	KAT	CAST	On Demand	Manually
SHA2-384 (A3575)	KAT	CAST	On Demand	Manually
SHA2-512 (A3575)	KAT	CAST	On Demand	Manually
KDF TLS (A3584)	KAT	CAST	On Demand	Manually
KDF TLS (A3575)	KAT	CAST	On Demand	Manually
KDF TLS (A3588)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3584) - SHA2-256	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3575) - SHA2-256	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3588) - SHA2-256	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3584) - SHA2-384	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3575) - SHA2-384	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3588) - SHA2-384	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
TLS v1.2 KDF RFC7627 (A3584) - SHA2-512	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3575) - SHA2-512	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A3588) - SHA2-512	KAT	CAST	On Demand	Manually

Table 25: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error	General purpose error state	Failure of a CAST or integrity test	Power-cycle the module	CKR_DEVICE_ERROR error code returned

Table 26: Error States

In the error state, the output interface is inhibited, and the module accepts no more inputs or requests. To recover from the error state, the module must be unloaded and subsequently reloaded, which can be initiated by power-cycling the platform on which the module runs. If failures persist, the module must be re-installed.

10.5 Operator Initiation of Self-Tests

The pre-operational and conditional known-answer self-tests can be executed on-demand by unloading and subsequently re-initializing the module or by invoking the sftk_FIPSRepeatIntegrityCheck() function, which will perform all pre-operational and conditional self-tests. The pair-wise consistency tests can be invoked on demand by requesting a key-pair generation service.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

11.1.1 Module Installation

The Netscape Portable Runtime (NSPR) package (mozilla-nspr-4.23-3.9.1.x86_64.rpm) is a prerequisite for the module. The mozilla-nspr package must be installed in the operating environment.

The Crypto Officer can install the RPM packages containing the module using the zypper tool. The integrity of the RPM package is automatically verified during the installation, and the Crypto Officer shall not install the RPM package if there is any integrity error.

11.1.2 Operating Environment Configuration

The operating environment needs to be configured to support FIPS, so the following steps shall be performed with the root privilege:

1. Install the dracut-fips RPM package:

```
# zypper install dracut-fips
```

2. Recreate the INITRAMFS image:

```
# dracut -f
```

3. After regenerating the initrd, the Crypto Officer has to append the following parameter in the /etc/default/grub configuration file in the GRUB_CMDLINE_LINUX_DEFAULT line:

```
fips=1
```

4. After editing the configuration file, please run the following command to change the setting in the boot loader:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

If /boot or /boot/efi resides on a separate partition, the kernel parameter boot=<partition of /boot or /boot/efi> must be supplied. The partition can be identified with the command "df /boot" or "df /boot/efi" respectively. For example:

```
# df /boot
Filesystem     1K-blocks     Used     Available     Use%     Mounted on
/dev/sda1      233191        30454    190296        14%      /boot
```

The partition of /boot is located on /dev/sda1 in this example. Therefore, the following string needs to be appended in the aforementioned grub file:

```
"boot=/dev/sda1"
```

5. Reboot to apply these settings.

Now, the operating environment is configured to support FIPS operation. The Crypto Officer should check the existence of the file /proc/sys/crypto/fips_enabled, and verify it contains a numeric value “1”. If the file does not exist or does not contain “1”, the operating environment is not configured to support FIPS and the module will not properly operate as a FIPS validated module.

11.1.3 Access to Audit Data

The module may use the Unix syslog function and the audit mechanism provided by the operating system to audit events. Auditing is turned off by default. Auditing capability must be turned on as part of the initialization procedures by setting the environment variable NSS_ENABLE_AUDIT to 1. The Crypto Officer must also configure the operating system's audit mechanism.

The module uses the syslog function to audit events, so the audit data are stored in the system log. Only the root user can modify the system log. On some platforms, only the root user can read the system log; on other platforms, all users can read the system log. The system log is usually under the /var/log directory. The exact location of the system log is specified in the /etc/syslog.conf file. The module uses the default user facility and the info, warning, and err severity levels for its log messages.

The module can also be configured to use the audit mechanism provided by the operating system to audit events. The audit data would then be stored in the system audit log. Only the root user can read or modify the system audit log. To turn on this capability, it is necessary to create a symbolic link from the library file /usr/lib64/libaudit.so.1 to /usr/lib64/libaudit.so.1.0.0.

11.1.4 Module Installation for Vendor Affirmed Platforms

The following table includes the information on module installation process for the vendor affirmed platforms that are listed in Section 2.2.

Product	Link
SUSE Linux Enterprise Micro 5.3	https://documentation.suse.com/sle-micro/5.3/single-html/SLE-Micro-security/#sec-fips-slemicro-install
SUSE Linux Enterprise Server for SAP 15SP4	https://documentation.suse.com/sles/15-SP4/html/SLES-all/book-security.html
SUSE Linux Enterprise Base Container Image 15SP4	https://documentation.suse.com/smart/linux/html/concept-bci/index.html
SUSE Linux Enterprise Desktop 15SP4	https://documentation.suse.com/sled/15-SP4/html/SLED-all/book-security.html
SUSE Linux Enterprise Real Time 15SP4	https://documentation.suse.com/sle-rt/15-SP4/

Table 27 - Installation for Vendor Affirmed Platforms

Note: Per Section 7.9 in the FIPS 140-3 Management Manual, the Cryptographic Module Validation Program (CMVP) makes no statement as to the correct operation of the module or the security strengths of the generated keys when this module is ported and executed in an operational environment not listed on the validation certificate.

