Fortanix SDKMS Appliance (FX2200, Version 3.10.16)

FIPS 140-2 Level 3 Non-Proprietary Security Policy

Date: 05/02/2021
Version Number: 2.1

Table of Contents

Table of Figures

Table of Tables

1. Module Overview

Fortanix SDKMS appliance is the building block for running Fortanix Self-Defending Key Management Service™ (SDKMS), a unified HSM and Key Management solution. With SDKMS, you can securely generate, store, and use cryptographic keys and certificates, as well as secrets, such as passwords, API keys, tokens, or any blob of data. SDKMS ensures that you remain in complete control over your keys and secrets. Your business-critical applications and containers can integrate with SDKMS using legacy cryptographic interfaces or using its native RESTful interface. SDKMS provides control of and visibility into your key management operations using a centralized web-based UI with enterprise level access controls and comprehensive auditing. SDKMS is built to scale horizontally and geographically as your demand for managing your keys and secrets increase, while providing automated load-balancing and high availability.

FIPS 140-2 conformance testing was performed at Security Level 3. The following configuration was tested by the lab.

Module Name and Version	Firmware Version
Fortanix SDKMS Appliance (FX2200)	3.10.16

Table 1 - Configurations tested

FIPS Security Area	Security Level
Cryptographic Module Specification	3
Cryptographic Module Ports and Interfaces	3
Roles, Services and Authentication	3
Finite State Model	3
Physical Security	3
Operational Environment	N/A
Cryptographic Key Management	3
EMI/EMC	3
Self-tests	3
Design Assurance	3

FIPS Security Area	Security Level
Mitigation of Other Attacks	N/A
Overall Security Level	3

Table 2- Module Security Level Statement

1.1 Cryptographic Boundary

The cryptographic boundary of the module is the enclosure that contains components of the module. The strong enclosure of the cryptographic module is opaque within the visible spectrum. The module uses tamper evident labels to provide the evidence of tampering. The module contains tamper response and zeroization circuitry.

Figure 1 - Fortanix SDKMS Appliance (FX2200)

The module ships with a separate removable bezel. Bezel is not part of cryptographic boundary. Following picture shows the module with the bezel added.

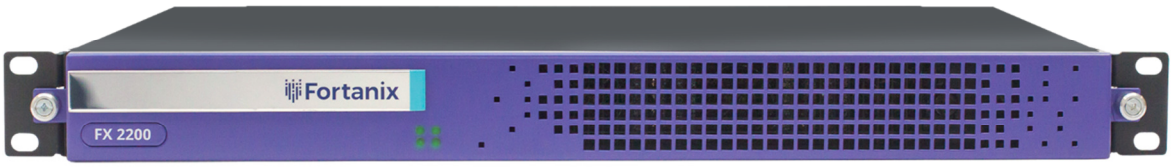

Figure 2 - Fortanix SDKMS Appliance (FX2200) with bezel

2. Modes of Operations

The module always operates in the FIPS approved mode. The Crypto Officer shall follow these steps to verify the module is running in the FIPS Approved Mode:

1. Invoke the version API provided by the "Get status" service
2. Verify that the output is correct, with the following format and value of "fips_level" attribute is 3:

```
{
   "version":"3.10.16",
   "api_version":"v1-20170718",
   "server_mode":"Sgx",
   "fips_level":3
}
```

2.1 Approved Cryptographic Functions

There are some algorithm modes that were tested but not implemented by the module. Only the algorithms, modes, and key sizes that are implemented by the module are shown in this table.

CAVP Cert #	Algorithm	Standard	Model/ Method	Key Lengths, Curves or Moduli	Use
5282	AES	FIPS 197, SP 800-38F SP 800-38C, SP 800-38D, SP 800-38G, SP 800-38B	ECB, CBC, OFB, CTR, CFB 128, GCM, CCM, KW, KWP, CMAC, FF1[1]	128, 192, 256	Data Encryption/ Decryption KTS (AES Cert. #5282 and HMAC Cert. #3489; key establishment methodology provides 256 bits of encryption strength) Message Authentication
1875	CVL[2] TLS 1.2	SP 800-135	SHA-256 SHA-384		Key Derivation
2115	DRBG[3]	SP 800-90A	CTR_DRBG with		Deterministic

[1] FF1 is a method for format-preserving encryption (FPE). It is vendor affirmed. FF1 supports radix values of 2 to 36, min length is based on radix such that $radix^{minlen} >= 100$, max length is 2^{16} and Max tweak length (maxTlen) is 2^{16}.

[2] All API calls into the module are done over TLS V1.2. No parts of these protocols, other than the KDFs, have been tested by the CAVP and CMVP.

[3] DRBG is seeded with minimum 459 bits of entropy by the module for use in key generation.

