FIPS 140-3 Non-Proprietary Security Policy

AWS Key Management Service HSM
Hardware version 3.0, firmware version 1.8.104

Document Version 0.35

October 25, 2024

Table of Contents

List of Tables

List of Figures

1. General

This document defines the Non-Proprietary Security Policy for the AWS Key Management Service HSM module by Amazon Web Services, Inc. The module meets the FIPS 140-3 overall Level 3 requirements. Table 1 lists the security level of for each area in the FIPS 140-3 validation:

ISO/IEC 24759 Section 6	FIPS 140-3 Section Title	Security Level
1	General	3
2	Cryptographic module specification	3
3	Cryptographic module interfaces	3
4	Roles, services, and authentication	3
5	Software/Firmware security	3
6	Operational environment	N/A
7	Physical security	3
8	Non-invasive security	N/A
9	Sensitive security parameter management	3
10	Self-tests	3
11	Life-cycle assurance	3
12	Mitigation of other attacks	N/A

Table 1 – Security Levels

2. Cryptographic Module Specification

The AWS Key Management Service HSM is used exclusively by AWS as a component of the AWS Key Management Service (KMS). The module is not directly accessible to customers of KMS. The cryptographic functions of the module are used to fulfill requests under specific public AWS KMS APIs.

The module runs firmware versions 1.8.104 on hardware version 3.0 and is classified as a Hardware module with a multi-chip standalone embodiment. The cryptographic boundary is defined as the module case, and the module runs on a non-modifiable operating environment. The module follows the initialization/installation requirements found in Section 11.

Model	Hardware [Part Number and Version]	Firmware Version	Distinguishing Features
AWS Key Management Service HSM	3.0	1.8.104	DC power input. No maintenance cover

Table 2 - Cryptographic Module Tested Configuration

The AWS Key Management Service HSM operates only in an Approved mode of operation. The module does not support any non-approved algorithms not allowed in the Approved mode of operation.

The module's cryptographic algorithm implementations have received the following certificate numbers from the Cryptographic Algorithm Validation Program (CAVP). Although additional modes and key lengths were included in the CAVP algorithm testing, the table below represents the actual modes and key lengths used by the services of the module.

CAVP Cert[1]	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
AWS Key Management Service Cryptographic Library				
A1908	AES FIPS 197, SP 800-38A	ECB, CBC, CTR	Direction: Decrypt, Encrypt Key Length: 128, 256	Encryption, Decryption
A1908	GCM[2] SP 800-38D	AES	GCM: Direction: Decrypt, Encrypt IV Generation: External[3] IV Generation Mode: 8.2.2 Key Length: 128, 256 Tag Length: 96, 128 IV Length: 96 Payload Length: 64, 128, 192 AAD Length: 128, 256	Generation, Authentication, Encryption, Decryption

[1] There are algorithms, modes, and key/moduli sizes that have been CAVP-tested but are not used by any approved service of the module. Only the algorithms, modes/methods, and key lengths/curves/moduli shown in this table are used by an approved service of the module.

[2] Per IG C.H (Scenario 2), IVs are internally generated using an approved DRBG, with length of 96 bits (per SP 800-38D).

[3] The IV generation is internal to the module, but external to the algorithm boundary

CAVP Cert[1]	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A1908	KTS SP 800-38F per IG D.G	AES KWP	Direction: Decrypt, Encrypt Cipher: Cipher Key Length: 256 Payload Length: 128, 192, 512	Key Transport using AES KWP
A1908	KTS SP 800-38D and SP 800-38F per IG D.G	AES GCM	Direction: Decrypt, Encrypt Cipher: Cipher Key Length: 256 Payload Length: 160, 256, 384, 512, 2048, 3072, 4096	Key Transport using AES GCM
A1908	DRBG SP 800-90A	CTR DRBG	Capabilities: Mode: AES-256 Derivation Function Enabled: Yes Additional Input: 384 Entropy Input: 384 Nonce: 384 Personalization String Length: 384 Returned Bits: 512	Random Bit Generation
A1908	ECDSA FIPS 186-4	KeyGen	Curve: P-256, P-384, P-521 Secret Generation Mode: Extra Bits, Testing Candidates	Key Pair Generation
		KeyVer	Curve: P-256, P-384, P-521	Public Key Validation
		SigGen Component	Curve: P-256, P-384, P-521 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512	Signature Generation Component
		SigGen	Curve: P-256, P-384, P-521 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512	Signature Generation
		SigVer	Curve: P-256, P-384, P-521 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512	Signature Verification
A1908	HMAC FIPS 198-1	SHA-1	MAC: 80-160 Increment 8 Key Length: 160	Generation, Authentica-tion
		SHA2-256	MAC: 128-256 Increment 8 Key Length: 256	
		SHA2-384	MAC: 192-384 Increment 8 Key Length: 384	
		SHA2-512	MAC: 256-512 Increment 8 Key Length: 512	

CAVP Cert[1]	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A1908	RSA FIPS 186-4	KeyGen	Capabilities: Key Generation Mode: B.3.3 Properties: Modulo: 2048, 3072, 4096 Primality Tests: Table C.2 Properties: Modulo: 2048, 3072, 4096 Primality Tests: Table C.3 Public Exponent Mode: Random Private Key Format: Chinese Remainder Theorem	Key Pair Generation
		SigGen	Signature Type: PKCSPSS Properties: Modulo: 2048, 3072, 4096 (Note: All supported modulus sizes have been algorithm tested according to IG C.F) Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 0 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 0 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 0	Signature Generation
		SigVer	Signature Type: PKCSPSS Properties: Modulo: 2048, 3072, 4096 (Note: All supported modulus sizes have been algorithm tested according to IG C.F) Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 0 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 0 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 0	Signature Verification
		Decryption Primitive	Modulo Length: 2048	Component Test
		Signature Primitive	Private Key Format: standard Public Exponent Mode: random	Signature Generation Component

CAVP Cert[1]	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A1908	SHS FIPS 180-4	SHA-1	Message Length: 0-65536 Increment 8	non-Digital Signature Applications
		SHA2-256	Message Length: 0-65536 Increment 8	Digital Signature Generation and Verification
		SHA2-384	Message Length: 0-65536 Increment 8	Digital Signature Generation and Verification
		SHA2-512	Message Length: 0-65536 Increment 8	Digital Signature Generation and Verification
A1908	KTS-IFC SP 800-56Brev2 per IG D.G	RSA-OAEP without key confirmation Key sizes: 2048, 3072, and 4096 bits Hybrid Key-Transport scheme incorporating KTS-OAEP and SP 800-38F	Modulo: 2048, 3072, 4096 Key Generation Methods: rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme: KTS-OAEP-basic: Key Transport Method: Hash Algorithms: SHA-1, SHA2-256 Supports Null Associated Data Associated Data Encoding: concatenation KAS Role: initiator, responder Key Length: 1024 SSP establishment methodology provides between 112 and 150 bits of encryption strength	Key Transport, Optional RSA encapsulation schemes for protecting keys that customers import into AWS KMS
A1908	KAS SP 800-56Arev3 per IG D.F Scenario 2, path (2)	KAS-ECC (Cofactor) Ephemeral Unified scheme with key confirmation	P-384 curve providing 192 bits of encryption strength	Key Agreement
A1908	KAS SP 800-56Arev3 per IG D.F Scenario 2, path (2)	KAS-ECC (Cofactor) One-Pass Diffie-Hellman scheme with key confirmation	P-384 curve providing 192 bits of encryption strength	Key Agreement
A1908	KDA SP 800-56Crev1	[SP 800-56Crev1] One-step key derivation	Auxiliary Function Methods: Auxiliary Function Name: SHA2-256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-384 MAC Salting Methods: default, random	Key Derivation

CAVP Cert[1]	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use / Function
Vendor Affirmed IG D.H	CKG SP 800-133rev2	[SP 800-133rev2, Section 4] Seeding for asymmetric key generation uses un-modified DRBG output [SP 800-133rev2, Section 6.1] Symmetric key genera-tion uses unmodified DRBG output [SP 800-133rev2, Section 6.2] Symmetric keys can be derived	N/A	Key Generation
AWS Key Management Service Key Derivation Function Library				
A1910	KBKDF SP 800-108	Counter Mode HMAC-based KDF with SHA2-256	Capabilities: KDF Mode: Counter MAC Mode: HMAC-SHA2-256 Supported Lengths: 8-4096 Increment 8 Fixed Data Or-der: Before Fixed Data Counter Length: 32 Supports Empty IV Custom Key In Length: 0	Key Derivation
Entropy Source				
N/A	ENT (P) SP 800-90B	Entropy source	384 bits	Provides seeding mate-rial for the DRBG
A1791	Conditioning Components	AES-ECB AES-CBC-MAC Counter DRBG	Key Length: 128 Payload Length: 128	Provides seeding mate-rial for the DRBG

