Cisco Systems, Inc.

FIPS 140-3 Non-Proprietary Security Policy

For

Cisco FIPS Object Module

Last Updated: July 30, 2024, Version 1.3

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Table of Contents

List of Tables

List of Figures

1 General

This document is the Non-Proprietary Security Policy for the cryptographic module "Cisco FIPS Object Module", firmware version 7.3a by Cisco Systems, Inc. (hereinafter referred to as FOM or module). This Security Policy is provided in accordance with ISO/IEC 19790 Annex B, FIPS 140-3, and NIST SP800-140B. This Security Policy was prepared as part of the Level 1 FIPS 140-3 validation of the module.

The following table lists the level of validation for each area in the FIPS PUB 140-3.

ISO/IEC 24759 Section 6.	FIPS 140-3 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	1
8	Non-Invasive Security	N/A
9	Sensitive Security Parameter Management	1
10	Self-Tests	1
11	Life-Cycle Assurance	1
12	Mitigation of Other Attacks	N/A
Overall Level		1

Table 1 - Security Levels

2 Cryptographic Module Specification

The Cisco FIPS Object Module (FOM) cryptographic module is a hybrid firmware cryptographic library in a multi-chip standalone embodiment that allows for a vast array of Cisco's networking and collaboration products to use approved algorithms.

The module is intended to run on the tested platforms listed in Table 2 and on other various host platforms, so the physical perimeter of the module is the tested platforms. The cryptographic module comprises Cisco's FIPS Object Module (FOM) cryptographic module (Firmware Version: 7.3a) and the processors (only for algorithm acceleration) and only operates in the approved mode of operation. The module is validated according to FIPS 140-3 at overall security level 1. Please refer to Table 1 above for the individual areas.

The cryptographic module provides the cipher operations and Key Derivation functions to support the following protocols: IKEv2/IPSec, sRTP, SSH, TLS and SNMPv3. Full implementations of these protocols are not supported by the module.

The module has been tested on the following Operational Environments.

#	Operating System	Hardware Platform	Processor	PAA/Acceleration
1	Linux 4.5	Cisco Unified Computing System (UCS)	Intel Xeon Gold 6244 (Cascade Lake) with AES-NI	With PAA
2	Linux 5.4	ISR 4321	Intel Atom C2558 (Silvermont) with AES-NI	With PAA
3	Linux 4.4	Cisco Catalyst 9300	Intel Xeon D-1526 (Broadwell) with AES-NI	With PAA

Table 2 - Tested Operational Environments

In addition to the platforms listed in Table 2, Cisco has also tested the module on the following platforms and claims vendor affirmation on them.

#	Operating System	Hardware Platform
1	Linux 4 (FX-OS)	C220 M5 w/KVM/AWS
2	Linux 4 (FX-OS)	C240 M5 w/ESXi/KVM/AWS
3	Linux 4 (FX-OS)	C480 M5 w/ESXi/KVM/AWS
4	Linux 4 (FX-OS)	E160-M3 w/ESXi/KVM/AWS
5	Linux 4 (FX-OS)	E180D-M3 w/ESXi/KVM/AWS

Table 3 - Vendor Affirmed Operational Environments

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

Modes of Operation

By design, the module is only able to support approved mode of operations following the steps in Section 11 of this document. The module doesn't claim the implementation of a degraded mode operation.

The table below lists all Approved security functions of the module, including specific key size(s) - in bits otherwise noted - employed for approved services, and implemented modes of operation. Only the algorithms, modes, and key sizes that are implemented by the module are shown in this table.

CAVP Cert	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4446	AES [FIPS 197, SP800-38A]	CBC, ECB, CTR, CFB 1/8/128, OFB	Key Length: 128, 192 and 256 bits	Block cipher providing encryption/decryption with data confidentiality from the modes of operation.

A4446	AES [FIPS 197, SP800-38C]	CCM	Key Length: 128, 192 and 256 bits	Block cipher providing confidentiality an authentication through Counter with Cipher Block Chaining-Message Authentication Code
A4446	AES [FIPS 197, SP800-38B]	CMAC	Key Length: 128, 192 and 256 bits	A cipher (AES) based MAC providing authentication, encryption and decryption
A4446	AES [FIPS 197, SP800-38D]	GCM, GMAC	Key Length: 128, 192 and 256 bits	Authentication and encryption. Providing confidentiality of data thru Authentication, encryption and decryption.
A4446	AES [SP800-38F]	KW, KWP	Key Length: 128, 192 and 256 bits	Key wrap/unwrap. Key establishment methodology provides between 128 and 256 bits of encryption strength
A4446	AES [FIPS 197, SP800-38E]	XTS	Key Length: 128 and 256 bits	Authenticated Symmetric Encryption and Decryption; XTS mode is only approved for storage applications per SP800-38E.
A4446	SHS [FIPS 180-4]	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	N/A	Message Digest; non-digital-signature and legacy use for SHA1, all other SHAs acceptable for hash functions applications.
A4446	HMAC [FIPS 198-1]	HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512	Key Length: 112 bits or greater	Integrity based on secret key. Using standard SHA HASH with secret key for calculations and verification.
A4446	CTR_DRBG [SP800-90Arev1]	AES-128/192/256 Derivation Function Enabled; Prediction Resistance: Yes	N/A	Deterministic Random Bit Generators (DRBG); uses an algorithm to produce random output