CAVP Cert #	Algorithm	Standard	Model/ Method	Key Lengths, Curves or Moduli	Use
			derivation function and AES-256		Random Bit Generation
1441 1874 (CVL)	ECDSA	FIPS 186-4	SHA-1 SHA-256 SHA-384 SHA-512	P-192[4], P-224, P-256, P-384, P-521	Key Pair Generation and Signature Verification – Cert # 1441, Digital Signature Generation - Cert # 1874
3489	HMAC	FIPS 198-1	HMAC-SHA-1 HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	112, 128, 160, 192, 256, 384	Message Authentication KTS (AES Cert. #5282 and HMAC Cert. #3489; key establishment methodology provides 256 bits of encryption strength)
203	KBKDF	SP 800-108			Key Derivation
2904	RSA	FIPS 186-2[5]	PKCS1 v1.5;	1024[6], 2048,	Key Generation[7],

[4] Module does not allow P-192 and/or SHA-1 for ECDSA signature generation. The minimum hash sizes allowed by the module are SHA-256 for P-224, SHA-256 for P-256, SHA-384 for P-384, and SHA-512 for P-521. Module does not allow key pair generation with P-192.

[5] For FIPS186-2, only RSA signature generation with key length of 4096 bit and RSA signature verification are implemented.

CAVP Cert #	Algorithm	Standard	Model/ Method	Key Lengths, Curves or Moduli	Use
		FIPS 186-4	GenKey9.31; PSS SHA-1, SHA-256, SHA-384, SHA-512	3072, 4096	Digital Signature Generation and Verification
4241	SHS	FIPS 180-4	SHA-1, SHA-256, SHA-384 SHA-512		Message Digest
C1461	SHA-3	FIPS 202	SHA3-224, SHA3-256, SHA3-384, SHA3-512		Message Digest
1873	CVL Partial DH	SP 800-56A	ECC SHA-512	P-224, P-256, P-384, P-521	Shared Secret Computation
CKG (vendor affirmation)	Cryptographic Key Generation	SP 800-133			Key Generation[8]

[6] Module does not allow 1024-bit keys and/or SHA-1 for RSA Signature Generation.

[7] Minimum RSA key size generated by the module is 2048.

[8] Module directly uses the output of the DRBG. The resulting generated symmetric key and/or generated seed for asymmetric key generation are from the unmodified output of SP 800-90A DRBG.

Table 3 - Table of Approved Algorithms

The module complies with FIPS 140-2 IG A.5 requirements for AES-GCM:

1. For TLS V1.2 Protocol, the module constructs the IV (internally) as allowed per Technique #1 in FIPS 140-2 IG A.5 for Industry Protocols. The AES-GCM implementation complies with RFC 5288 and SP 800-52. The IV total length is 96-bits, where the fixed IV length is 32-bits and `nonce_explicit` part of the IV is a 64-bit counter. The GCM key and IV are session specific; if the module loses power the implementation re-initializes a TLS V1.2 session, creating a new IV altogether. The implementation ensures that when the counter exceeds the maximum value, the session is renegotiated, and new keys are established. AES GCM is only used with in TLS 1.2.
2. For the Encrypt/Decrypt service, a 96-bit IV is constructed from the output of the CTR_DRBG, allowed as per Technique #2 in FIPS 140-2 IG A.5 for IVs generated "internally at its entirety randomly". In case the module's power is lost and then restored, a new IV for use with the AES GCM encryption/decryption will be generated from the output of the CTR_DRBG.

2.2 Non-FIPS Approved But Allowed Cryptographic Functions

Algorithm	Caveat	Use
NDRNG	Only used to seed the CTR_DRBG with derivation function.	Seeding for the Approved DRBG
RSA Key Wrapping	Provides between 112 and 201 bits of encryption strength.	Used for key encapsulation
EC DH	Provides between 112 and 256 bits of encryption strength	Calculate a shared secret

Table 4 - Table of Non-Approved but Allowed Algorithms

2.3 All other algorithms

Algorithm	Use
PBKDF (No security claimed)	Used for obfuscation of authentication data, considered as plaintext.

Table 5 – All Other Algorithms

3. Ports and Interfaces

The following table describes physical ports and logical interfaces of the module.

Port Name	Count	Interface(s)
Ethernet Ports	2	Data Input, Data Output, Control Input, Status Output
IPMI Port	1	Chassis management
VGA Port	1	Data Output, Status Output
Serial Port	1	Data Input, Data Output, Control Input, Status Output
USB Port	3	Data Input, Control Input
Power Receptacle	2	Power Input
Network Link LEDs	2	Status Output
Hard Disk Activity LED	1	Status Output
Power Supply Failure Indicator LED	1	Status Output
Power button	1	Control Input
ID button	1	Control Input
Reset button	1	Control Input
Front panel display LCD	1	Status output

Table 6- Ports and Interfaces

The logical interfaces are implemented as application programming interfaces (API). The logical interfaces of the module offer services. The applications interacting with the module input control information and data to the module through the input fields of the API and receive output data and/or status information via the output parameters of the API. API documentation describes in detail the successful operation output and error in case of a failed operation. Each of the FIPS 140-2 logical interfaces relates to the module's application programming interface as follows:

Logical Interface	Description
Data Input	Input / Request payload of API
Data Output	Output / Response payload of API
Control Input	API call
Status Output	API returning status information and return status codes provided by API Status output via console Status output via LEDs

Table 7- Specification of Cryptographic Module Logical Interfaces

4. Roles, Services and Authentication

The module supports identity-based authentication for all operators. The module supports a Crypto Officer and User Role.