Table 3 –Approved Algorithms

Algorithm	Caveat	Use / Function
ECDSA secp256k1	key agreement; key establishment methodol-ogy provides 128 bits of encryption strength	[IG C.A] Curves: secp256k1 may only be used in block-chain related applications

Table 4 – Non-Approved Algorithms Allowed in the Approved Mode of Operation

Algorithm	Caveat	Use / Function
HMAC-SHA1 (non-com-pliant)	No security claimed	Used as defined by the IPMI specification on the Baseboard Management Controller (BMC) which operates completely independently from the rest of the module's functionality
HMAC-SHA1-96 (non-compliant)	No security claimed	Used as defined by the IPMI specification on the Baseboard Management Controller (BMC) which operates completely independently from the rest of the module's functionality

Algorithm	Caveat	Use / Function
HMAC-MD5	No security claimed	Used as defined by the IPMI specification on the Baseboard Management Controller (BMC) which operates completely independently from the rest of the module's functionality
HMAC-SHA2-256-128 (non-compliant)	No security claimed	Used as defined by the IPMI specification on the Baseboard Management Controller (BMC) which operates completely independently from the rest of the module's functionality
AES-CBC-128 (non-compliant)	No security claimed	Used as defined by the IPMI specification on the Baseboard Management Controller (BMC) which operates completely independently from the rest of the module's functionality

Table 5 - Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed

The cryptographic boundary consists of the entire module as shown in Figures 1 and 2.

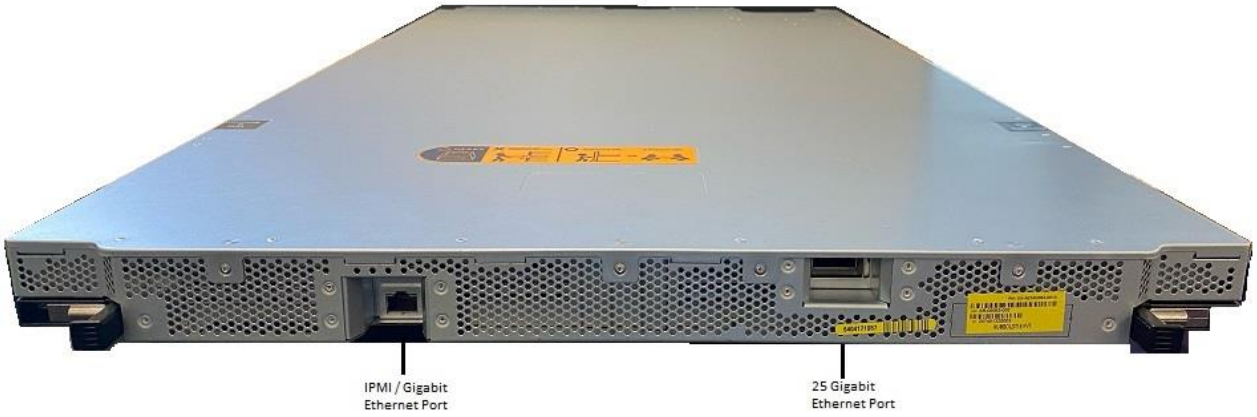

Figure 1 – Cryptographic Module Boundary (Front)

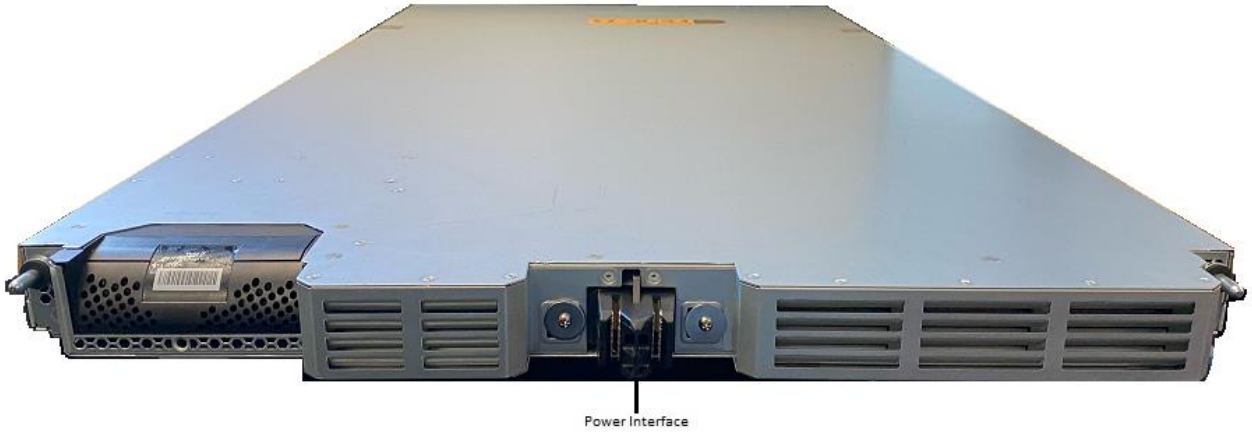

Figure 2 - Cryptographic Module Boundary (Back)

3. Cryptographic Module Interfaces

The module provides a number of physical and logical interfaces to the device, and the physical interfaces provided by the module are mapped to four FIPS 140-3 defined logical interfaces: data input, data output, control input, and status output. The control output interface is not applicable. The logical interfaces and their mapping are provided in the following table:

Physical port	Logical inter-face	Data that passes over port/interface
25 Gigabit Ethernet Port	Data Input	Main session interface for cryptographic services
25 Gigabit Ethernet Port	Data Output	Main session interface for cryptographic services
25 Gigabit Ethernet Port	Control Input	Main session interface for cryptographic services
IPMI / Gigabit Ethernet Port	Control Input	Provides serial console access, query power on / off
25 Gigabit Ethernet Port	Status Output	Main session interface for cryptographic services
IPMI / Gigabit Ethernet Port	Status Output	Provides serial console access, query power on / off
Power	Power	N/A

Table 6 – Ports and Interfaces

4. Roles, Services, and Authentication

Operators of the module may assume the following three roles implicitly:

KMS Front End Role (KMS-FE) - The KMS front end hosts perform actions on behalf of customers of AWS KMS.

KMS Coordinator Role (KMS-C) - Non-public facing KMS hosts perform actions on behalf of KMS administrators in the Administrator Role.

Administrator Role (Admin) - Employees of AWS who are authorized to manage the module.

For FIPS 140-3 purposes, the KMS Coordinator and Administrator roles serve as the Cryptographic Officer role per FIPS 140-3 requirements. The KMS-Front End role serves as the User role per FIPS 140-3 requirements.

The module supports only identity-based authentication and requires RSA or ECDSA signatures using RSA with 2048-bit, 3072-bit, or 4096-bit keys, or ECDSA with P-384. Operators of the module are identified by unique Operator Signature Public Key (QOS). The list of operator keys and the role of each operator are configured using either the Initialize or InitializeAndCreateDomain service. Operators interact with the module by submitting digitally signed commands to the module. The module authenticates operators by verifying the digitally signed commands submitted to the module.

Role	Authentication Method	Authentication Strength
KMS Front End Role (KMS-FE)	Identity based authentication. Commands are signed using the operator's RSA 2048, 3072, 4096 or ECDSA P384 key	112 to 192 bits of security
KMS Coordinator Role (KMS-C)	Identity based authentication. Commands are signed using the operator's RSA 2048, 3072, 4096 or ECDSA P384 key	112 to 192 bits of security
Administrator Role (Admin)	Identity based authentication. Commands are signed using the operator's RSA 2048, 3072, 4096 or ECDSA P384 key	112 to 192 bits of security

Table 7 – Roles and Authentication

The list of services supported by the module are listed in Table 8. Unless otherwise specified, access to services can be configured to require one or more members of one or more roles listed in Table 7. These services are used only by components of KMS to fulfill requests under specific public AWS KMS APIs and cannot be used directly by KMS customers. See http://docs.aws.amazon.com/kms/latest/APIReference/Welcome.html for a list of the current public AWS KMS APIs.

Strength of Authentication

Authentication to the module requires RSA (2048 or 4096-bit) or ECDSA (P-384) signature verification. These authentication methods are cryptographically strong and provide between 112 to 192 bits of security. The possibility of a single random authentication attempt succeeding is 2^{-112} which is far less than the required minimum of less than 1/1,000,000.