A4446	Hash_DRBG [SP800-90Arev1]	SHA-1/224/256/384/512	N/A	Deterministic Random Bit Generators (DRBG); uses an algorithm to produce random output
A4446	HMAC_DRBG [SP800-90Arev1]	HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512	N/A	Deterministic Random Bit Generators (DRBG); uses an algorithm to produce random output
A4446	DSA [FIPS 186-4]	DSA KeyGen	Key lengths: 2048, 3072 bits	DSA key generation
A4446	DSA [FIPS 186-4]	DSA PQGGen	Key lengths: 2048, 3072 bits	DSA domain parameter generation
A4446	DSA [FIPS 186-4]	DSA PQGVer	Key lengths: 2048, 3072 bits (PQGVer has Key Length 1024 with SHA-1)	DSA domain parameter verification
A4446	DSA [FIPS 186-4]	DSA SigGen	Key lengths: 2048, 3072 bits	DSA Signature Generation
A4446	DSA [FIPS 186-4]	DSA SigVer	Key lengths: 2048, 3072 bits (SigVer has Key Length 1024 with SHA-1)	DSA signature verification
A4446	ECDSA [FIPS 186-4]	ECDSA KeyGen	Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	Deterministic ECDSA digital signature key pair generation
A4446	ECDSA [FIPS 186-4]	ECDSA KeyVer	Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	Deterministic ECDSA digital signature key pair verification,
A4446	ECDSA [FIPS 186-4]	ECDSA SigGen	Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	Deterministic ECDSA digital signature generation
A4446	ECDSA [FIPS 186-4]	ECDSA SigVer	Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	Deterministic ECDSA digital signature verification, Accept or reject the signature
A4446	RSA [FIPS 186-4]	RSA KeyGen: - Mode: B.3.4	Modulus: 2048/3072/4096 bits	Digital signature key pair consists of an RSA private key, which is

		- 2048/3072/4096 with SHA-256		used to compute a digital signature, and an RSA public key, which is used to verify a digital signature. Key pair shall not be used for other purposes
A4446	RSA [FIPS 186-4]	RSA SigGen: - PKCS1-v1.5 - 2048/3072/4096 bits with SHA-224/256/384/512	Modulus: 2048/3072/4096 bits	Private key to generate a digital signature
A4446	RSA [FIPS 186-4]	RSA SigVer: - PKCS1-v1.5 - 2048/3072/4096 bits with SHA-1/224/256/384/512	Modulus: 1024/2048/3072/4096 bits	RSA signature verification, accept or reject the signature
A4446	KAS (ECC) [SP800-56Arev3]	KAS (ECC): Scheme: ephemeralUnified KAS Role: initiator, responder KAS (KAS-SSC Cert. #A4446)	KAS (ECC): Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521with TLSv1.2 KDF (SP800-135rev1)	Key Agreement Scheme per SP800-56Arev3 with key derivation function (SP800-135rev1) Note: The module's KAS (ECC) implementation is FIPS 140-3 IG D.F Scenario 2 (path 2) compliant
A4446	KAS-ECC CDH Component [SP 800-56Arev3]	KAS-ECC CDH	Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	ECC CDH Primitive used in shared secret computation
A4446	KAS-SSC (ECC) [SP 800-56Arev3]	KAS-ECC-SSC: Scheme: ephemeralUnified: KAS Role: initiator, responder	KAS-ECC-SSC: Curves: P-224, P-256, P-384, P-521	Key establishment methodology provides between 112 and 256 bits of encryption strength
A4446	KAS-SSC (FFC) [SP 800-56Arev3]	KAS-FFC-SSC: Scheme: dhEphem: KAS Role: initiator, responder	KAS-FFC-SSC: FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, modp-2048, modp-3072, modp-4096, modp-6144, modp-8192	Key establishment methodology provides between 112 and 200 bits of encryption strength
A4446	KAS-SSC (IFC) [SP 800-56Brev2]	KAS-IFC-SSC	KAS-IFC-SSC: MODP-2048/3072/4096	Key establishment methodology provides between 112 and 152 bits of encryption strength
A4446	CVL [SP800-135rev1]	SSHv2 KDF (SSH-KDF), TLS v1.2 KDF RFC7627 (SSH-TLS)	N/A	Key Derivation function. SNMPv3,

				sRTP, TLS, SSHv2, IKEv2
A4446	KDA HKDF [SP800-56Crev1]	SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	Key Length: 2048	Subset of Two-Step Key Derivation
A4446	KDA OneStep [SP800-56Crev1]	SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512	Key Length: 2048	Mode of the JSON Key Derivation
A4446	KDF IKEv2 [SP800-135rev1]	SHA-1	Diffie-Hellman Shared Secret Length: 2048 Derived Keying Material Length: 3072	Key Derivation Function for IKEv2
A4446	KDF SNMP [SP800-135rev1]	Shared password	Length 64, 256	Key Derivation Function for SNMP
A4446	KDF SP800-108	HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512	Key Length: 112 bits or greater	Key Derivation Function using HMAC for pseudorandom functions (PRF)
A4446	KDF SRTP [SP800-135rev1]	AES	Key Length: 128, 192, 256	Key Derivation Function for SRTP
A4446	KDF SSH [SP800-135rev1]	AES	Key Length:128, 192, 256	Key Derivation Function for SSH
A4446	KTS-IFC [SP800-56Brev2]	RSA	Modulus: 2048, 3072, 4096	Key Generation and transport
A4446	PBKDF [SP800-132]	HMAC Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512,	N/A	Password based Key Derivation

