FIPS 140-2 Non-Proprietary Security Policy

AWS Key Management Service HSM

(Hardware version 3.0, firmware version 1.7.100, 1.7.102 and 1.7.103)

Document Version 1.14

May 9th, 2023

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 About FIPS 140

Federal Information Processing Standards (FIPS) Publication 140-2 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic modules to be deployed in a Sensitive but Unclassified environment. The National Institute of Standards and Technology (NIST) and Canadian Centre for Cyber Security (CCCS) Cryptographic Module Validation Program (CMVP) run the FIPS 140-2 program. The National Voluntary Laboratory Accreditation Program (NVLAP) accredits independent testing labs to perform FIPS 140-2 testing; the CMVP validates modules meeting FIPS 140-2 requirements. *Validated* is the term given to a module that is documented and tested against the FIPS 140-2 criteria.

More information is available on the CMVP website at http://csrc.nist.gov/groups/STM/cmvp/index.html.

1.2 About this Document

This non-proprietary Cryptographic Module Security Policy for the AWS Key Management Service (KMS) Hardware Security Module (HSM) from Amazon Web Services (AWS) provides an overview of the HSM and a high-level description of how it meets the security requirements of FIPS 140-2. This document contains details on the module's cryptographic keys and critical security parameters. This Security Policy concludes with instructions and guidance on running the module in a FIPS 140-2 mode of operation.

AWS Key Management Service HSM may also be referred to as the "module" in this document.

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.

1.3 External Resources

The AWS website (http://aws.amazon.com/kms/) contains information on AWS services that utilizes the module. The list of public AWS KMS APIs is found on the AWS documentation website: (http://docs.aws.amazon.com/kms/latest/APIReference/Welcome.html).

The Cryptographic Module Validation Program website contains links to the FIPS 140-2 certificate and AWS contact information.

1.4 Notices

This document may be freely reproduced and distributed in its entirety without modification.

1.5 Acronyms

Table 1 defines acronyms found in this document:

Acronym	Term
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
AWS	Amazon Web Services
CBC	Cipher Block Chaining
CCCS	Canadian Centre for Cyber Security
CDK	Customer Data Key
CKG	Cryptographic Key Generation
CMK	Customer Master Key
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CRK	Customer Replication Key
CSK	Customer Supplied Key
CSP	Critical Security Parameter
CTR	Counter
DH	Diffie-Hellman
DK	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
ENT	Entropy Source
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
GMAC	Galois Message Authentication Code
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
KDA	Key Derivation Algorithm
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
QCDEK	Customer Data Encryption Public Key
QOEAK	Operator Ephemeral Agreement Public Key
QOS	Operator Signature Public Key
RAK	Replication Agreement Key
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
RSK	Replication Signing Key
RWK	Replication Wrapping Key
SHA	Secure Hash Algorithm
SP	Special Publication

Table 1 – Acronyms and Terms

2 AWS Key Management Service HSM

2.1 Cryptographic Module Specification

AWS customers can use the AWS Key Management Service to generate and manage cryptographic keys and operate as a cryptographic service provider for protecting data within AWS. The AWS Key Management Service HSM provides dedicated cryptographic functions for the AWS Key Management Service.

The module runs firmware versions 1.7.100, 1.7.102 and/or 1.7.103 on hardware version 3.0 and is classified as a multi-chip standalone cryptographic module. The physical cryptographic boundary is defined as the module case, and the module runs on a non-modifiable operating environment.