Assuming an upper bound of 2^{32} authentication requests per second, the possibility of a random authentication succeeding within a one-minute period is $(60*2^{32})/2^{112} = 15/2^{78}$ which is significantly less than 1/100,000. The cryptographic strengths of the digital signatures used for authentication create such difficulty in achieving a successful random authentication attempt that even the theoretical maximum bandwidth of the 25 Gb/second Ethernet port is not significant enough to allow enough attempts in a one-minute period.

Services

Role	Service	Input	Output
Cryptographic Services			
KMS-FE, KMS-C, Admin	**Create**	None	A HSM Backing Key encrypted with the active Domain Key (DKn), or An Import Wrapping Key Pair (dIWK, QIWK) The IWK private key is encrypted with the active Domain Key (DKn) The IWK public key
KMS-FE, KMS-C, Admin	**ImportKey**	The private key of an Import Wrapping Key Pair (IWK) encrypted with the active or a recent iteration of domain key (DKn or DKn-1) Customer Supplied Key (CSK), encrypted with the public key of the Import Wrapping Key. This may use the wrapping methods as defined in section 9.2 or 9.3 of SP 800-56B, using the ephemeral Import Wrapping Envelope Key (IWEK)	The Customer Supplied Key, encrypted with the current active domain key (DKn)
KMS-FE, KMS-C, Admin	**RefreshKey**	HBK or CSK encrypted with a recent iteration of a Domain Key (DKn-1)	HBK or CSK encrypted with the active domain key (DKn)
KMS-FE, KMS-C, Admin	**Encrypt**	A HBK or CSK encrypted with the active or a recent iteration of domain key (DKn or DKn-1)	N/A (encrypted ciphertext)
KMS-FE, KMS-C, Admin	**Decrypt**	A HBK or CSK encrypted with a Domain Key (DKn) Ciphertext or encrypted Customer Data Key (CDK) Customer Data Encryption Public Key (QCDEK)	Arbitrary data or CDK encrypted using the HOSK

Role	Service	Input	Output
KMS-FE, KMS-C, Admin	**ReEncrypt**	A HBK or CSK encrypted with the active or a recent iteration of domain key (DKn or DKn-1) used to decrypt the provided ciphertext A HBK or CSK encrypted with the active or a recent iteration of domain key (DKn or DKn-1) used to encrypt the resulting plaintext Ciphertext or encrypted Customer Data Key (CDK)	N/A (encrypted ciphertext)
KMS-FE, KMS-C, Admin	**Sign**	HBK or CSK encrypted with the active domain key (DKn)	None (signature)
KMS-FE, KMS-C, Admin	**Verify**	HBK or CSK encrypted with the active domain key (DKn) (signature to be verified)	None
KMS-FE, KMS-C, Admin	**EncryptRandomBytes**	HBK or CSK encrypted by the active domain key (DKn)	A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HBK or CSK
KMS-FE, KMS-C, Admin	**GenerateAndEncryptRandomBytes**	HBK or CSK encrypted by the active domain key (DKn) Customer Data Encryption Public Key (QCDEK)	A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HOSK A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HBK or CSK
KMS-FE, KMS-C, Admin	**GenerateDataKeyPair**	HBK or CSK encrypted by the active domain key (DKn) Customer Data Encryption Public Key (QCDEK)	An asymmetric Customer Data Key (CDK) private key encrypted by the HOSK An asymmetric Customer Data Key (CDK) private key encrypted by the HBK or CSK
KMS-FE, KMS-C, Admin	**GenerateDataKeyPairWithoutPlaintext**	HBK or CSK encrypted by the active domain key (DKn)	An asymmetric Customer Data Key (CDK) private key encrypted by the HBK or CSK
KMS-FE, KMS-C, Admin	**Generate**	Customer Data Encryption Public Key (QCDEK)	None

Role	Service	Input	Output
KMS-FE, KMS-C, Admin	**GetParametersForReplication**	None	Public Replication Agreement Key ($QRAK_1$) Private Replication Agreement Key ($dRAK_1$) encrypted by the active domain key (DKn)
KMS-FE, KMS-C, Admin	**WrapKeyForReplication**	Public Replication Agreement Key ($QRAK_1$) HBK encrypted by the active domain key (DKn) Replication Agreement Key Pair ($dRAK_2$, $QRAK_2$)	Public Replication Agreement Key ($QRAK_2$) Customer Replicated Key (CRK) encrypted by the Replication Wrapping Key (RWK)
KMS-FE, KMS-C, Admin	**ImportReplicatedKey**	Private Replication Agreement Key ($dRAK_1$) encrypted by the active domain key (DKn) Public Replication Agreement Key ($QRAK_2$) Customer Supplied Key (CSK) encrypted by the Replication Wrapping Key (RWK)	HBK encrypted by the active domain key (DKn) Customer Replication Key (CRK)
Configuration Services			
KMS-FE, KMS-C, Admin	**CreateDomain**	List of Operator Signature Public Keys (QOS)	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Initial Domain Key (DK0) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSK0) • Public Replication Signing Key (QRSK0)

Role	Service	Input	Output
KMS-FE, KMS-C, Admin	**IngestDomain**	A Domain Token containing the following CSPs: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)	The unmodified input Domain Token
KMS-FE, KMS-C, Admin	**ForgetDomain**	A Domain Token containing the following CSPs: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)	The unmodified input Domain Token
KMS-FE, KMS-C, Admin	**GetDomain**	None	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)

Role	Service	Input	Output
KMS-FE, KMS-C, Admin	**ChangeDomain**	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) • Public Replication Signing Key (QRSKn) HSM Signature Public Keys (QHSK) and HSM Key Agreement Public Keys (QHAK) of the domain members to be added (optional) List of Operator Signature Public Keys (QOS) (optional) List of Public Replication Signing Keys (QRSKm, …, QRSKn) (optional)	An updated Domain Token containing the following CSPs: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)
KMS-FE, KMS-C, Admin	**Initialize**	One or more Domain Tokens. Each Domain Token contains: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)	None

Role	Service	Input	Output
All (un-authenticated)	**InitializeAndCreateDomain**	List of Operator Signature Public Keys (QOS)	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Initial Domain Key (DK0) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn)
KMS-FE, KMS-C, Admin	**Attest**	HSM Signature Key Pair (dHSK, QHSK) Host Agreement Public Key (QHAK) Operator Signature Public Key(s) (QOS) HSM Session Key Encryption Key (HSKEK) HSM-to-Operator Session Key (HOSK)	HSM Signature Public Key (QHSK) HSM Agreement Public Key (QHAK)
KMS-FE, KMS-C, Admin	**GetAttestationChallenge**	None	None
KMS-FE, KMS-C, Admin	**GetAttestationIdentity**	None	None
All (un-authenticated)	**Wipe**	None	None
All (un-authenticated)	**GetInitialDomainName**	None	None
All (un-authenticated)	**DeactivateAndReboot**	None	None

Role	Service	Input	Output
One member from any role	**NegotiateSessionKey**	Operator Ephemeral Agreement Public Key (QOEAK)	Encrypted HSM-Operator Session Key (HOSK) encrypted with the Domain Key (DKn) or HSM Session Key Encryption Key (HSKEK) HSM-Operator Session Key (HOSK) encrypted with a 256-bit key derived from the shared secret established using elliptic curve Diffie Hellman key exchange (NIST-P384) using the HSM Ephemeral Agreement Public Key (QE) and the Operator Ephemeral Agreement Public Key (QOEAK) HSM Ephemeral Agreement Public Key (QE)
KMS-FE, KMS-C, Admin	**UpdateHostConfiguration**	None	None
Audit Log Services			
KMS-FE, KMS-C, Admin	**ListLogs**	None	None
KMS-FE, KMS-C, Admin	**GetLog**	None	None
KMS-FE, KMS-C, Admin	**DeleteLog**	None	None
Other Services			
All (unauthenticated)	**Ping**	None	Returns “healthy” if the module is operating in Approved mode. Returns “failure” if the module is not operating in Approved mode.
All (unauthenticated)	**Approved**	None	Returns “healthy” if the module is operating in Approved mode. Returns “failure” if the module is not operating in Approved mode.
All (unauthenticated)	**Version**	None	Module name, hardware version and firmware version

Role	Service	Input	Output
All (un-authenticated)	**Hardware monitoring**	None	Hardware sensor data
All (un-authenticated)	**Power management**	None	None
All (un-authenticated)	**Serial over LAN (SOL)**	None	None

Table 8 – Roles, Service Commands, Input and Output

Each approved service provides an indicator when the service utilizes an approved cryptographic algorithm, security function, or process in an approved manner. Per IG 2.4.C, the module implements a global indicator via the "Approved" service which is a persistent indicator that only returns healthy if the module is running in its approved mode of operation where approved services are executing.