		SHA3-224, SHA3-256, SHA3-384, SHA3-512,		
A4446	SHS [FIPS 180-4]	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	N/A	Message Digest; non-digital-signature and legacy use for SHA1, all other SHAs acceptable for hash functions applications.
A4446	Safe Prime	Key Generation	ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, modp-2048, modp-3072, modp-4096, modp-6144, modp-8192	Key Generation
A4446	Safe Prime	Key Verification	ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, modp-2048, modp-3072, modp-4096, modp-6144, modp-8192	Key Verification
A4446	Shake [FIPS 202]	Shake-128, Shake-256	Key Length: 128 and 256	Message digest, extendable-output function (XOF) output can be extended to any desired length
A4446	TLS v1.2 KDF RFC7627	SHA2-256, SHA2-384, SHA2-512	N/A	Key Derivation Function for TLS
A4446	TLS v1.3 KDF	HMAC SHA2-256, HMAC SHA3-384	Key Length: 112 bits or greater	Key Derivation Function for TLS
A4446	Triple-DES	CBC, CFB1/CFB8/CFB64, CTR, ECB, OFB, CMAC	Keying option 1	Only for legacy decrypt operation.
None	CKG	N/A	N/A	Key Generation. In accordance with FIPS 140-3 IG D.H, the cryptographic module performs Cryptographic Key Generation as per scenario 1 of section 4 in SP800-133rev2.

Table 4 - Approved Algorithms

Notes:

- The module's AES-GCM implementation conforms to FIPS 140-3 IG C.H techniques 1, 3 and 5 depending on the protocol using it.
- No parts of any protocols, other than the KDFs, have been tested by the CAVP and CMVP

- In accordance with SP 800-132 PBKDF iteration count runs 10-1000 Increment 1. Further in keeping with IG D.N. module used option 1a HMAC with various SHAs
- In accordance with FIPS 140-3 IG D.H, the cryptographic module performs Cryptographic Key Generation as per scenario 1 of section 4 in SP800-133rev2. The resulting generated symmetric key and the seed used in the asymmetric key generation are the unmodified output from SP800-90Arev1 DRBG

As the module can only be operated in the Approved mode of operation, and any algorithms not listed in the Table 4 above will be rejected by the module while in the approved mode, the tables defined in SP800-140B for the following categories are missing from this document:

- Non-Approved Algorithms Allowed in Approved Mode of Operation with No Security Claimed
- Non-Approved Algorithms Not Allowed in the Approved Mode of Operation
- Non-Approved Algorithms Allowed in the Approved Mode of Operation

Cryptographic boundary

The FOM cryptographic module (red dash box) is a non-modifiable, multi-chip standalone hybrid firmware cryptographic module providing cryptographic support which takes data in and out from the host application via the API. All processing is done on the listed processors in Table 2 above. The FOM performs no communications other than with the consuming application. The block diagram below shows the boundary of the Tested Operational Environment's Physical Perimeter (TOEPP) being defined as the physical perimeter of the tested platform enclosure around which everything runs. The cryptographic boundary is the FOM (red dash box) and its interfaces with the operational environment.

Tested Platform TOEPP

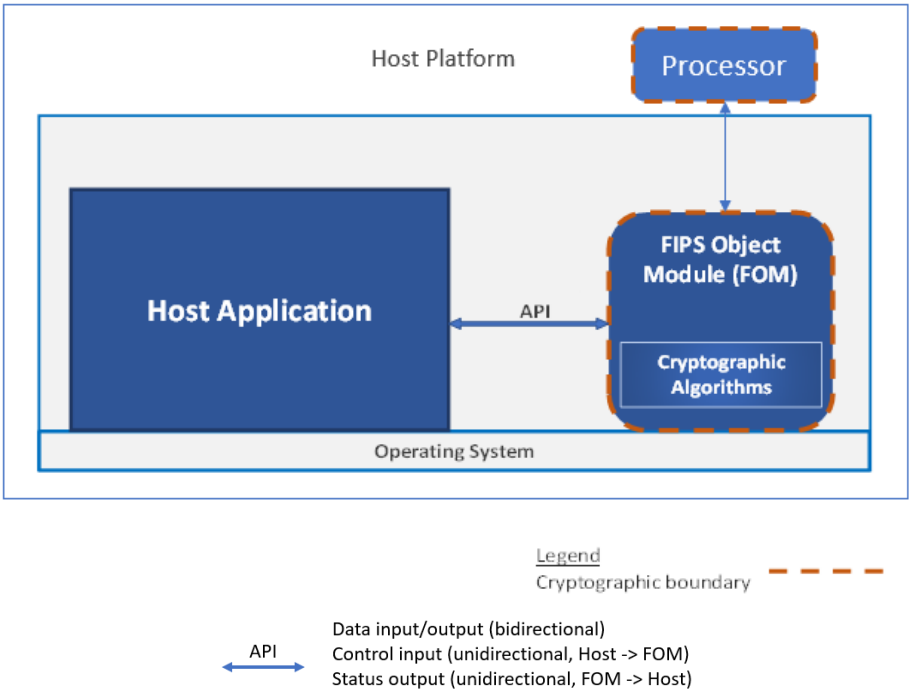

Figure 1 – Block Diagram

3 Cryptographic Module Interfaces

The module's physical perimeter encompasses the case of the tested platform mentioned in Table 2. The module provides its logical interfaces via Application Programming Interface (API) calls. The logical and physical interfaces provided by the module are mapped onto the FIPS 140-3 interfaces (data input, data output, control input, control output and status output) as follows.

Physical Port	Logical Interface	Data that passes over port/interface
Input registers	Data Input Interface	Arguments for an API call that provide the data to be used or processed by the module.
Output registers	Data Output Interface	Arguments output from an API call.
Control registers	Control Input Interface	Arguments for an API call used to control and configure module operation.
Status registers	Status Output Interface	Return values, and or log messages.
Power	N/A	Host platform power supply.

Table 5 - Ports and Interfaces

The control output interface does not apply to the module.