- The Crypto Officer installs and administers the module.
- The User uses the cryptographic services provided by the module. This role is assumed both by an actual user of the system and an external system that requires cryptographic services.

The module supports a variety of roles that are mapped to the two FIPS roles. Following table enumerates the mapping between module roles and FIPS roles:

Module Role	FIPS Role
System Administrator	Crypto Officer
System Operator	Crypto Officer
Account Administrator	Crypto Officer, User
Account Member	Crypto Officer, User
Account Auditor	Crypto Officer
Group Administrator	Crypto Officer, User
Group Auditor	Crypto Officer
Application	User
Node	Crypto Officer

Table 8 – Mapping of Module Roles to FIPS roles

4.1 Services

The module provides the following services:

Service	Corresponding Roles	Types of Access to Cryptographic Keys and CSPs R – Read or Execute W – Write or Create Z – Zeroize
Zeroization	Crypto Officer	All: Z
Firmware update[9]	Crypto Officer	Firmware update key: R
Module Configuration	Crypto Officer	N/A
Random Number Generation	User	DRBG seed R, W
Create/Generate key	User	DRBG seed R, W Key: W
Encrypt/Decrypt	User	AES key: R
Sign/Verify	User	RSA keys: R ECDSA keys: R
Wrap/Unwrap	User	AES key: R RSA keys: R
HMAC	User	HMAC key: R
Digest	User	N/A
Derive Key	User	Symmetric Keys: R, W
Import Key	User	Any key: W TLS Keys: R
Export Key	User	Exportable keys: R TLS Keys: R
Run self-tests	Does not require	N/A

[9] Only CMVP validated version can be used for upgrade.

Service	Corresponding Roles	Types of Access to Cryptographic Keys and CSPs R – Read or Execute W – Write or Create Z – Zeroize
	assumption of a role	
Get status	Does not require assumption of a role	N/A
Platform setup	Does not require assumption of a role	N/A

Table 9 - Services Authorized for Roles

4.2 Authentication

The module supports the following authentication mechanisms.

Module Role	Authentication Type	Authentication Data
System Administrator System Operator Account Administrator Account Member Account Auditor Group Administrator Group Auditor	Identity Based	User name and password

Module Role	Authentication Type	Authentication Data
System Administrator System Operator Account Administrator Account Member Account Auditor Group Administrator Group Auditor	Identity Based	JWT
Application	Identity Based	API key
Application	Identity Based	RSA Public key of external Application
Application	Identity Based	Trusted CA certificate
Application	Identity Based	JWT
System Administrator System Operator Account Administrator Account Member Account Auditor Group Administrator Group Auditor	Identity Based	User 2FA device public key
System Administrator System Operator Account Administrator Account Member Account Auditor Group Administrator Group Auditor Application	Identity Based	Bearer token
Node	Identity Based	Public key of an outside entity/server (Another SDKMS node)

Table 10- Roles and required Identification and Authentication

Our password authentication policy is as described for the Memorized Secret Authenticators in NIST SP 800-63B (8 characters or longer). The module supports concurrent operators and the module levies a restriction on session expiry time where if inactive, the Application's role session will expire in 60 minutes by default. Similarly, for all other Module roles there is a session expiry time of 24 hours. Session expiry time can be customized.

Authentication Mechanism	Strength of Mechanism
User name and password	Minimum password length is 8 characters. For a user who just meets the minimum password length, each of the eight characters will have at least 95 possible characters if we consider just the printable characters, although module supports UTF-8 characters for password and the number of possible characters with UTF-8 is much higher. Total number of password permutations with eight characters is 95^8 = 6,634,204,312,890,625. Therefore, the probability of guessing a password is significantly less than one in 1,000,000. Module only allows at the most 10 authentication attempts in a second. Therefore, a user could try at most 600 passwords in a minute. Given the total number of possible permutations (as shown above), the probability a random attempt in one-minute period to be correct will be 600/6,634,204,312,890,625. Therefore, the probability of guessing a password in a one-minute period is significantly less than one in 100,000.
API key	An application authenticates using an API key which contains application Id and application secret. App secret is a 64 bytes random data. Total number of permutations for app secret will be 2^512. Therefore, the probability of guessing an application's secret is significantly less than one in 1,000,000. Module only allows at the most 10 authentication attempts in a second. Therefore, a user could try at most 600 attempts in a minute. Given the total number of possible permutations (as shown above), the probability a random attempt in one-minute period to be correct will be 600/(2^512). Therefore, the probability of guessing an app