Approved Services

Approved services supported by the module are listed in Table 9.

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Create	Generates and encrypts either an HSM Backing Key (HBK) or an Import Wrapping Key Pair (dIWK, QIWK) private key	CTR DRBG AES GCM KBKDF RSA (keygen) ECDSA (keygen) CKG	HSM Backing Key IWK public and private keys Active Domain Key (DK_n) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ImportKey	Decrypts a Customer Supplied Key (CSK) and re-encrypts it with the active Domain Key (DK_n)	AES GCM KBKDF KTS-IFC (RSA-OAEP)	The private key of an Import Wrapping Key Pair (dIWK, QIWK) Customer Supplied Key (CSK) Active Domain Key (DK_n) HSM-to-Operator Session Key (HOSK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"
RefreshKey	Re-encrypts an HSM Backing Key (HBK) key or Customer Supplied Key (CSK) encrypted with a recent iteration of the domain key (DK_{n-1}) with the active domain key (DK_n)	AES GCM KBKDF	HBK or CSK encrypted with a recent iteration of a Domain Key (DKn or DKn-1) Active or a recent iteration of Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"
Encrypt	Encrypt an arbitrary set of bytes using the DEK derived from the provided HBK or CSK	AES GCM	A HBK or CSK encrypted with the active or a recent iteration of domain key (DKn or DKn-1) Active or a recent iteration of Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK) Data Encryption Key (DEK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Decrypt	Decrypts ciphertext using the DEK derived from the provided HBK or CSK	AES GCM	A HBK or CSK encrypted with a Domain Key (DK_n) Ciphertext or encrypted Customer Data Key (CDK) Arbitrary data or CDK encrypted using the HOSK Active or a recent iteration of domain key (DK_n or DK_{n-1}) HSM-to-Operator Session Key (HOSK) Data Encryption Key (DEK) Customer Data Encryption Public Key (QCDEK) Customer Data Encryption Symmetric Key (SCDEK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write Generate	“healthy”
ReEncrypt	Decrypts ciphertext using the DEK derived from the provided HBK or CSK, then re-encrypts the resulting plaintext under the DEK from a separately provided HBK or CSK This operation does not expose the plaintext	AES GCM	A HBK or CSK encrypted with the active or a recent iteration of domain key (DK_n or DK_{n-1}) used to decrypt the provided ciphertext A HBK or CSK encrypted with the active or a recent iteration of domain key (DK_n or DK_{n-1}) used to encrypt the resulting plaintext Ciphertext or encrypted Customer Data Key (CDK) Active or a recent iteration of Domain Key (DK_n or DK_{n-1}) HSM-to-Operator Session Key (HOSK) Data Encryption Key (DEK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Sign	Performs an ECDSA or RSA sign operation, or HMAC operation using the provided HBK or CSK	CTR DRBG AES GCM RSA ECDSA SHS HMAC	HBK or CSK encrypted with the active domain key (DKn) Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"
Verify	Performs an ECDSA or RSA verify, or HMAC operation using the provided HBK or CSK	AES GCM RSA ECDSA SHS HMAC	HBK or CSK encrypted with the active domain key (DKn) Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"
EncryptRandomBytes	Generate a number of random bytes and encrypt it using the DEK derived from the specified HBK or CSK The random bytes may be used as cryptographic key material as Customer Data Keys (CDK)	CTR DRBG AES GCM CKG	HBK or CSK encrypted with the active domain key (DKn) A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HBK or CSK Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key Data Encryption Key (DEK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
GenerateAndEncryptRandomBytes	Generate a number of random bytes for use and encrypt it using the DEK derived from the specified HBK or CSK The random bytes may be used as cryptographic key material as Customer Data Keys (CDK) Note that the GenerateAndEncryptRandomBytes API will return encrypted versions of the random bytes in 2 forms	CTR DRBG AES GCM CKG	HBK or CSK encrypted with the active domain key (DKn) A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HBK or CSK A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HOSK Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key Data Encryption Key (DEK) Customer Data Encryption Public Key (QCDEK) Customer Data Encryption Symmetric Key (SCDEK)	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize Write	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
GenerateDataKey-Pair	Generate an asymmetric key pair and encrypt it with the specified HBK or CSK The asymmetric key pair will be used as cryptographic key material as Customer Data Keys (CDK) Note that the GenerateDataKeyPair API will return encrypted versions of the CDK in 2 forms	CTR DRBG RSA (keygen) ECDSA (keygen) AES GCM CKG	HBK or CSK encrypted by the active domain key (DK_n) An asymmetric Customer Data Key (CDK) private key encrypted by the HOSK An asymmetric Customer Data Key (CDK) private key encrypted by the HBK or CSK Active or a recent iteration of domain key (DK_n or DK_{n-1}) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key Customer Data Encryption Public Key (QCDEK) Customer Data Encryption Symmetric Key (SCDEK)	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize Write	“healthy”
GenerateDataKey-PairWithoutPlaintext	Generate an asymmetric key pair and encrypt it with the specified HBK or CSK The asymmetric key pair will be used as cryptographic key material as Customer Data Keys (CDK)	CTR DRBG RSA (keygen) ECDSA (keygen) AES GCM CKG	HBK or CSK encrypted by the active domain key (DK_n) An asymmetric Customer Data Key (CDK) private key encrypted by the HBK or CSK Active or a recent iteration of domain key (DK_n or DK_{n-1}) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize Write	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Generate	Generate a specified number of random bytes, up to 1024 bytes	CTR DRBG AES GCM CKG	HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key Customer Data Encryption Public Key (QCDEK) Customer Data Encryption Symmetric Key (SCDEK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	"healthy"
GetParameters-ForReplication	This API generates a new Replication Agreement Key Pair ($dRAK_1$, $QRAK_1$) The Private Replication Agreement Key ($dRAK_1$) is encrypted with the domain key (DK_n) The API also signs all output with the Private Replication Signing Key ($dRSK_n$ or $dRSK_{n-1}$)	CTR DRBG ECDSA (keygen) AES GCM CKG	Public Replication Agreement Key ($QRAK_1$) Private Replication Agreement Key ($dRAK_1$) encrypted by the active domain key (DK_n) Replication Agreement Key Pair ($dRAK_1$, $QRAK_1$) HSM-to-Operator Session Key (HOSK) Active or a recent iteration of domain key (DK_n or DK_{n-1}) Active or a recent iteration of a Private Replication Signing Key ($dRSK_n$ or $dRSK_{n-1}$) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
WrapKeyForReplication	This API takes an input a public Replication Agreement Key ($QRAK_1$) generated from an HSM, and generates a new Replication Agreement Key pair ($dRAK_2$, $QRAK_2$) $QRAK_1$ and $dRAK_2$ are combined using the Diffie-Hellmann key exchange to produce a shared secret and derive a symmetric secret key (the Replication Wrapping Key, RWK) The RWK is then used to encrypt an HBK, resulting in a Customer Replicated Key (CRK)	KAS (ECCDH) KDA (one-step KDF SHA2) AES GCM ECDSA	Public Replication Agreement Key ($QRAK_1$) HBK encrypted by the active domain key (DK_n) Replication Agreement Key Pair ($dRAK_2$, $QRAK_2$) Public Replication Agreement Key ($QRAK_2$) Customer Replicated Key (CRK) encrypted by the Replication Wrapping Key (RWK) HSM-to-Operator Session Key (HOSK) Active or a recent iteration of domain key (DK_n or DK_{n-1}) Active or a recent iteration of the Private Replication Signing Key ($dRSK_n$ or $dRSK_{n-1}$) Active or a recent iteration of the Public Replication Singing Key ($QRSK_n$ or $QRSK_{n-1}$) Replication Agreement RWK Shared Secret Z (RRZ) Customer Replication Key (CRK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ImportReplicat-edKey	This API combines two Replication Agreement Key ($dRAK_1$ and $QRAK_2$) using the Diffie-Hellmann key ex-change to produce a shared secret and derive a Replication Wrapping Key (RWK) The RWK is used to decrypt the Cus-tomer Replicated Key (CRK), obtain-ing an HBK, which is then re-encrypted using the Domain Key (DK_n) The API also vali-dates input using the Public Replica-tion Signing Key ($QRSK_n$ or $QRSK_{n-1}$)	KAS (ECCDH) KDA (one-step KDF SHA2) AES GCM ECDSA SHS	Private Replication Agreement Key ($dRAK_1$) encrypted by the active domain key (DK_n) Public Replication Agreement Key ($QRAK_2$) Customer Supplied Key (CSK) encrypted by the Replication Wrapping Key (RWK) HBK encrypted by the active domain key (DK_n) HSM-to-Operator Session Key (HOSK) Active or a recent iteration of domain key (DK_n or DK_{n-1}) Active or a recent iteration of the Public Replication Singing Key ($QRSK_n$ or $QRSK_{n-1}$) Replication Agreement RWK Shared Secret Z (RRZ) Customer Replication Key (CRK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
CreateDomain	Creates a new domain token for a new domain, but does not join the HSM to the domain yet	CTR DRBG KAS (ECCDH) KDA (one-step KDF SHA2) AES GCM ECDSA RSA SHS	List of Operator Signature Public Keys (QOS) HSM Signature Key Pair (dHSK, QHSK) HSM Agreement Key Pair (dHAK, QHAK) HSM Agreement DKEK Shared Secret Z (HDKZ) HSM Agreement DKEK Wrapping Key (HDWK) Initial Domain Key (DK_0) Replication Signing Key ($dRSK_0$, $QRSK_0$) A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Initial Domain Key (DK_0) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_0$) • Public Replication Signing Key ($QRSK_0$) • DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize	"healthy"