4 Roles, Services, and Authentication

The module supports Crypto Officer (CO) role. The cryptographic module does not provide any authentication methods. The module does not allow concurrent operators. The Crypto Officer is implicitly assumed based on the service requested. The module provides the following services to the Crypto Officer.

Role	Service	Input	Output
Crypto Officer	Show Status	API commands	Module's current status ("FIPS Mode: ON")
Crypto Officer	Perform Self-Tests	Power cycle the host platform	Output on each algorithm running self-test and pass or fail
Crypto Officer	Show Version	API commands	Output the version
Crypto Officer	Configure Security	API commands	Output each approved algorithm available
Crypto Officer	Configure Symmetric Encryption/Decryption	API commands, keys/data	FOM followed by the encryption/decryption in use and ciphertext/plaintext data
Crypto Officer	Shared Secret Computation	API commands, keys/data	FOM followed by the shared secret
Crypto Officer	Configure Signature Generation/Verification	API commands, keys/message/signature	FOM followed by the signature/message
Crypto Officer	Configure Key Generation/Verification	API commands, keys	FOM followed by the key pair, status
Crypto Officer	Configure Key Derivation Function	API commands, secrets/passphrase	FOM followed by the keys
Crypto Officer	Key Wrapping	API commands, wrapping key, key	FOM followed by the crypto key in use being wrapped
Crypto Officer	Configure Keyed Hash	API commands, keys/data	FOM followed by the hash in use and keyed hash output
Crypto Officer	Configure Message Digest	API commands, data	FOM followed by the digest in use and hashed output
Crypto Officer	Configure Random Number Generation	API commands	FOM followed by the random strings in use
Crypto Officer	Perform Zeroisation	API commands	N/A.

Table 6 - Roles, Service Commands, Input and Output

The table below lists all approved services that can be used in the approved mode of operation. The abbreviations of the access rights to keys and SSPs have the following interpretation:

G = Generate: The module generates or derives the SSP.
R = Read: The SSP is read from the module (e.g. the SSP is output).
W = Write: The SSP is updated, imported, or written to the module.
E = Execute: The module uses the SSP in performing a cryptographic operation.
Z = Zeroise: The module zeroises the SSP.
N/A = The service does not access any SSP during its operation.

Along with the global indicator, the return code obtained by `echo $?` can be used to determine whether the last command was successful or not. (0 = successful; anything else = unsuccessful).

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys	Indicator

					and/or SSPs	
Show Status	Provide module's current status (status message)	N/A	N/A	Crypto Officer	N/A	Global Indicator API output as designed by HOST system using the FOM
Perform Self-Test	Execute the FIPS 140 CAST and Health tests outlined in Section 10 below	N/A	AES Key, Authentication	Crypto Officer	E	Global Indicator API output as designed by HOST system using the FOM and output pass or fail
Show Version	Provide module's name and version information	N/A	N/A	Crypto Officer	N/A	Global Indicator API output as designed by HOST system using the FOM and output version, OS and hardware
Configure Symmetric Encryption and Decryption	Configure Symmetric cipher operation	AES (CBC, CFB1, CFB8, CFB128, CTR, ECB, OFB, KW, KWP) XTS-AES A4446	AES key	Crypto Officer	W,E,Z	Global Indicator API output as designed by HOST system using the FOM
Shared Secret Computation	Configure and derive Shared Secret and related Keys	KAS-ECC-SSC, KAS-FFC-SSC, KAS-ECC CDH A4446	Diffie-Hellman Public Key, Diffie-Hellman Private Key, EC Diffie-Hellman Public Key, EC Diffie-Hellman Private Key, Diffie-Hellman Shared Secret, EC Diffie-Hellman Shared Secret	Crypto Officer	W,E,Z	Global Indicator API output as designed by HOST system using the FOM
Configure Keyed Hash	Configure HMAC, CMAC, GMAC usage	HMAC SHA-1/ 224/256/384/ 512, AES 128/192/256 A4446	Authentication	Crypto Officer	W,E,Z	Global Indicator API output as designed by HOST system using the FOM
Configure Message Digest	Configure SHS usage	SHA-1/224/256/384/512, SHA3-224/256/384/512 A4446	None	Crypto Officer	N/A	Global Indicator API output as designed by HOST system using the FOM
Configure Random	Configure DRBG Usage		DRBG entropy input	Crypto Officer	W, E, Z	Global Indicator API output as

Number Generation		DRBG (Hash, HMAC or AES CTR) A4446	DRBG Seed, DRBG V and C, DRBG Key	Crypto Officer	G, E, Z	designed by HOST system using the FOM
Configure Key Derivation Function	Configure Key Derivation	KDA HKDF, IKEv2 KDF, SNMP KDF, SP800-108 KDF, SRTP KDF, SSH KDF, PBKDF, TLSv1.2/1.3 KDF A4446	Key Derivation Function (KDF) secret values	Crypto Officer	G, E, Z	Global Indicator API output as designed by HOST system using the FOM
Configure Key Generation and Verification	Configure Key Generation and Verification	DSA, ECDSA, RSA A4446	DSA Public Key, DSA Private Key, ECDSA Public Key, ECDSA Private Key, RSA Public Key, RSA Private Key	Crypto Officer	G, E, Z	Global Indicator API output as designed by HOST system using the FOM
Key Wrapping	Configure key wrapping	AES-KW, AES-KWP A4446	AES Key	Crypto Officer	W,G, E, Z	Global Indicator API output as designed by HOST system using the FOM.
Configure Signature Generation and Verification	Configure Signature Generation and Verification	DSA, ECDSA, RSA A4446	DSA Public Key, DSA Private Key, ECDSA Public Key, RSA Private Key, RSA Public Key, RSA Private Key	Crypto Officer	W,G, E, Z	Global Indicator API output as designed by HOST system using the FOM
Perform Zeroization	Perform Zeroization	N/A	All SSPs	Crypto Officer	Z	None

Table 7 - Approved Services

5 Software/Firmware Security

Integrity Techniques

The module is provided in the form of binary executable code. To ensure security, the module is protected by HMAC SHA-1 (HMAC Cert. #A4446) algorithm. The firmware integrity test key (Not an SSP) was preloaded to the module's binary the factory and used for firmware integrity test only at the Pre-Operational Self-Test. At module's initialization, the integrity of the runtime executable is verified using a HMAC SHA-1 digest which is compared to a value computed at build time. If at the load time the MAC does not match the stored, known MAC value, the module would enter to an Error state with all crypto functionality inhibited.