Authentication Mechanism	Strength of Mechanism
	secrete in a one minute period is significantly less than one in 100,000.
User 2FA device public key	The module allows users to use a second factor authentication mechanism in addition to username and password. The strength of this combination mechanism relies upon the strength of the User password mechanism (described earlier) combined with the strength of two factor authentication. This mechanism adds more strength to the password mechanism which already far exceeds the FIPS requirements. U2F signature verification uses U2F device's public key which is an EC P-256 key. Security strength of this key is 128 bits. So, the probability of a random success will be 1 in 2^128. Probability of this combined scheme = (Probability of guessing username and password) * (Probability from signature verification scheme), which is 1/(95^8) * 1/(2^128). Therefore, the probability of guessing a password is significantly less than one in 1,000,000. Module only allows at the most 10 authentication attempts in a second. Therefore, a user could try at most 600 attempts in a minute. Given the total number of possible permutations (as shown above), the probability a random attempt in one-minute period to be correct will be 600/(95^8 * 2^128). Therefore, the probability of guessing a password in a one-minute period is significantly less than one in 100,000. Therefore, this mechanism of additional 2FA also far exceeds the FIPS requirements.
RSA Public key of external Application	The strength of this mechanism is based on the size of the private key space. The module relies upon minimum RSA 2048-bit keys. This provides an encryption strength of 112 bits, so the probability of a random success will be 1 in

Authentication Mechanism	Strength of Mechanism
	2^112, which is significantly less than one in 1,000,000. Using this mechanism, one can make very few attempts in one-minute period. Each attempt will require the module to check the signature on the certificate using FIPS approved signature algorithm and establishing TLS session with this certificate. On an average only one attempt can be made in a second. Therefore, at the most 60 attempts can be made in a one minute period. Therefore, the probability of guessing a 2048-bit private key and succeeding in a one minute period is 60/(2^112) which is significantly less than one in 100,000.
Trusted CA Certificate	The strength of this mechanism is based on the size of the private key space. The module relies upon minimum RSA 2048-bit keys. This provides an encryption strength of 112 bits, so the probability of a random success will be 1 in 2^112, which is significantly less than one in 1,000,000. Using this mechanism, one can make very few attempts in one-minute period. Each attempt will require the module to check the signature on the certificate using FIPS approved signature algorithm and establishing TLS session with this certificate. On an average only one attempt can be made in a second. Therefore, at the most 60 attempts can be made in a one minute period. Therefore, the probability of guessing a 2048-bit private key and succeeding in a one minute period is 60/(2^112) which is significantly less than one in 100,000.
JWT	The strength of this mechanism is based on the size of the private key space. The module relies upon minimum RSA 2048-bit keys. This provides an encryption strength of 112 bits, so the probability of a random success will be 1 in

Authentication Mechanism	Strength of Mechanism
	2^112, which is significantly less than one in 1,000,000. Using this mechanism, one can make very few attempts in one-minute period. Each attempt will require the module to check the signature on the JWT using FIPS approved signature algorithm. On an average only 4 attempts can be made in a second. Therefore, at the most 240 attempts can be made in a one minute period. Therefore, the probability of guessing a 2048-bit private key and succeeding in a one minute period is 240/(2^112) which is significantly less than one in 100,000.
Bearer token	This authentication mechanism builds upon other authentication mechanisms and it maps to the original authentication credentials that were used to establish an authenticated session. The bearer token is a base64 encoded random 64 bytes data which is generated using approved DRBG in SDKMS. Total number of permutations is 2^512. Therefore, the probability of guessing the token is 1/(2^512), which is significantly less than one in 1,000,000. Each authentication attempt takes approximately 12ms or more. Therefore, a user could try at most 5,000 attempts in a minute. Given the total number of possible permutations (as shown above), the probability a random attempt in one-minute period to be correct will be 5000/(2^512). Therefore, the probability of guessing a password in a one-minute period is significantly less than one in 100,000.
Public key of an outside entity/server (Another SDKMS node)	The strength of this mechanism is based on the size of the private key space. The module relies upon RSA 2048-bit node keys. This provides an encryption strength of 112 bits, so the probability of a random success will be 1 in 2^112,

Authentication Mechanism	Strength of Mechanism
	which is significantly less than one in 1,000,000. Each attempt will require the module to check the signature on the certificate using FIPS approved signature algorithm and establishing TLS session with this certificate. Each attempt takes 100ms or more. Therefore, at the most 600 attempts can be made in a one minute period. Therefore, the probability of guessing a 2048-bit private key and succeeding in a one minute period is 600/(2^112) which is significantly less than one in 100,000.

Table 11 - Strength of Authentication Mechanisms

5. Secure Operation Rules

5.1 Module Initialization and Setup

The Crypto Officer is required to follow the vendor procedural control guidelines to setup and install the module after it is received. Here is a brief summary of the procedure. For more information, please refer to user guide.