IngestDomain	Joins a domain or receive an updated domain token	CTR DRBG KAS (ECCDH) (one-step KDF SHA2) AES GCM ECDSA RSA SHA2	A Domain Token containing the following CSPs: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DKn) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key (dRSKn) • Public Replication Signing Key (QRSKn) HSM Signature Public Key (QHSK) of a known member of the domain HSM Agreement Private Key (dHAK) HSM Agreement DKEK Shared Secret Z (HDKZ) HSM Agreement DKEK Wrapping Key (HDWK) Operator Signature Public Keys (QOS) Domain Key (DKn) Operator Signature Public Keys (QOS) HSM Signature Public Keys (QHSK) of all members of the domain HSM Key Agreement Public Keys (QHAK) of all members of the domain Encrypted Private Replication Signing Key (dRSKn) Public Replication Signing Key (QRSKn) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Read Execute Zeroize Write	“healthy”
ForgetDomain	Deletes domain in-formation as it pertains to a	ECDSA RSA	A Domain Token containing the following CSPs:	KMS-FE, KMS-C, Admin	Read Execute	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
	particular domain on the module including all Domain Keys (DK_n, DK_{n-1}), effectively leaving the domain	SHA2	• List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DK_n) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) Domain Key (DK_n) Operator Signature Public Keys (QOS) HSM Signature Public Keys (QHSK) of all members of the domain HSM Key Agreement Public Keys (QHAK) of all members of the domain		Zeroize	
GetDomain	Retrieves the current version of the domain token for a specified domain	ECDSA RSA SHA2	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DK_n) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) Domain Key (DK_n) Operator Signature Public Keys (QOS)	KMS-FE, KMS-C, Admin	Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ChangeDomain	Modifies the current state of an operational domain	CTR DRBG KAS (ECCDH) (one-step KDF SHA2) AES GCM ECDSA RSA SHA2	A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DK_n) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) HSM Signature Public Keys (QHSK) and HSM Key Agreement Public Keys (QHAK) of the domain members to be added (optional) List of Operator Signature Public Keys (QOS) (optional) List of Public Replication Signing Keys ($QRSK_m$, …, $QRSK_n$) (optional) Domain Key Encrypting Key (DKEK) Domain Key (DK_n) HSM Ephemeral Agreement Key Pair (dE, QE) HSM Agreement Key (HAK) HSM Signature Key (HSK) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Generate Read Execute Zeroize Write	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Initialize	Initializes the HSM by generating the HSM Signature Key and HSM Agree-ment Key and configuring the HSM's operator and access control using a domain token from another HSM The Initialize API is only used during the module setup and initialization process If the HSM is already initialized by a call to either the Initialize or Ini-tializeAndCreateDo main API, the Ini-tialize API will return an error as the HSM cannot be Initialized again without a reboot	CTR DRBG ECDSA (keygen, sign) KAS (EC CDH) (one-step KDF SHA2) AES GCM CKG	One or more Domain Tokens. Each Domain Token contains: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Domain Keys (DK_n) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) HSM Signature Key (HSK) HSM Agreement Key (HAK) HSM Agreement HSKEK Shared Secret Z (HHKZ) HSM Session Key Encryption Key (HSKEK) Operator Signature Public Keys (QOS) DRBG (CTR AES) V and AES key DRBG (CTR AES) Seed Entropy Input String	All / unauthenti-cated	Generate Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
InitializeAndCreateDomain	Initializes the HSM by generating the HSM Signature Key and HSM Agreement Key, configuring the list of operators, roles and the quorum-based access control ruleset for all services / APIs The InitializeAndCreateDomain API is only used during the module setup and initialization process If the HSM is already initialized by a call to either the Initialize or InitializeAndCreateDomain API, the InitializeAndCreateDomain API will return an error as the HSM cannot be Initialized again without a reboot	CTR DRBG ECDSA (keygen, sign) KAS (EC-CDH) (one-step KDF SHA2) AES GCM CKG	List of Operator Signature Public Keys (QOS) HSM Signature Key Pair (dHSK, QHSK) HSM Agreement Key Pair (dHAK, QHAK) HSM Agreement DKEK Shared Secret Z (HDKZ) HSM Agreement DKEK Wrapping Key (HDWK) HSM Agreement HSKEK Shared Secret Z (HHKZ) HSM Session Key Encryption Key (HSKEK) Initial Domain Key (DK_0) A Domain Token containing: • List of Operator Signature Public Keys (QOS) • List of HSM Signature Public Keys (QHSK) of all members of the domain • List of HSM Key Agreement Public Keys (QHAK) of all members of the domain • Encrypted Initial Domain Key (DK_0) • Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) DRBG (CTR AES) V and AES key	All / unauthenticated	Generate Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Attest	The Attest API is used by operators to attest an initialized HSM to ensure that the system is running the correct software, and to obtain an authentic copy of its credentials prior to being added to a domain	CTR DRBG ECDSA (verify) SHA2 AES GCM	HSM Signature Public Key (QHSK) HSM Agreement Public Key (QHAK) HSM Signature Key Pair (dHSK, QHSK) Operator Signature Public Key(s) (QOS) HSM Agreement HSKEK Shared Secret Z (HHKZ) HSM Session Key Encryption Key (HSKEK) HSM-to-Operator Session Key (HOSK) DRBG (CTR AES) V and AES key	KMS-FE, KMS-C, Admin	Read Execute Zeroize	"healthy"
GetAttestationChallenge	The GetAttestationChallenge API is used by operators to retrieve a token that can be used to validate the identity of another HSM	AES GCM	Active or a recent iteration of Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize	"healthy"
GetAttestationIdentity	The GetAttestationIdentity API is used by operators to retrieve information to attest the identity of the HSM	AES GCM	Active or a recent iteration of Domain Key (DKn or DKn-1) HSM-to-Operator Session Key (HOSK)	KMS-FE, KMS-C, Admin	Read Execute Zeroize	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Wipe	The Wipe API will delete the HSM Signature Key and HSM Agreement Key from volatile memory The Wipe API will fail unless all previously created domains in the module have been deleted using the ForgetDomain API	N/A	HSM Signature Key Pair (dHSK, QHSK) HSM Agreement Key Pair (dHAK, QHAK) HSM Session Key Encryption Key (HSKEK)	All / unauthenticated	Zeroize	"healthy"
GetInitialDomainName	Retrieves the initial domain name from an initialized HSM that is used as part of the domain creation bootstrap process	N/A	N/A	All / unauthenticated	N/A	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
DeactivateAndRe-boot	The Deactivate-AndReboot API returns the HSM to the factory state and reboots after verifying the HSM Signature Key and HSM Agreement Key have been de-leted by the Wipe API (The module will perform self-tests after during reboot process)	N/A	N/A	All / unauthenti-cated	N/A	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
NegotiateSes-sionKey	Uses a set of iden-tity keys to securely negotiate a session key that can be used between a KMS host and any HSM in the domain The NegotiateSes-sionKey API will return encrypted versions of the HSM-Operator Ses-sion Key (HOSK) in 2 forms	CTR DRBG RSA (verify) ECDSA (verify) SHA2 KAS (ECCDH) (one-step KDF SHA2) AES GCM	Operator Ephemeral Agreement Public Key (QOEAK) HSM Ephemeral Agreement Key Pair (dE, QE) HSM-Operator Session Key (HOSK) Encrypted HSM-Operator Session Key (HOSK) encrypted with the Domain Key (DKn) or HSM Session Key Encryption Key (HSKEK) HSM-Operator Session Key (HOSK) en-crypted with a 256 bit key derived from the shared secret established using elliptic curve Diffie Hellman key exchange (NIST-P384) using the HSM Ephemeral Agree-ment Key (QE) and the Operator Ephemeral Agreement Public Key (QOEAK) HSM Ephemeral Agreement Public Key (QE) Operator Signature Public Key (QOS) HSM Signature Key (dHSK) DRBG (CTR AES) V and AES key	One member from any role	Generate Read Execute Zeroize	“healthy”
UpdateHostConfig-uration	Allows updates of non-security-rele-vant host configuration	RSA (verify) ECDSA (verify) SHA2	Operator Signature Public Key (QOS)	KMS-FE, KMS-C, Admin	Execute	“healthy”