11.2 Administrator Guidance

The binaries of the module are contained in the RPM packages for delivery. The Crypto Officer shall follow Section 11.1 to configure the operational environment and install the module to be operated as a FIPS 140-3 validated module.

The following table lists the RPM packages that contain the FIPS validated module. The "Show module name and version" service is implemented by accessing the CKA_NSS_VALIDATION_MODULE_ID attribute of the CKO_NSS_VALIDATION object in the default slot. The object attribute contains the value “SUSE Linux Enterprise NSS 3.79.4-150400.3.29.1” which matches the service output and the version information provided in the RPM packages where the module is distributed. This value maps to version 3.1 of the cryptographic module.

Processor Architecture	RPM Packages
Intel 64-bit	libsoftokn3-3.79.4-150400.3.29.1 .x86_64.rpm libsoftokn3-hmac-3.79.4-150400.3.29.1 .x86_64.rpm libfreebl3-3.79.4-150400.3.29.1 .x86_64.rpm libfreebl3-hmac-3.79.4-150400.3.29.1 .x86_64.rpm
AMD 64-bit	libsoftokn3-3.79.4-150400.3.29.1 .x86_64.rpm libsoftokn3-hmac-3.79.4-150400.3.29.1 .x86_64.rpm libfreebl3-3.79.4-150400.3.29.1 .x86_64.rpm libfreebl3-hmac-3.79.4-150400.3.29.1 .x86_64.rpm
IBM z15	libsoftokn3-3.79.4-150400.3.29.1 .s390x.rpm libsoftokn3-hmac-3.79.4-150400.3.29.1 .s390x.rpm libfreebl3-3.79.4-150400.3.29.1 .s390x.rpm libfreebl3-hmac-3.79.4-150400.3.29.1 .s390x.rpm
ARMv8 64-bit	libsoftokn3-3.79.4-150400.3.29.1 .aarch64.rpm libsoftokn3-hmac-3.79.4-150400.3.29.1 .aarch64.rpm libfreebl3-3.79.4-150400.3.29.1 .aarch64.rpm libfreebl3-hmac-3.79.4-150400.3.29.1 .aarch64.rpm
IBM Power10 64-bit	libsoftokn3-3.79.4-150400.3.29.1 .aarch64.rpm libsoftokn3-hmac-3.79.4-150400.3.29.1 .aarch64.rpm libfreebl3-3.79.4-150400.3.29.1 .aarch64.rpm libfreebl3-hmac-3.79.4-150400.3.29.1 .aarch64.rpm

Table 28 – RPM packages

11.2.1 Considerations for the Approved Mode

In order to run in the approved mode, the module must be operated using the approved services with their corresponding approved and allowed cryptographic algorithms provided in Section 2.5 of this Security Policy. In addition, key sizes must comply with SP 800-131A Rev. 2.

The following module initialization steps must be followed before starting to use the NSS module:

- Set the environment variable NSS_ENABLE_AUDIT to 1.
- Use the FC_GetFunctionList function to obtain pointer references to the API. The function returns a CK_FUNCTION_LIST structure containing function pointers named as the API functions but with the “C_” prefix (e.g. C_Initialize and C_Finalize). The function pointers reference the “FC_” prefixed functions.

- Use FC_Initialize (function pointer C_Initialize) to initialize the module. Ensure that the function returns CKR_OK, which means that the module was properly configured and the power-on self-tests were successful. If the function returns a different code, the module must be reset and initialized again.

The module can be configured to use different private key database formats: key3.db or key4.db. The key3.db format is based on the Berkeley database engine and should not be used concurrently by multiple processes. The key4.db format is based on the SQL database engine and can be used concurrently by multiple processes. Both databases are outside the cryptographic boundary and all data stored in these databases are considered to be stored in plaintext for FIPS validation purposes. The interface code of the NSS cryptographic module which accesses data stored in the database is considered part of the cryptographic boundary.

Secret and private keys, plaintext passwords, and other security-relevant data items are maintained under the control of the cryptographic module. Secret and private keys must be entered into the module by the calling application and output from the module to the calling application in encrypted form using the FC_WrapKey and FC_UnwrapKey functions, respectively. The cryptographic algorithm allowed for this purpose in the approved mode of operation is AES in KW mode.

All cryptographic keys used in the approved mode of operation must be generated in the approved mode or imported while running in the approved mode.

11.3 Non-Administrator Guidance

There is no specific non-administrator guidance.

11.4 End of Life

For secure sanitization of the cryptographic module, the module must first be powered off, which will zeroize all keys and CSPs in volatile memory. Then, for actual deprecation, the module shall be upgraded to a newer version that is FIPS 140-3 validated.