1. Module unpacking must be done in a secure location where only authorized personnel have access.
2. The installation must be carried out by authorized personnel who has crypto officer role in the organization. The installation must be carried out in a secure location which is accessible only by authorized personnel.

3. The module is automatically installed in FIPS mode and no special steps are required to put the module in FIPS mode. General installation steps are described in Fortanix User Guide – SDKMS FX200 FIPS version 2.4, Section 6.
4. Once setup is complete, run version command to check firmware version and verify FIPS mode.

6. Self-tests

The module performs the following power-up and conditional self-tests. Upon successful execution of **all** power-up self-test, module provides the following status:

"*Software Integrity test succeeded*"
"*Power-up self-tests succeeded*"

Upon failure of a power-up or conditional self-test, the module halts its operation and enters the error state. The following tables describe self-tests implemented by the module along with status messages.

6.1 Power-Up Self Tests

Algorithm	Test	Status
AES 128-bit key size in ECB, CBC, CFB128, and CTR Modes 192-bit key size ECB, CBC, and CFB128 Modes 256-bit key size ECB, CBC, and CFB128 Modes	KAT (encryption)	Success: "*Power-up self-tests succeeded*" Error: "*AES self test failed*"
AES 128-bit key size in ECB, CBC, CFB128, and CTR Modes 192-bit key size ECB, CBC, and CFB128 Modes 256-bit key size ECB, CBC, and CFB128 Modes	KAT (decryption)	Success: "*Power-up self-tests succeeded*" Error: "*AES self test failed*"
AES GCM 128-bit, 192-bit, and 256-bit key size	KAT (encryption)	Success: "*Power-up self-tests succeeded*" Error: "*GCM self test failed*"

Algorithm	Test	Status
AES GCM 128-bit, 192-bit, and 256-bit key size	KAT (decryption)	Success: "*Power-up self-tests succeeded*" Error: "*GCM self test failed*"
AES CCM 128-bit key size	KAT (encryption)	Success: "*Power-up self-tests succeeded*" Error: "*CCM self test failed*"
AES CCM 128-bit key size	KAT (decryption)	Success: "*Power-up self-tests succeeded*" Error: "*CCM self test failed*"
AES KW 128-bit, 192-bit, and 256-bit key size	KAT (Wrap and Unwrap)	Success: "*Power-up self-tests succeeded*" Error: "*KW/KWP self test failed*"
AES KWP 128-bit, 192-bit, and 256-bit key size	KAT (Wrap and Unwrap)	Success: "*Power-up self-tests succeeded*" Error: "*KW/KWP self test failed*"
ECC CDH Primitive "Z" P-224 Curve	KAT	Success: "*Power-up self-tests succeeded*" Error: "*KAS ECC Primitive Z test failed*"
SHA-1	KAT	Success: "*Power-up self-tests succeeded*" Error: "*SHA1 self test failed*"
SHA-256	KAT	Success: "*Power-up self-tests succeeded*" Error: "*SHA256 self test failed*"
SHA-512	KAT	Success: "*Power-up self-tests succeeded*" Error: "*SHA512 self test failed*"
HMAC-SHA-1	KAT	Success: "*Power-up self-tests succeeded*"

Algorithm	Test	Status
128-bit key size		Error: *"HMAC SHA1 self test failed"*
HMAC-SHA-256 128-bit key size	KAT	Success: "*Power-up self-tests succeeded"* Error: *"HMAC SHA256 self test failed"*
HMAC-SHA-512 2048-bit key size	KAT	Success: "*Power-up self-tests succeeded"* Error: *"HMAC SHA512 self test failed"*
SP 800-90A DRBG	KAT	Success: "*Power-up self-tests succeeded"* Error: *"CTR DRBG self test failed"*
RSA 2048-bit key size, SHA-256 (PKCS1 v1.5)	Signature generation/verification KAT	Success: "*Power-up self-tests succeeded"* Error: *"RSA self test failed"*
ECDSA P-224 curve, SHA-256	Signature generation/verification pairwise consistency test	Success: "*Power-up self-tests succeeded"* Error: *"ECDSA self test failed"*
SP 800-135 TLS V1.2 KDF	KAT	Success: "*Power-up self-tests succeeded"* Error: *"TLS 1.2 KDF self test failed"*
SP 800-108 KDF 256-bit key size	KAT	Success: "*Power-up self-tests succeeded"* Error: *"KDF108 self test failed"*
HMAC-SHA-256 256-bit key size	Firmware integrity test	Success: *"Software Integrity test succeeded"* Error: *"Software integrity check failed"*
FF1	KAT (Encypt and Decrypt)	Success: "*Power-up self-tests succeeded"* Error: *"FF1 self test failed"*
CMAC 128-bit, 192-bit, and 256-bit key size	KAT	Success: "*Power-up self-tests succeeded"* Error: "*CMAC self test failed"*