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ListLogs	Returns a list of audit log file names	RSA (verify) ECDSA (verify) SHA2	Operator Signature Public Key (QOS)	KMS-FE, KMS-C, Admin	Execute	"healthy"
GetLog	Retrieves specified audit log files	RSA (verify) ECDSA (verify) SHA2	Operator Signature Public Key (QOS)	KMS-FE, KMS-C, Admin	Execute	"healthy"
DeleteLog	Deletes specified audit log file	RSA (verify) ECDSA (verify) SHA2	Operator Signature Public Key (QOS)	KMS-FE, KMS-C, Admin	Execute	"healthy"
Ping	Returns "healthy" if the module is initialized and has ingested a domain Returns "failure" otherwise	N/A	N/A	All / unauthenticated	None	"healthy"
Approved	Approved mode indicator that apply to approved services on the 25G Ethernet port Returns "healthy" if the module is operating in Approved mode Returns "failure" if the module is not operating in Approved mode	N/A	N/A	All / unauthenticated	None	"healthy"

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Version	Returns the module name, hardware version and firmware version	N/A	N/A	All / unauthenticated	None	N/A
Hardware monitoring	Provide access via IPMI to hardware sensor data to monitor temperatures, fan speed, etc	None	N/A	All / unauthenticated	None	Successful completion of service
Power management	Turns on and off the module via IPMI	None	N/A	All / unauthenticated	None	Successful completion of service
Serial over LAN (SOL)	Provides access to the module's console before the module enters Approved mode via IPMI In Approved mode, the SOL link is active but the module firmware blocks all input commands and status output to the console	None	N/A	All / unauthenticated	None	Successful completion of service

Table 9 – Approved Services

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.

5. Software/Firmware Security

The module performs integrity check on all firmware components using a 256-bit error detection code (EDC) on all module components. The integrity check is performed upon the initialization of the module and does not require operator intervention to run. If the check fails, the module will enter into an error state. The module does not support firmware loading.

The operator can run the integrity test on demand by rebooting the module using the DeactivateAndReboot API.

6. Operational Environment

The module has a non-modifiable operational environment and does not allow loading of any additional firmware while the module is operating in Approved mode.

7. Physical Security

The module is a hardware module with a multiple-chip standalone embodiment and conforms to the Level 3 requirements for physical security. The module's production-grade enclosure is made of hard metal, and the enclosure does not provide a removable cover. The baffles installed by AWS satisfy FIPS 140-3 requirements for module opacity and probing.

Physical Security Mechanism	Recommended Frequency of Inspection/Test	Inspection/Test Guidance Details
Tamper-evident physical enclosure with no removable cover	Inspect when the module unexpectedly reboots or becomes unresponsive	Inspect the physical enclosure for evidence of tampering, such as dents, signs of drilling or prying, cracks in the hard plastic portion of the enclosure

Table 10 – Physical Security Inspection Guidelines

The module supports environments failure protection and shuts down if the temperature or voltage is outside of the values described in Table 11.

	Temperature or voltage measurement	Specify EFP or EFT	Specify if this condition results in a shutdown or zeroisation
Low Temperature	- 8 °C	EFP	Shutdown
High Temperature	54 °C	EFP	Shutdown
Low Voltage	10 V	EFP	Shutdown
High Voltage	14 V	EFP	Shutdown

Table 11 – EFP/EFT

	Hardness tested temperature measurement
Low Temperature	- 8 °C
High Temperature	52 °C

Table 12 – Hardness Testing Temperature Ranges

8. Non-invasive Security

This section is not applicable. The module does not implement non-invasive attack mitigation techniques.

9. Sensitive Security Parameters Management

Table 13 provides a complete list of Critical Security Parameters used within the module. All keys and SSPs are zeroized by powering off the module.

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
HSM Backing Key (HBK) CSP/PSP	256 bits (AES) 160-256 bits (HMAC) 112 – 128 bits (RSA 2048, 3072 or 4096 bits) 128 – 256 bits (ECDSA P-256, P-384, P-521, or secp256k1)	AES GCM RSA ECDSA HMAC (A1908) CKG	Internally using DRBG or imported from another member of a Domain	**Input**: Encrypted with the Domain Key using AES GCM (electronically) **Output**: Encrypted with the Domain Key using AES GCM (electronically)	N/A	Volatile memory	Overwrite with all zeros	Used as input to a SP 800-108 KBKDF to derive the DEK

Customer Data Key (CDK) CSP/PSP	For symmetric keys, random bits length specified by customer (in the range of 8 bits to 65536 bits) 112 – 128 bits (RSA 2048, 3072 or 4096 bits) 128 – 256 bits (ECDSA P-256, P-384, P-521, or secp256k1)	AES RSA ECDSA (A1908) CKG	Internally using DRBG or imported from another member of a Domain	**Input**: Encrypted using AES GCM with the DEK derived from an HBK or CSK (electronically) **Output**: Encrypted in 2 forms by the GenerateAndEncryptRandomBytes and GenerateDataKeyPair APIs: 1. Encrypted with the DEK derived from an HBK or CSK; and 2. Encrypted with the HOSK to provide secure transport to the requesting service operator/role EncryptRandomBytes and GenerateDataKeyPairWithoutPlaintext APIs export the CDK encrypted with the DEK from an HBK or CSK (electronically)	N/A	Volatile memory	Overwriting with all zeros	Used outside of the module

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Data Encryption Key (DEK) CSP	256 bits (AES)	AES GCM (A1908)	Derived internally using SP 800-108 KBKDF	**Input**: N/A **Output**: N/A	N/A	Volatile memory	Overwriting with all zeros	The DEK is derived from either the HBK or CSK and is used to encrypt the CDK
HSM Agreement Key Pair (dHAK, QHAK) CSP/PSP	192 bits (ECDH P384)	KAS (A1908) CKG	Internally using DRBG	**Input**: N/A **Output**: The public key (QHAK) is exported in plaintext (electronically)	N/A	Volatile memory	Overwriting with all zeros	The dHAK/QHAK are used in key agreement operations to encrypt the DKEK
HSM Ephemeral Agreement Key Pair (dE, QE) CSP/PSP	192 bits (ECDH P384)	KAS (A1908)	Internally using DRBG	**Input**: N/A **Output**: The public key (QE) is exported in plaintext (electronically)	N/A	Volatile memory	Overwriting with all zeros	The dE/QE is used in key agreement operations to encrypt the DKEK
HSM Agreement DKEK Shared Secret Z (HDKZ) CSP	192 bits (ECDH P384)	KAS (A1908)	N/A	N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The HDKZ is the shared secret value Z computed using the HSM Agreement Key (dHAK) and the HSM Ephemeral Agreement Key (QE) The HDKZ is used to derive the HDWK

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
HSM Agreement DKEK Wrapping Key (HDWK) CSP	256 bits (One-Step KDF SHA2-256)	KDA (A1908)	N/A	N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The HDWK is derived from the HDKZ and is used to wrap the DKEK

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Domain Key Encryption Key (DKEK) CSP	256 bits (AES)	AES GCM (A1908)	Internally using DRBG or imported from another member of a Domain	**Input**: The DKEK is encrypted with the HDWK derived using the shared secret (HDKZ) generated from the HSM's Key Agreement Key (QHAK) and another HSM's Ephemeral Key Agreement Key (dE) (electronically) **Output**: The DKEK is encrypted with the HDWK derived using the shared secret (HDKZ) generated from the HSM's Key Agreement Key (dHAK) and another HSM's Ephemeral Key Agreement Key (QE) (electronically)	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation KTS (SP 800-38F)	Volatile memory	Overwriting with all zeros	The DKEK is used to encrypt the DKn when imported to other members of a Domain