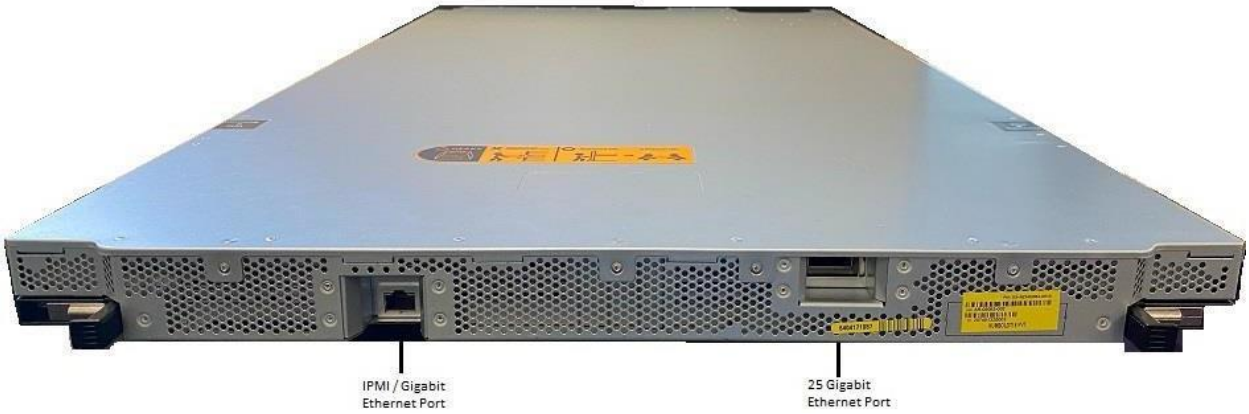

Figure 1 – Cryptographic Module Boundary (Front)

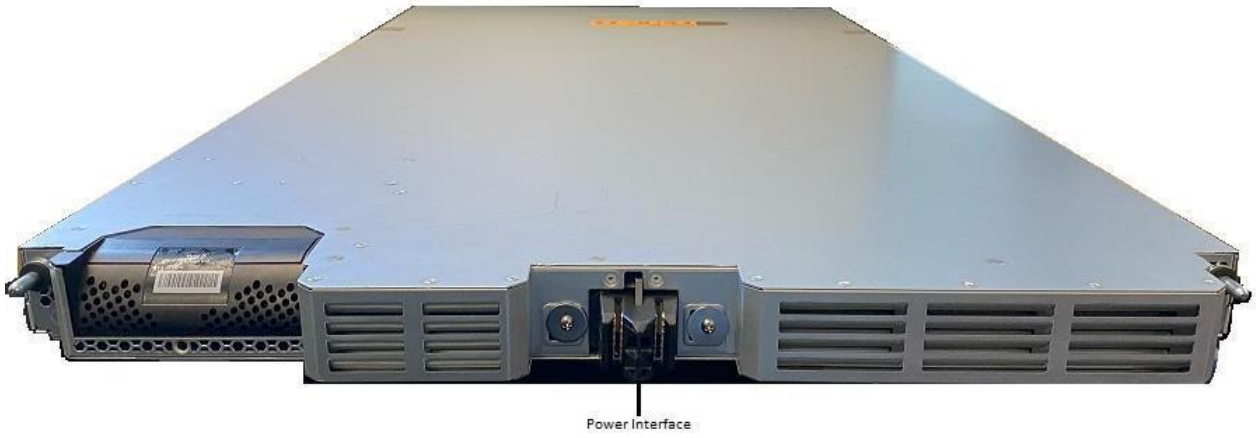

Figure 2 - Cryptographic Module Boundary (Back)

2.1.1 Validation Level Detail

Table 2 lists the level of validation for each area in FIPS 140-2:

FIPS 140-2 Section Title	Validation 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
Electromagnetic Interference / Electromagnetic Compatibility	3
Self-Tests	3
Design Assurance	3
Mitigation of Other Attacks	N/A
Overall Level	**3**