The module does not support firmware loading.

Integrity Test On-Demand

Integrity test is performed as part of the Pre-Operational Self-Tests. It is automatically executed at power-on. The operator can power-cycle or reboot the tested platform to initiate the firmware integrity test on-demand.

6 Operational Environment

The module is operated in a non-modifiable operational environment per FIPS 140-3 level 1 specifications.

The cryptographic module has control over its own SSPs. The process and memory management functionality of the host device's OS prevents unauthorized access to plaintext private and secret keys, intermediate key generation values and other SSPs by external processes during module execution. The module only allows access to SSPs through its well-defined API. The operational environments provide the capability to separate individual application processes from each other by preventing uncontrolled access to CSPs and uncontrolled modifications of SSPs regardless of whether this data is in the process memory or stored on persistent storage within the operational environment. Processes that are spawned by the module are owned by the module and are not owned by external processes/operators.

The module's firmware version running on each tested platform is 7.3a.

7 Physical Security

Per FIPS 140-3 classification, this is a multi-chip standalone cryptographic module. Cisco FIPS Object Module (v7.3a) is a hybrid firmware module, which runs on a production grade chassis.

8 Non-invasive Security

Currently, non-invasive security is not required by FIPS 140-3 (see NIST SP 800-140F). The requirements of this area are not applicable to the module.

9 Sensitive Security Parameter Management

The following table summarizes the keys and Sensitive Security Parameters (SSPs) that are used by the cryptographic services implemented in the module.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
DRBG entropy input	>112 bits	N/A	Obtained from the Entropy Source within TOEPP (GPS INT Pathways)	Import to the module via module's API	N/A	N/A: The module does not provide persiste nt	Zeroized when the tested platform is powered down	Random Number Generation

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
				Export: No		keys/SSPs storage.		
DRBG Seed	384 bits	SP800-90Arev1 CTR_DRBG, DRBG_HASH, DRBG_HMAC Cert. #A4446	Generated using DRBG derivation function that includes the entropy input.	Import: No Export: No	N/A	N/A: The module does not provide persistent keys/SSPs storage.	Crypto_free_rng() or Power cycle the device	Internal state of the DRBG.
DRBG V	128 440/888 bits 160/256/384/512 bits	SP800-90Arev1 CTR_DRBG, DRBG_HASH, DRBG_HMAC Cert. #A4446	Generated first during DRBG instantiation and then subsequently updated using the DRBG update function.	Import: No Export: No	Generated internally	N/A: The module does not provide persistent keys/SSPs storage.	Crypto_free_rng() or Power cycle the device	Internal state of the DRBG.
DRBG C	128/192/256 440/888 bits	SP800-90Arev1 CTR_DRBG, DRBG_HASH Cert. #A4446	Generated first during DRBG instantiation and then subsequently updated using the DRBG update function.	Import: No Export: No	Generated internally	N/A: The module does not provide persistent keys/SSPs storage.	Crypto_free_rng() or Power cycle the device	Internal state of the DRBG.
DRBG Key	128/192/256 bits 160/256/384/512 bits	SP800-90Arev1 CTR_DRBG, DRBG_HMAC Cert. #A4446	Established per SP 800-90Arev1 CTR_DRBG and HMAC_DRBG	Import: No Export: No	Generated internally	N/A: The module does not provide persistent keys/SSPs storage.	Crypto_free_rng() or Power cycle the device	Internal state of the DRBG.
AES Key	128,192,256 bits	AES (CBC, CCM, CFB1,	Generated externally and passed	Import: Yes	Generated externally	N/A: The module does not	Crypto_free_cipher() crypto_free_ablkcipher()	AES session key, XTS mode is

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
		CFB128, CFB8, CMAC, CTR, ECB, GCM, GMAC, KW, KWP, OFB, XTS)Cert. #A4446	into the module.	Export: No		provide persistent keys/SSPs storage.	crypto_free_blkcipher() crypto_free_skcipher() crypto_free_aead() or Power cycle the device	only approved for storage applications per SP800-38E. KW/KWP mode is used for key wrapping (KTS).
AES GCM IV	96-bit	AES GCM Cert. #A4446	Generated internally or generated externally in compliance with industry standards then passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	Power cycle the module	Initialization vector for AES GCM
RSA Public Key	112, 128, 152 bits	RSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	Signature verification
RSA Private Key	112, 128, 152 bits	RSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	Signature generation
DSA Public Key	112, 128 bits	DSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent	FIPS_openssl_cleanse() function or power cycle	DSA signature verification