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Domain Key (DKn) CSP	256 bits (AES)	AES GCM (A1908) KBKDF (A1910)	Internally using DRBG or imported from another member of a Domain	**Input**: DKn encrypted with the DKEK and may be imported from other members of a Domain (electronically) **Output**: DKn encrypted with the DKEK and may be exported to other members of a Domain (electronically)	N/A	Volatile memory	Overwriting with all zeros	Keys derived from the DKn are used to encrypt HBKs and CSKs
HSM Agreement HSKEK Shared Secret Z (HHKZ) CSP	192 bits (ECDH P384)	KAS (A1908)	N/A	N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The HHKZ is the shared secret value Z computed using the HSM Agreement Key (dHAK) and the Operator Ephemeral Agreement Public Key (QOEAK) The HHKZ is used to derive the HSKEK

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
HSM Session Key Encryption Key (HSKEK) CSP	256 bits (AES)	AES GCM (A1908)	Internally using DRBG	**Input**: N/A **Output**: N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The HSKEK encrypts the HSM-Operator Session Key (HOSK) for the following operations: Initialize, InitializeAndCreateDomain, Attest, GetAttestationIdentity, and Wipe
HSM Signature Key Pair (dHSK, QHSK) CSP/PSP	192 bits (ECDSA P384)	ECDSA (A1908) CKG	Internally using DRBG	**Input**: N/A **Output**: The public key (QHSK) is exported in plaintext (electronically)	N/A	Volatile memory	Overwriting with all zeros	The dHSK is used to sign data created on the HSM

HSM-Operator Session Key (HOSK) CSP	256 bits (AES)	AES GCM (A1908)	Internally using DRBG, or imported from an HSM that is a member of the same domain	**Input**: The HOSK is input encrypted with the domain key (DKn) (electronically) **Output**: The HOSK is encrypted in two forms to be output The first form is encrypted with either the Domain Key (DKn) or the HSM Session Key Encryption Key (HSKEK) using AES GCM (electronically) The second form is encrypted using AES GCM with a 256-bit key derived from the shared secret established using elliptic curve Diffie-Hellman key exchange (NIST-P384) using the HSM Ephemeral Agreement Key Pair (dE,QE) and the Operator Ephemeral Agreement Public Key (dOEAK,	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation KTS (SP 800-38F)	Volatile memory	Overwriting with all zeros	The HOSK is used to encrypt communications between a user and HSMs in the same Domain

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
				QOEAK) (electronically)				
Import Wrapping Key Pair (dIWK, QIWK) CSP/PSP	112 – 128 bits (RSA 2048, 3072 or 4096 bits)	KTS (RSA-OAEP) (A1908)	Internally using DRBG or imported from another member of a Domain	**Input**: The private key (dIWK) is encrypted with the Domain Key (DKn) using AES-GCM for input (electronically) **Output**: the private key (dIWK) is encrypted with the Domain Key (DKn) using AES-GCM. The public key (QIWK) is exported in plaintext (electronically)	N/A	Volatile memory	Overwriting with all zeros	The public key is used by customers of KMS to wrap their CSK for import via the public AWS KMS API

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Import Wrapping Envelope Key (IWEK) CSP	256 bits (AES)	AES KWP (A1908)	Externally by AWS KMS customers	**Input**: IWEK is encrypted using the Import Wrapping Key (QIWK) when used with the ImportKey API when the customer imports a CSK into the AWS KMS system (electronically) **Output**: N/A	KTS-RSA	Volatile memory	Overwriting with all zeros	This key is generated by a customer external to the AWS KMS system and is used to encrypt CSKs for the ImportKey API when AES-KWP is used per SP 800-56B

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Customer Supplied Key (CSK) CSP/PSP	256 bits (AES) 160-256 bits (HMAC) 112 – 128 bits (RSA 2048, 3072 or 4096 bits) 128 – 256 bits (ECDSA P-256, P-384, P-521, or secp256k1)	AES GCM HMAC RSA ECDSA (A1908)	Externally by AWS KMS customers	**Input**: CSK is encrypted using Import Wrapping Key (QIWK) (and, optionally, the ephemeral Import Wrapping Envelope Key (IWEK)) when used with the ImportKey API when the customer imports the key into the AWS KMS system After import, the CSK is encrypted with the Domain Key using AES GCM (electronically) **Output**: CSK encrypted by a Domain Key (DKn) (electronically)	KTS-OAEP without key confirmation KTS-RSA Hybrid Key-Transport scheme incorporating KTS-OAEP and SP 800-38F	Volatile memory	Overwriting with all zeros	This key is generated by a customer of KMS outside the AWS KMS system to sign or encrypt plaintext It can also be used to encrypt CDKs
Entropy Input String CSP	384 bits	Random Number Generation ENT (P)	Internal entropy source	**Input:** N/A **Output:** N/A	N/A	Volatile memory	Overwriting with all zeros	Random Number Generation

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
DRBG (CTR AES) V and AES key CSP	SP 800-90A CTR DRBG V (128 bits) AES key (256 bits)	DRBG AES CTR AES-ECB (A1908)	Internal entropy source	**Input**:N/A **Output**: N/A	N/A	Volatile memory	Overwriting with all zeros	Entropy input (length dependent on security strength)
DRBG (CTR AES) Seed CSP	256 bits	DRBG AES CTR AES-ECB (A1908)	Internal entropy source	**Input**: N/A **Output**: N/A	N/A	Volatile memory	Overwriting with all zeros	Seeding material for the DRBG. Used to derive the DRBG (AES CTR) V and AES key
Replication Signing Key Pair (dRSKn, QRSKn) CSP/PSP	192 bits (ECDSA P384)	ECDSA (A1908)	Internally using DRBG or imported from another member of a Domain	**Input**: $dRSK_n$ encrypted with the DKEK may be imported from other members of a Domain; $QRSK_n$ may be imported by an operator (electronically) **Output**: $dRKS_n$ encrypted with the DKEK may be exported to other members of a Domain; $QRSK_n$ may be exported in plaintext (electronically)	N/A	Volatile memory	Overwriting with all zeros	The private key ($dRSK_n$) is used to sign the outputs of GetParametersForReplication and WrapKeyForReplication APIs The public key ($QRSK_n$) is used to verify the input of WrapKeyForReplication and ImportReplicatedKey APIs

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Replication Agreement Key ($dRAK_k$, $QRAK_k$) CSP/PSP	192 bits (ECDH P384)	ECDH (A1908) CKG	Internally using DRBG or imported from a member of a different Domain	**Input**: $QRAK_k$ may be imported in plaintext from another HSM; $dRAK_k$ may be imported encrypted with the domain key (DK_n) from another HSM (electronically) **Output**: $QRAK_k$ may be exported in plaintext; $dRAK_k$ may be exported encrypted with the domain key (DK_n) (electronically)	N/A	Volatile memory	Overwriting with all zeros	Keys used for key agreement to derive a Replication Wrapping Key (RWK)

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Replication Agreement RWK Shared Secret Z (RRZ) CSP	192 bits (ECDH P384)	KAS (A1908)	N/A	N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The RRZ is the shared secret value Z computed using the private portion of a region's Replication Agreement Key ($dRAK_k$) and the public portion of another region's Replication Agreement Key ($QRAK_k$) The RRZ is used to derive the RWK
Replication Wrapping Key (RWK) CSP	256 bits (AES)	AES GCM (A1908)	Internally derived from a Public Replication Agreement Key ($QRAK_1$) and a Private Replication Agreement Key ($dRAK_2$)	**Input**: N/A **Output**: N/A	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The RWK is used to encrypt an HBK. It is derived from a key agreement operation between the $QRAK_k$ from an HSM in another security domain and the $dRAK_k$ in the local HSM security domain