Algorithm	Test	Status
SHA3 SHA3-224, SHA3-256, SHA3-384, SHA3-512	KAT	Success: *"Power-up self-tests succeeded"* Error: *"SHA3 self test failed"*

Table 12 – Power-Up Self-tests

6.2 Conditional Self Tests

Algorithm	Test	Status
Continuous RNG test performed on output of NDRNG	Continuous Random Number Generator (RNG) Test	Error: *"FIPS conditional test failure: Error in cryptographic operation – RNG failed"*
Continuous RNG test performed on output of software-based Approved SP 800-90A CTR_DRBG	Continuous Random Number Generator (RNG) Test	Error: *"FIPS conditional test failure: Error in cryptographic operation – RNG failed"*
RSA with SHA-256	Pairwise Consistency Test (Sign and Verify, Encrypt and Decrypt)	Error: *"FIPS conditional test failure: Pairwise consistency test failed. Sign / Verify test failed."* Error: *"FIPS conditional test failure: Pairwise consistency test failed. Encryption / Decryption test failed."*
ECDSA SHA-256	Pairwise Consistency Test (Sign and Verify)	Error: *"FIPS conditional test failure: Pairwise consistency test failed. Sign / Verify test failed."*
Firmware Load Test ECDSA P-224	Signature verification test	Error: *"Firmware verification failed."*

Algorithm	Test	Status
SHA-256		

Table 13 - Conditional Self-tests

7. Cryptographic Keys and CSPs

The module does not support the import or export of unprotected CSPs. The table below describes cryptographic keys and CSPs used by the module.

Keys / CSPs & Description	Type	Generation / Establishment	Storage	Zeroization
Firmware update key Public key used to validate the signature of firmware update	ECDSA P-256	Generated externally and loaded at build time	Plaintext in persistent storage	N/A
Personalization Key AES 256 bits This key is used to derive the persisted sealing key	AES	SP 800-90A CTR_DRBG	Plaintext in battery backed memory.	On tamper detection.
Persisted Sealing Key AES 256 bits This key is used to wrap top level keys (Cluster Master key and node private key)	AES	Derived from personalization key using NIST SP 800-108 KDF in Feedback Mode (§5.2);	Not stored persistently	Effectively zeroized on tamper due to zeroization of personalization key.

Cluster Master key Key Derivation Key 256 bits This key is used to derive System key and Account Wrapping key	SP 800-108 KDF (Key derivation key)	SP 800-90A CTR_DRBG	This key is stored wrapped using persisted sealing key. Wrapping is done using AES GCM.	Effectively zeroized on tamper due to zeroization of personalization key.
System key AES 256 bits This key is used to wrap all user and session information that is stored in persistent storage	AES GCM	Derived from Cluster Master key using NIST SP 800-108 KDF in Feedback Mode (§5.2);	Not stored persistently	Effectively zeroized on tamper due to zeroization of personalization key.
Account Wrapping key AES 256 bits This key is used to wrap Account key when it is stored in persistent storage	AES GCM	Derived from Cluster Master key using NIST SP 800-108 KDF in Feedback Mode (§5.2)	Not stored persistently	Effectively zeroized on tamper due to zeroization of personalization key.
Account key Key Derivation Key 256-bits This key is used to derive Database Wrapping key and Cipher State Wrapping key	SP 800-108 KDF (Key derivation key)	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 Account Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
Database Wrapping key AES 256 bits This key is used to wrap all account / tenant data and keys that belong to a specific account / tenant when it is stored in persistent storage	AES GCM	Derived from Account key using NIST SP 800-108 KDF in Feedback Mode (§5.2)	Not stored persistently	Effectively zeroized on tamper due to zeroization of personalization key.

Cipher State Wrapping key AES 128 bits This key is used to wrap all cipher state data that belong to a specific account / tenant.	AES GCM	Derived from Account key using NIST SP 800-108 KDF in Feedback Mode (§5.2)	Not stored persistently	Effectively zeroized on tamper due to zeroization of personalization key.
Symmetric key AES – 128,192,256	AES ECB, CBC, CTR, CFB 128, OFB, GCM, CCM, KW, KWP, CMAC, FF1	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
HMAC key HMAC-SHA-1: 112-bit minimum key HMAC-SHA-256: 128-bit minimum key HMAC-SHA-384: 192-bit minimum key HMAC-SHA-512: 256-bit minimum key	HMAC	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
RSA private key for Digital Signatures RSA – 2048 to 8192	RSA	SP 800-90A CTR_DRBG; this key is used for Digital Signature Generation.	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.

RSA private key for Key Encapsulation Operations RSA – 2048 to 8192	RSA	SP 800-90A CTR_DRBG; this key is used for Key Un-encapsulation (decryption) operations.	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
ECDSA private key EC – P-224, P-256, P-384, P-521	EC	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
ECDSA random number "k" k = 224-bits (P-224) k = 256-bits (P-256) k = 384-bits (P-384) k = 521-bits (P-521)	EC	SP 800-90A CTR_DRBG	Not stored persistently	Power cycle
ECCDH private key EC – P-224, P-256, P-384, P-521	EC	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
Cluster RSA private key for TLS RSA – 2048 bits	RSA	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 System key	Effectively zeroized on tamper due to zeroization of personalization key.