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
						keys/SSPs storage.		
DSA Private Key	112, 128 bits	DSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	DSA signature generation
ECDSA Public Key	112, 128, 192, 256 bits	ECDSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	ECDSA signature verification
ECDSA Private Key	112, 128, 192, 256 bits	ECDSA Cert. #A4446	Internally generated or externally generated and passed into the module.	Import: Yes Export: No	Generated internally or externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	ECDSA signature generation
Diffie-Hellman Private Key	112, 128, 152, 176, 200 bits	KAS-SSC (FFC) KAS-SSC Cert. #A4446	Internally generated conformant to SP800-133r2 (CKG) using SP800-56A rev3 Diffie-Hellman key generation method, and the random value used in key generation is generated using SP800-90Arev1 DRBG	Import: Yes Export: No	N/A	N/A: The module does not provide persistent keys/SSPs storage.	Zeroized when the tested platform is powered down	Used to derive Diffie-Hellman Shared Secret

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
Diffie-Hellman Public Key	112, 128, 152, 176, 200 bits	KAS-SSC (FFC) KAS-SSC Cert. #A4446	Internally derived internally per the Diffie-Hellman key agreement (SP800-56Arev3)	Import: Yes Export: No	N/A	N/A: The module does not provide persiste nt keys/ SSPs storage.	Zeroized when the tested platform is powered down	Used to derive Diffie-Hellman Shared Secret
Diffie-Hellman Shared Secret	N/A	KAS-SSC (FFC) KAS-SSC Cert. #A4446	Internally generated using SP800-56Arev3 DH shared secret computation	Import: Yes Export: No	N/A	N/A: The module does not provide persiste nt keys/SS Ps storage.	Zeroized when the tested platform is powered down	Used to derive SSH, TLS or IPSec/IKE related keys
EC Diffie-Hellman Private Key	112, 128, 192, 256 bits	KAS-SSC (ECC) KAS-SSC Cert. #A4446	Internally generated conformant to SP800-133r2 (CKG) using SP800-56A rev3 EC Diffie-Hellman key generation method, and the random value used in key generation is generated using SP800-90Arev1 DRBG	Import: Yes Export: No	N/A	N/A: The module does not provide persiste nt keys/ SSPs storage.	Zeroized when the tested platform is powered down	Used to derive EC Diffie-Hellman Shared Secret
EC Diffie-Hellman Public Key	112, 128, 192, 256 bits	KAS-SSC (ECC) KAS-SSC Cert. #A4446	Internally derived internally per the EC Diffie-Hellman key agreement (SP800-56Arev3)	Import: Yes Export: No	N/A	N/A: The module does not provide persiste nt keys/ SSPs storage.	Zeroized when the tested platform is powered down	Used to derive EC Diffie-Hellman Shared Secret
EC Diffie-Hellman Shared Secret	N/A	KAS-SSC (ECC)	Internally generated using SP800-56Arev3	Import: Yes	N/A	N/A: The module does not	Zeroized when the tested platform is	Used to derive TLS or IPSec/IKE

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroisation	Use & Related Keys
		KAS-SSC Cert. #A4446	ECDH shared secret computation	Export: No		provide persistent keys/SSPs storage.	powered down	related keys
RSA Key Wrapping/ Transport Key	112, 128, 152 bits	RSA Cert. #A4446	Generated externally and passed into the module.	Import: Yes Export: No	Generated externally	N/A: The module does not provide persistent keys/SSPs storage.	FIPS_openssl_cleanse() function or power cycle	RSA Key Transport/ Wrapping
Authentication	128 to 256 bits	HMAC (SHA1, SHA224, SHA256, SHA384, SHA512), AES-CMAC, AES-GMAC Cert. #A4446	Generated externally and passed into the module.	Import: Yes Export: No	Generated externally	N/A: The module does not provide persistent keys/SSPs storage.	Crypto_free_shash() crypto_free_ahash() or Power cycle the device	Integrity assurance
Key Derivation Function (KDF) secret values	N/A	IKEv2, SNMP, SRTP, SSH,800-108 KDF Cert. #A4446	Generated internally	Import: No Export: No	Generated internally	N/A: The module does not provide persistent keys/SSPs storage	FIPS_openssl_cleanse() function or power cycle	Deriving keys per SP800-135rev1 and SP800-56Crev1
Firmware Integrity Key (not a SSP)	128 bits	HMAC-SHA-1 Cert. #A4446	Pre-loaded at the factory (in the module's binary)	Import: No Export: No	N/A	Stored in the module binary computed during build.	This key is used for firmware integrity test and not subject to key zeroization requirements according to FIPS140-3 IG 9.7.B.	Used for firmware integrity test. This is not a SSP

Table 8 - SSPs

The module uses approved DRBG for the generation of random strings and passes them to the calling application only upon their request. The cryptographic module is passed a pointer to the cryptographic keys as API parameters, associated by memory location. The application calling the cryptographic module passes keys in plaintext within the physical perimeter. The module does not perform storage of keys. All SSPs can be zeroized by power cycling the host.

Note 1: Use of external IV with GCM is exclusively permitted for decryption operations or where the GCM is used to support the protocol specific implementation used to protect connections to the calling applications.

Note 2: The check for Key_1 ≠ Key_2 is done before using the keys in the XTS-AES algorithm to process data and is in accordance with IG C.I requirements.

Note 3: No parts of SSH, TLS, SNMPv3, sRTP and IKE protocols, other than the KDFs, have been tested by the CAVP and CMVP.

Entropy sources	Minimum number of bits of entropy	Details
Entropy within the TOEPP was passively loaded into the module to seed the 800-90Arev1 DRBG by the Operating System	At least 112 bits	While operating in the approved mode, the entropy and seeding material for the SP800-90Arev1 DRBG are provided by the external calling application (and not by the module) which is outside the module's cryptographic boundary but contained within the module's Tested Operational Environment's Physical Perimeter (TOEPP) boundary. The module receives a LOAD command with entropy obtained from the entropy source (Intel CPU processor with instructions RDRand) inside the TOEPP. The minimum effective strength of the SP 800-90Arev1 DRBG seed is required to be at least 112 bits when used in an approved mode of operation, therefore the minimum number of bits of entropy requested when the module makes a call to the SP 800-90Arev1 DRBG is at least 112 bits. Per the IG 9.3.A Entropy Caveats, the following caveat applies: *No assurance of the minimum strength of generated SSPs (e.g., keys)*.