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Customer Replication Key (CRK) CSP/PSP	256 bits (AES, HMAC) 112 to 128 bits (RSA: 2048, 3072, or 4096 bits) 128 to 256 bits (ECDSA: P256, P384, P521, or secp256k1)	AES GCM HMAC RSA ECDSA (A1908)	Internally from an HBK encrypted with a domain key (DKn)	**Input**: CRK may be imported by decrypting an HBK using a domain key (DK_n) and re-encrypting it using a Replication Wrapping Key (RWK) (electronically) **Output**: CRK is exported encrypted with a Replication Wrapping Key (RWK) (electronically)	KAS (SP 800-56Arev3) (Cofactor) One-Pass Diffie-Hellman (ECC CDH) scheme with key confirmation	Volatile memory	Overwriting with all zeros	The CRK is the customer key that is being transmitted between two HSMs CRKs are wrapped with the RWK
Operator Ephemeral Agreement Public Key (QOEAK) PSP	192 bits (ECDH P384)	ECDH (A1908)	Externally by the module operator	**Input**: When an operator calls the NegotiateSessionKey service (electronically) **Output**: N/A	N/A	Volatile memory	Overwriting with all zeros	The QOEAK is provided by an operator to establish a session key (HOSK) It is used with the HSM ephemeral agreement key (dE) using ECC CDH

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Operator Signature Public Key (QOS) PSP	192 bits (ECDSA P384) 112 to 128 bits (RSA: 2048, 3072, or 4096 bits)	ECDSA RSA (A1908)	Externally by the module operator	**Input**: The public key (QOS) is imported in plaintext when an administrator calls InitializeAndCreateDomain, CreateDomain, and ChangeDomain They are also imported by APIs that accept a Domain Token (electronically) **Output**: The public keys are exported from the HSM in plaintext by APIs that export a Domain Token (electronically)	N/A	Volatile memory	Overwriting with all zeros	The QOS is used by the HSM to authenticate operators

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Customer Data Encryption Public Key (QCDEK) PSP	112 to 128 bits (RSA: 2048, 3072, or 4096 bits)	RSA (A1908)	Externally by the module operator	**Input:** The public key (QCDEK) is optionally provided when an operator calls Generate, GenerateAndEncryptRandomBytes, GenerateDataKeyPair, and Decrypt (electronically) **Output:** N/A	N/A	Volatile memory	Overwriting with all zeros	The QCDEK is provided by an operator or customer to encrypt the SCDEK, which encrypts customer data
Customer Data Encryption Symmetric Key (SCDEK) PSP	128 bits, 256 bits (AES)	AES GCM AES CBC (A1908)	Internally using DRBG	**Input:** N/A **Output:** Encrypted by QCDEK (electronically)	N/A	Volatile memory	Overwriting with all zeros	The SCDEK encrypts customer plaintext data. If a QCDEK is optionally provided for Generate, GenerateAndEncryptRandomBytes, GenerateDataKeyPair, or Decrypt, a SCDEK will be generated within the module to encrypt the resulting customer plaintext data.

Table 13 – SSPs

Entropy sources	Minimum number of bits of entropy	Details
Intel Deterministic Random Number Generator	384 bits of seed material is requested from the entropy source which provides full entropy	Used only to seed the DRBG in the module. 512 bits of entropy data with 0.7 bits of min entropy per bit is provided to the vetted conditioning function, 128-bit AES-CBC-MAC. The conditioning function is called three times for the 384-bit entropy input into the DRBG.

Table 14 – Non-Deterministic Random Number Generator Specification

10. Self-Tests

FIPS 140-3 requires the module to perform self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. Some functions require conditional tests during normal operation of the module. All of these tests are listed and described in this section. In the event of a self-test error, the module will log the error and enter the error state. Once in the error state, all SSPs are zeroized and the module becomes unusable.

Pre-Operational Self-Tests

Pre-operational self-tests are run upon the initialization of the module and do not require operator intervention to run. If any of the tests fail, the module will not initialize. The module will enter an error state and no services can be accessed by the operator. The module implements the following pre-operational self-tests:

Integrity Check

256-bit error detection code (EDC) on all module components

The module performs all pre-operational self-tests automatically when the module is initialized. All pre-operational self-tests must be passed before a Crypto Officer can perform services. The pre-operational self-tests can be run on demand by rebooting the module.

Conditional Self-Tests

The module performs all conditional self-tests automatically when the module is initialized. All conditional self-tests must be passed before a Crypto Officer can perform services.. If any of these tests fail, the module will enter an error state, where no services can be accessed by the operators. The module can be re-initialized to clear the error and resume Approved mode of operation. Each module performs the following conditional self-tests:

Cryptographic Algorithm Self Tests

- AES (Encryption in ECB mode with 128 bit key) KAT
- AES (Decryption in ECB mode with 128 bit key) KAT
- AES GCM (Generation with 128 bit key) KAT
- AES GCM (Verification with 128 bit key) KAT
- ECC KAS (ECDH) (Primitive Z test with EC P-256 parameter set) KAT
- ECDSA (Signature generation with P-256 curve) KAT
- ECDSA (signature verification with P-256 curve) KAT
- RSA (Signature generation, key transport SP800-56B per IG D.G with 2048 bit key) KAT
- RSA (Signature verification, key transport SP800-56B per IG D.G with 2048 bit key) KAT
- HMAC (Generation with SHA2-256, SHA2-512) KAT
- HMAC (Verification with SHA2-256, SHA2-512) KAT
- SHS (SHA-1, SHA2-256, SHA2-512) KAT
- SP 800-90 CTR_DRBG KAT

- DRBG Health Tests
  Performed on DRBG, per SP 800-90A Section 11.3
- SP 800-108 KBKDF (HMAC-SHA2-256) KAT
- KDA (OneStep KDF) (SHA2-256) KAT

Pair-wise Consistency Tests

- RSA key pair generation
- ECDSA / ECDH key pair generation

SP 800-56A Assurances

- Performed per SP 800-56Arev3 Sections 5.5.2 and 5.6.2

SP 800-90B Health Tests (Critical function test)

- NIST SP 800-90B ENT Health Tests, per SP 800-90B Section 4.5

The module does not perform a firmware load test because no additional firmware can be loaded in the module while operating in the Approved mode. Please see Section 3 for guidance on configuring and maintaining Approved mode.

On-demand Self-Tests

On-demand self-tests can be performed by rebooting the module which will perform the pre-operational self-tests.

Periodic Self-Tests

All conditional self-tests are automatically run once a day. The specific time is randomly selected by the module between 23 to 24 hours since the last run. The tests are executed in the background.

11. Life-cycle Assurance

Delivery and Operation

The AWS Key Management Service HSM is designed to be mounted in a rack only. Before mounting onto a rack, the module should be inspected for signs of physical tampering. Connect the power interface to the power connector in the rack.

Power up the module. The module will start up in the approved mode of operation. No other configuration is necessary.

End of Life

To prepare a module for disposal:

1. Remove all domain information on the module using the ForgetDomain API
2. Delete the HSM Signature Key and HSM Agreement Key from the HSM using the Wipe API
3. Return the HSM to the factory state using the DeactivateAndReboot API. This step also zeroizes volatile memory as part of the reboot process
4. Power down the module by disconnecting the module from the power source

To securely destroy a module:

1. To open the chassis, drill though all fasteners that secure the cover to the chassis and remove the cover.
2. Remove and destroy the solid state drive and memory modules in accordance with NIST SP 800-88rev1.

12. Mitigation of Other Attacks

Not Applicable.

Appendix A - Acronyms

AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
AWS	Amazon Web Services
CBC	Cipher Block Chaining
CDK	Customer Data Key
CMK	Customer Managed Key
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CSE	Communications Security Establishment Canada
CSK	Customer Supplied Key
CSP	Critical Security Parameter
CTR	Counter
DH	Diffie-Hellman
DKn	Domain Key
DKEK	Domain Key Encryption Key
DRBG	Deterministic Random Bit Generator
ECB	Electronic Codebook
EC	Elliptic Curve
ECDSA	Elliptic Curve Digital Signature Algorithm
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
HBK	HSM Backing Key
HMAC	(Keyed-) Hash Message Authentication Code
HOSK	HSM-to-Operator Session Key
HSK	HSM Signature Key Pair
HSKEK	HSM Session Key Encryption Key
HSM	Hardware Security Module
IPMI	Intelligent Platform Management Interface
KAS	Key Agreement Scheme
KAT	Known Answer Test
KBKDF	Key Based Key Derivation Function
KDF	Key Derivation Function
KMS	Key Management Service
KTS	Key Transport Scheme

MAC	Message Authentication Code
MD	Message Digest
NIST	National Institute of Standards and Technology
NMI	Non-Maskable Interrupt
OAEP	Optimal Asymmetric Encryption Padding
PKCS	Public-Key Cryptography Standards
PSS	Probabilistic Signature Scheme
QOEAK	Operator Ephemeral Agreement Public Key
QOS	Operator Signature Public Key
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
SHA	Secure Hash Algorithm
SP	Special Publication
SSP	Sensitive Security Parameter