SP800-135 TLS KDF internal state [128-byte internal state]	TLS v1.2 KDF (HMAC-SHA-256 PRF or HMAC-SHA-384 PRF)	N/A	Not stored persistently	Power cycle
TLS integrity key (HMAC)	HMAC HMAC-SHA-384 (384-bit key)	Derived from TLS master secret using SP 800-135 KDF	Not stored persistently	Keys are destroyed when session is teared down.
TLS encryption key (AES)	AES AES-256-GCM	Derived from TLS master secret using SP 800-135 KDF	Not stored persistently	Keys are destroyed when session is teared down.
TLS pre-master secret [48-bytes]	Random data	SP 800-90A CTR_DRBG; generated only when the module behaves as a TLS Client.	Not stored persistently	Keys are destroyed when session is teared down.

TLS master secret [48-bytes]	Random data	Derived from TLS pre-master secret using SP 800-135 KDF	Not stored persistently	Keys are destroyed when session is teared down.
AgreeKey shared secret Z P-224 = 224-bit Z P-256 = 256-bit Z P-384 = 384-bit Z P-521 = 528-bit Z (rounded to nearest byte)	Shared Secret	N/A	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
CTR_DRBG CSPs: entropy input, V and Key	DRBG	Internally generated by the NDRNG/DRBG	Not stored persistently	Power cycle
SP 800-108 KDF internal state 256-bit internal state	SP 800-108 KDF in Feedback Mode (§5.2) with HMAC-SHA-256	SP 800-108 KDF in Feedback Mode (§5.2)	Not stored persistently	Zeroized when the function completes
Node RSA private key for SDKMS RSA – 2048 bits	RSA	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 persisted sealing key	Effectively zeroized on tamper due to zeroization of personalization key.
User password Minimum 8 bytes	String of ASCII characters	N/A - Entered by user	Encrypted in persistent storage with AES-GCM-256 System key	Effectively zeroized on tamper due to zeroization of personalization key.

API key 64 bytes	Application Authentication Data	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 Database Wrapping key	Effectively zeroized on tamper due to zeroization of personalization key.
Bearer token 64 bytes	Authentication Data	SP 800-90A CTR_DRBG	Encrypted in persistent storage with AES-GCM-256 System key	Effectively zeroized on tamper due to zeroization of personalization key.

Table 14 - Cryptographic Keys and CSPs

8. Physical Security

The cryptographic module consists of production-grade components. The strong enclosure of the cryptographic module is opaque within the visible spectrum. The removable covers are protected with tamper-evident seals. The tamper-evident seals must be checked periodically by the Crypto Officer. If the tamper-evident seals are broken or missing, the Crypto Officer must halt the operation of the module and ship the module to Fortanix for replacement.

The module contains tamper response and zeroization circuitry. The tamper response and zeroization circuitry immediately zeroizes all plaintext secret and private keys and CSPs when a cover is removed. The tamper response and zeroization circuitry remains operational when plaintext secret and private cryptographic keys or CSPs are contained within the cryptographic module. Ventilation holes are constructed in a manner that prevents undetected physical probing inside the enclosure.

8.1 Inspection/Testing of Physical Security Mechanisms

The following guidelines should be considered when producing an Operational Policy for the environment for which the module is deployed.

The SDKMS appliance enclosure should be periodically checked by the Crypto Officer for evidence of tampering damage to the three tamper-evident labels and any physical damage to the enclosure material.

The frequency of a physical inspection depends upon the information being protected and the environment in which the unit is located. At a minimum, it would be expected that a physical inspection would be made by the Crypto Officer at least monthly.

The tamper evident labels are applied at the Fortanix manufacturing facility, are serialized, and are not available for order or replacement from Fortanix. The labels are designed and intended to stay intact for the entire life of the module. The labels are applied in the three positions shown in the figure below.

Figure 3 - Tamper Evident Label Positions

Following figure shows the tamper label. It leaves “VOID” markings in place of tamper label and the tamper label cannot be reapplied.

Figure 4 - Tamper Evident Label

Two tamper seals sit over a screw on the lid and extend over the lid seam to the module chassis, as shown in the figure below. One tamper seal sits over a screw on the front panel and extends to front chassis body. The only way to remove the cover is to break or damage the tamper seals.