Table 9 - Non-Deterministic Random Number Generation Specification

10 Self-Tests

When the module is loaded or instantiated (after being powered off, rebooted, etc.), the module runs Pre-Operational Self-Tests. The operating system is responsible for the initialization process and loading of the module. Prior to the module providing any data output via the data output interface, the module would perform and pass the Pre-Operational Self-Tests. Following the successful Pre-Operational Self-Tests, the module would execute the Conditional Cryptographic Algorithm Self-tests (CASTs).

The self-test success or failure messages were logged, which functions as the self-test status indicator. If any one of the self-tests fails, the module transitions into an error state and outputs the error message via the module's status output interface. The module has one error state, called Hard error state. When a self-test fails, the module outputs "POST Failed" error. While the

module is in the error state, all data through the data output interface and all cryptographic operations are disabled. The error state can only be cleared by reloading the module. All self-tests must be completed successfully before the module transitions to the operational state.

Below are the details of the self-tests conducted by the module.

Pre-Operational Self-Tests

- Firmware Integrity Test (HMAC-SHA-1)

The module conducts HMAC-SHA-1 KAT self-test before the integrity test is performed.

Conditional Self-Test

- Conditional Cryptographic Algorithm Self-Tests (CASTs)
    - AES-ECB 128-bit Encrypt KAT
    - AES-ECB 128-bit Decrypt KAT
    - AES-CCM 128-bit Encrypt KAT
    - AES-CCM 128-bit Decrypt KAT
    - AES-CCM 192-bit Encrypt KAT
    - AES-CCM 192-bit Decrypt KAT
    - AES-GCM 256-bit Encrypt KAT
    - AES-GCM 256-bit Decrypt KAT
    - AES-CMAC 128-bit Encrypt KAT
    - AES-CMAC 128-bit Decrypt KAT
    - AES-CMAC 192-bit Encrypt KAT
    - AES-CMAC 192-bit Decrypt KAT
    - AES-CMAC 256-bit Encrypt KAT
    - AES-CMAC 256-bit Decrypt KAT
    - AES-XTS 128-bit Encrypt KAT
    - AES-XTS 128-bit Decrypt KAT
    - AES-XTS 256-bit Encrypt KAT
    - AES-XTS 256-bit Decrypt KAT
    - DRBG AES CTR 128-bit KAT (Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG AES CTR 192-bit KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG AES CTR 256-bit KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG HMAC SHA-1KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG HMAC SHA-224 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG HMAC SHA-256 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG HMAC SHA-384 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
    - DRBG HMAC SHA-512 KAT(Health Tests: Generate, Reseed, Instantiate

functions per Section 11.3 of SP 800-90Arev1)
  - DRBG Hash SHA-1 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
  - DRBG Hash SHA-224 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
  - DRBG Hash SHA-256 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
  - DRBG Hash SHA-384 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
  - DRBG Hash SHA-512 KAT(Health Tests: Generate, Reseed, Instantiate functions per Section 11.3 of SP 800-90Arev1)
  - FIPS 186-4 DSA 2048-bit Sign KAT
  - FIPS 186-4 DSA 2048-bit Verify KAT
  - FIPS 186-4 ECDSA P-256 Sign KAT
  - FIPS 186-4 ECDSA P-256 Verify KAT
  - FIPS 186-4 RSA 2048-bit Sign KAT
  - FIPS 186-4 RSA 2048-bit Verify KAT
  - RSA IFC 2048-bit Encrypt KAT
  - RSA IFC 2048-bit Decrypt KAT
  - SHA-1 KAT
  - SHA3-256 KAT
  - HMAC-SHA-1 KAT
  - HMAC-SHA-224 KAT
  - HMAC-SHA-256 KAT
  - HMAC-SHA-384 KAT
  - HMAC-SHA-512 KAT
  - KAS-ECC-SSC Primitive Z KAT
  - KAS-ECC-SSC Primitive Z KAT
  - SP800-108 KDF (KBKDF)
  - SP800-56Crev1 One Step KDF KAT
  - SP800-56Crev1 HKDF KAT
  - SP800-132 PBKDF KAT
  - TLS 1.3 KDF KAT
  - SP800-135rev1 KDF KAT
    - SSHv2
    - SNMPv3
    - sRTP
    - TLS1.2
    - IKEv2

Conditional Pair-Wise Consistency Test (PCTs):

- RSA PCT.
- ECDSA PCT
- ECDH PCT

- DH PCT
- DSA PCT

FIPS_mode_set() is a function that checks the initialization sequence and the aforementioned self-tests have completed successfully.

Periodic/On-Demand Self-Tests

The module performs on-demand self-tests initiated by the operator, by powering off and powering the module back on through power cycling the host. The full suite of self-tests is then executed. The same procedure may be employed by the operator to perform periodic self-tests.

11 Life-Cycle Assurance

Secure Operation

The tested operating systems segregate user processes into separate process spaces. Each process space is an independent virtual memory area that is logically separated from all other processes by the operating system software and hardware. The module functions entirely within the process space of the process that invokes it. The module operates only in approved mode of operation. There is no non-approved mode of operation for the module.