The module does not provide persistent storage of SSPs, so further sanitization steps are not required.

12 Mitigation of Other Attacks

12.1 Attack List

12.1.1 Blinding Against RSA Timing Attacks

RSA is vulnerable to timing attacks. In a setup where attackers can measure the time of RSA decryption or signature operations, blinding must be used to protect the RSA operation from that attack.

The module uses the following blinding technique: instead of using the RSA decryption directly, a blinded value $y = x\, r^e \bmod n$ is decrypted and the unblinded value $x' = y'\, r^{-1} \bmod n$ returned. The blinding value r is a random value with the size of the modulus n.

12.1.2 Cache Invariant Modular Exponentiation

Modular exponentiation used in DSA and RSA is vulnerable to cache-timing attacks. The module implements a variant of the modular exponentiated proposed by Colin Percival to defend against these attacks.

12.1.3 Double-Checking RSA Signatures

Arithmetic errors in RSA signatures might leak the private key. After generating a signature with RSA, the module verifies the signature to ensure no errors occurred during generation.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
API	Application Programming Interface
CAST	Cryptographic Algorithm Self-Test
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CKG	Cryptographic Key Generation
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
HKDF	HMAC-based Key Derivation Function
HMAC	Keyed-Hash Message Authentication Code
IKE	Internet Key Exchange
KAS	Key Agreement Scheme
KAT	Known Answer Test
KDA	Key Derivation Algorithm
KDF	Key Derivation Function
KTS	Key Transport Scheme
KW	Key Wrap
KWP	Key Wrap with Padding
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
PAA	Processor Algorithm Acceleration

PAI	Processor Algorithm Implementation
PCT	Pair-wise Consistency Test
PKCS	Public Key Cryptography Standard
PRF	Pseudo-Random Function
PSP	Public Security Parameter
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SSC	Shared Secret Computation
SSP	Sensitive Security Parameter
TLS	Transport Layer Security

Appendix B. References

FIPS 140-3 **FIPS PUB 140-3 - Security Requirements For Cryptographic Modules**
March 2019
https://doi.org/10.6028/NIST.FIPS.140-3

FIPS 140-3 IG **Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program**
March 2024
https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements

FIPS 197 **Advanced Encryption Standard (AES)**
May 2023
https://doi.org/10.6028/NIST.FIPS.197-upd1

SP 800-38A **Recommendation for Block Cipher Modes of Operation Methods and Techniques**
December 2001
https://doi.org/10.6028/NIST.SP.800-38A

SP 800-38B **Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication**
May 2005
https://doi.org/10.6028/NIST.SP.800-38B

SP 800-38D **Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC**
November 2007
https://doi.org/10.6028/NIST.SP.800-38D

SP 800-38F **Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping**
December 2012
https://doi.org/10.6028/NIST.SP.800-38F

FIPS 180-4 **Secure Hash Standard (SHS)**
August 2015
https://doi.org/10.6028/NIST.FIPS.180-4

FIPS 198-1 **The Keyed-Hash Message Authentication Code (HMAC)**
July 2008
https://doi.org/10.6028/NIST.FIPS.198-1

FIPS 186-4 **Digital Signature Standard (DSS)**
July 2013
https://doi.org/10.6028/NIST.FIPS.186-4

SP 800-132	**Recommendation for Password-Based Key Derivation Part 1: Storage Applications** December 2010 https://doi.org/10.6028/NIST.SP.800-132
SP 800-108 Rev. 1	**Recommendation for Key Derivation Using Pseudorandom Functions** August 2022 https://doi.org/10.6028/NIST.SP.800-108r1-upd1
SP 800-56A Rev. 3	**Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography** April 2018 https://doi.org/10.6028/NIST.SP.800-56Ar3
SP 800-56C Rev. 2	**Recommendation for Key-Derivation Methods in Key-Establishment Schemes** August 2020 https://doi.org/10.6028/NIST.SP.800-56Cr2
SP 800-135 Rev. 1	**Recommendation for Existing Application-Specific Key Derivation Functions** December 2011 https://doi.org/10.6028/NIST.SP.800-135r1
SP 800-90A Rev. 1	**Recommendation for Random Number Generation Using Deterministic Random Bit Generators** June 2015 https://doi.org/10.6028/NIST.SP.800-90Ar1
SP 800-90B	**Recommendation for the Entropy Sources Used for Random Bit Generation** January 2018 https://doi.org/10.6028/NIST.SP.800-90B
SP 800-133 Rev. 2	**Recommendation for Cryptographic Key Generation** June 2020 https://doi.org/10.6028/NIST.SP.800-133r2
SP 800-52 Rev. 2	**Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations** August 2019 https://doi.org/10.6028/NIST.SP.800-52r2
SP 800-131A Rev. 2	**Transitioning the Use of Cryptographic Algorithms and Key Lengths** March 2019 https://doi.org/10.6028/NIST.SP.800-131Ar2
PKCS#1	**Public Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1** February 2003 https://www.ietf.org/rfc/rfc3447.txt