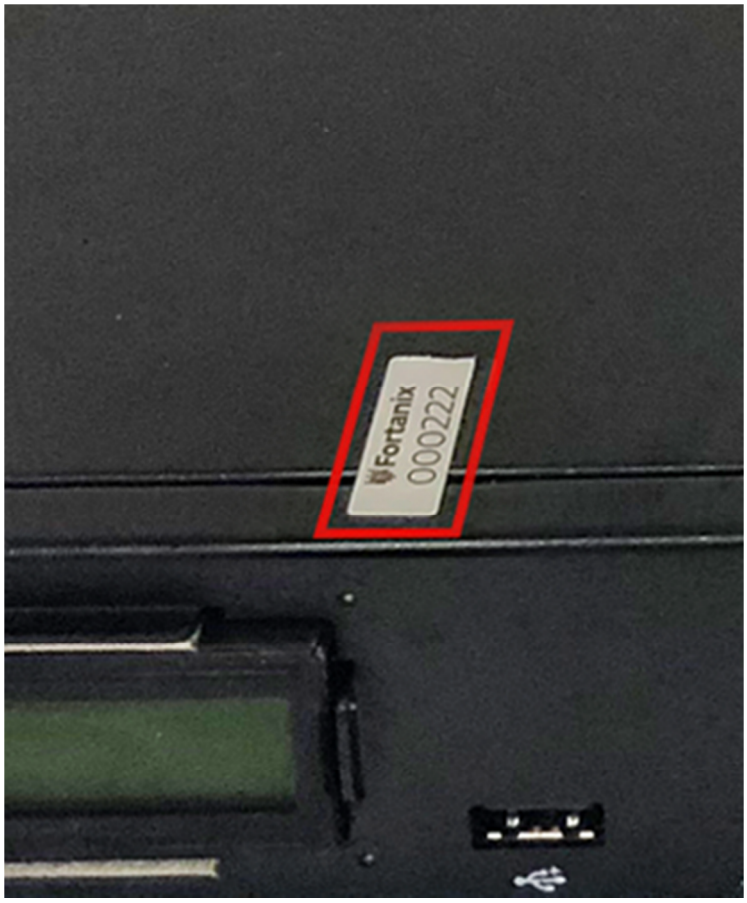

Figure 5 - Tamper Evident Label Closeup

9. Appendix A: Acronyms

TERM	DESCRIPTION
AES	Advanced Encryption Standard (FIPS-197)
API	Application Programming Interface
CBC	Cipher Block Chaining
CTR	Counter
CO	Crypto Officer
CMAC	Cipher-based Message Authentication Code
DRBG	Deterministic Random Bit Generator (SP 800-90Ar1)
EMI/EMC	Electromagnetic Interference/Electromagnetic Compatibility
FIPS	Federal Information Processing Standards
FIPS 140-2 IG	Federal Information Processing Standards 140-2 Implementation Guidance
FF1	FPE mode
FPE	Format Preserving Encryption
GCM	Galois/Counter Mode
HMAC	Keyed-hash Message Authentication Code (FIPS 198-1)
IV	Initialization Vector
JWT	JSON Web Token
KAT	Known Answer Test
KW	AES Key Wrap
KWP	AES Key Wrap with Padding
N/A	Not Applicable
NDRNG	Non-deterministic random number generator
OFB	Output Feedback
RAM	Random-access Memory
RBG	Random Bit Generator
RNG	Random Number Generator
SDKMS	Self-Defending Key Management Service™
SHA-1	Secure Hash Algorithm 1 (FIPS 180-4)
SHA-3	Secure Hash Algorithm 3 (FIPS 202)
USB	Universal Serial Bus
VGA	Video Graphics Array

Table 15 - Specification of acronyms and their descriptions

10. Appendix B: References

Reference	Specification
[ANS X9.31]	Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)
[FIPS 140-2]	Security Requirements for Cryptographic modules, May 25, 2001
[FIPS 180-4]	Secure Hash Standard (SHS)
[FIPS 186-2/4]	Digital Signature Standard
[FIPS 197]	Advanced Encryption Standard
[FIPS 198-1]	The Keyed-Hash Message Authentication Code (HMAC)
[FIPS 202]	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions
[PKCS#1 v2.1]	RSA Cryptography Standard
[PKCS#5]	Password-Based Cryptography Standard
[PKCS#12]	Personal Information Exchange Syntax Standard
[SP 800-38A]	Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode
[SP 800-38B]	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
[SP 800-38C]	Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality
[SP 800-38D]	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC
[SP 800-38F]	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping
[SP 800-38G]	Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption
[SP 800-56A]	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography
[SP 800-56B]	Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography
[SP 800-56C]	Recommendation for Key Derivation through Extraction-then-Expansion

Reference	Specification
[SP 800-67R1]	Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher
[SP 800-89]	Recommendation for Obtaining Assurances for Digital Signature Applications
[SP 800-90A]	Recommendation for Random Number Generation Using Deterministic Random Bit Generators
[SP 800-108]	Recommendation for Key Derivation Using Pseudorandom Functions
[SP 800-132]	Recommendation for Password-Based Key Derivation
[SP 800-135]	Recommendation for Existing Application –Specific Key Derivation Functions

Table 16 - References