Secure Initialization

The Operating System loads the module into its user space. The initialization sequence starts with a check of the integrity of the runtime executable using a HMAC-SHA1 digest computed at build time. If this computed HMAC-SHA1 digest matches the stored known digest then the POSTs, consisting of the algorithm specific Known Answer Tests, are performed. If any component of the POST fails an internal global error flag is set to prevent subsequent invocation of any cryptographic function calls. Any such POST failure is a hard error that can only be recovered by reinstalling the module.
Upon loading the cryptographic module, the consuming application must enable the approved mode of operation by calling the “FIPS_mode_set()” function. This function call verifies the POST outcome and returns a “1” for success and “0” for failure; interpretation of this return code is the responsibility of the host application. The function call “. /openssl version -a” returns the name, version of the module and approved mode status.
The module is installed using one of the sets of instructions in the ‘README.Cisco’ document appropriate to the target system available in the repository with the source code. The module does not support Non-Compliant state.

User Guidance

AES GCM IV Generation

In the case of AES-GCM, the IV generation method is user-selectable, and the value can be computed in more than one manner as follows:

1) **TLS 1.2:** The module's AES-GCM implementation conforms to IG C.H, scenario #1, following RFC 5288. The counter portion of the IV is set by the module within its cryptographic boundary. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition will trigger a handshake to establish a new encryption key in accordance with RFC 5246 for TLS 1.2, respectively. The module supports ciphersuites acceptable per SP 800-52rev2.
2) **IKEv2:** The module's AES-GCM implementations conforms to IG C.H, scenario #1 following RFC 7296 for IPSec/IKEv2. The AES GCM IV is generated according to RFC5282. The counter portion of the IV is set by the module within its cryptographic boundary. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition will trigger a handshake to establish a new encryption key. In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption shall be established.
3) **TLS 1.3:** The module's AES-GCM implementations conforms to IG C.H Scenario#5. The module is compatible with TLS v1.3 and provides support for the acceptable GCM cipher suites from Section 8.4 of RFC 8446 and confirms that the IV is generated and used within the protocol's implementation. The counter portion of the IV is set by the module within its cryptographic boundary. In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption must be established.
4) **Non-protocol specific usage:** The module's AES-GCM implementation conforms to IG C.H, scenario #3, when operating in an approved mode of operation, AES GCM, IVs are generated both internally and deterministically and are a minimum of 96-bits in length, which contain a nonce of 32-bits as specified in SP 800-38D, Section 8.2.1.

The selection of the IV construction method is the responsibility of the user of this cryptographic module. Counter mechanism set during IV initialization. When it gets to 95% the protocol gets notified. At 100% the connection is blocked.
AES GCM encryption is used in the context of the TLS protocol versions 1.2 and 1.3. To meet the AES GCM (key/IV) pair uniqueness requirements from NIST SP 800-38D, the module generates the IV as follows:

For TLS v1.2, the module supports acceptable AES GCM cipher suites from section 3.3.1 of NIST SP 800-52rev2. Per scenario 1 in FIPS 140-3 IG C.H, the mechanism for IV generation is compliant with RFC 5288. The implementation of the nonce_explicit management logic inside the module shall ensure that when the nonce_explicit part of the IV exhausts the maximum number of possible values for a given session key (e.g., a 64-bit counter starting from 0 and increasing, when it reaches the maximum value of 264 -1), either party (the client or the server) that encounters this:

- module operator (i.e., the first party, client, or server) to trigger this handshake in accordance with RFC 5246 when this condition is encountered.

The module supports internal IV generation using the module's Approved DRBG. The IV is at least 96 bits in length per section 8.2.2 of NIST SP 800-38D. Per NIST SP 800-38D and scenario 5 of FIPS 140-3 IG C.H, the DRBG generates outputs such that the (key/IV) pair collision probability is less than 2^{-32}.
In the event that power to the module is lost and subsequently restored, the calling application must ensure that any AES-GCM keys used for encryption or decryption are re-distributed.

PBKDF

In line with the requirements for SP 800-132, keys generated using the approved PBKDF must only be used for storage applications. Any other use of the approved PBKDF is non-conformant. In approved mode the module enforces that any password used must encode to at least 14 bytes (112 bits) and that the salt is at least 16 bytes (128 bits) long. The iteration count associated with the PBKDF should be as large as practical.
As the module is a general-purpose hybrid firmware module, it is not possible to anticipate all the levels of use for the PBKDF, however a user of the module should also note that a password should at least contain enough entropy to be unguessable and also contain enough entropy to reflect the security strength required for the key being generated.

The module uses PBKDF option 1a from section 5.4 of NIST SP 800-132.

- The iteration count shall be selected as large as possible, as long as the time required to generate the resultant key is acceptable for module operators. The iteration count shall be 10 up to 1000.

- The length of the passphrase used in the PBKDF shall be 14 up to 128, and shall consist of lower-case, upper-case, and numeric characters.

- Passphrases (used as an input for the PBKDF) shall not be used as cryptographic keys.

AES-XTS

The length of a single data unit encrypted or decrypted with the AES-XTS shall not exceed 2^{20} AES blocks; that is, 16 MB of data per AES-XTS instance. An XTS instance is defined in Section 4 of NIST SP 800-38E.

The AES-XTS mode shall only be used for the cryptographic protection of data on storage devices. The AES-XTS shall not be used for other purposes, such as the encryption of data in transit. The module implements the check to ensure that the two AES keys used in the XTS-AES algorithm are not identical.

12 Mitigation of Other Attacks

The requirements under INCITS+ISO+IEC 19790+2012[2014], section 7.12 “Mitigation of other attacks”, are not applicable to the module since the module currently doesn’t support any mitigation of other attacks services.