Table 2 – Validation Level by FIPS 140-2 Section

2.1.2 Approved Cryptographic Algorithms

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.	Algorithm	Standard	Mode/Method	Key Lengths, Curves or Moduli	Use
AWS Key Management Service Cryptographic Algorithm Library					
A1908	**AES**	FIPS 197 SP 800-38A	ECB, CBC, CTR	128, 256	Encryption, Decryption
A1908	**GCM/GMAC**[1]	SP 800-38D	AES	128, 256	Generation, Authentication, Encryption, Decryption
A1908	**KTS**	SP 800-38F IG D.9	AES KWP AES GCM	256	Key Transport using AES KWP, Key Transport using AES GCM
A1908	**DRBG**	SP 800-90A	CTR DRBG	256	Random Bit Generation
A1908	**ECDSA**	FIPS 186-4	SigGen SigGen Component KeyGen SigVer SigVer Component KeyVer	P-256, P-384, P-521	Signature Generation Component, Signature Verification Component, Key Pair Generation, Signature Generation, Signature Verification, Public Key Validation
A1908	**HMAC**	FIPS 198-1	SHS	160, 224, 256, 384 and 512 bits	Generation, Authentication
A1908	**RSA**	FIPS 186-4	PKCS#1 v1.5 and PSS	2048, 3072 and 4096[2] bits	Key Pair Generation, Signature Generation, Signature Verification, Component Test
A1908	**SHA**	FIPS 180-4	SHA-1, SHA-256, SHA-384, and SHA-512	--	Digital Signature Generation, Digital Signature Verification, non- Digital Signature Applications
A1908	**CVL**	FIPS 186-4	ECDSA SigGen Component	P-256, P-384, P-521	Signature Generation Component
			RSA Signature Primitive	-	Component Test

[1] The AES GCM IV is generated internally in the cryptographic module using the module's Approved NIST SP 800-90A DRBG and the IV length used is at least 96 bits (per SP 800-38D and IG A.5 Scenario 2).

[2] RSA 4096 was not tested by the CAVP; however, it is Approved for use per IG A.14, because RSA SigGen / SigVer were tested in accordance with FIPS 186-4 for the 2048 and 3072 bit modulus sizes, and testing for modulus sizes higher than 3072 is not available under CAVS.

CAVP Cert.	Algorithm	Standard	Mode/Method	Key Lengths, Curves or Moduli	Use
AWS Key Management Service Cryptographic Algorithm Library					
A1908	**KTS**	SP 800-56B	[SP 800-56B] RSA-OAEP with, and without key confirmation Key sizes: 2048, 3072, and 4096 bits Hybrid Key-Transport scheme incorporating KTS-OAEP and SP 800-38F	Key Sizes: 2048, 3072 and 4096 bits	Key Transport, Optional RSA encapsulation schemes for protecting keys that customers import into AWS KMS.
A1908	**KAS**	SP 800-56A	[SP 800-56Arev3] (Cofactor) Ephemeral Unified and (Cofactor) One-Pass Diffie-Hellman schemes without key confirmation	ECDH Curve: P384	Key Agreement
A1908	**KDA**	SP 800-56C	One-step key derivation SHA-256, SHA-384	256 bits, 384 bits	Key Derivation
AWS Key Management Service Key Derivation Function Library					
A1910	**KBKDF, using Pseudorandom Functions**	SP 800-108	Counter Mode HMAC-based KDF with SHA-256	256 bits	Key Derivation
Intel DRNG					
N/A	**ENT (P)**	SP 800-90B	Entropy source compliant with SP 800-90B, IG 7.14 Scenario 1A, IG 7.18 and 7.19	384 bits	Provides seeding material for the DRBG
A1791	**Conditioning Components**	SP 800-90B	AES-ECB AES-CBC-MAC Counter DRBG	128 bits	Provides seeding material for the DRBG

Table 3 – FIPS-Approved Algorithms and Certificate Numbers

The following Approved cryptographic algorithms were tested with vendor affirmation.

Algorithm	IG Reference	Use
CKG	Vendor Affirmed IG D.12	[SP 800-133rev2, Section 5] Seeding for asymmetric key generation uses unmodified DRBG output [SP 800-133rev2, Section 6.1] Symmetric key generation uses unmodified DRBG output

Table 4 – Approved Cryptographic Functions Tested with Vendor Affirmation

2.1.3 Non-Approved but Allowed Algorithms

The module supports the following non-FIPS 140-2 approved but allowed algorithms that may be used in the Approved mode of operation.

Algorithm	Use
RSA Key Transport with PKCS #1 v1.5	[IG D.9] Optional RSA encapsulation scheme for protecting keys that customers import into AWS KMS. Key sizes: 2048, 3072 and 4096 bits (key wrapping; key establishment methodology provides between 112 and 150 bits of encryption strength)
ECDSA secp256k1	[IG A.2] Curves: secp256k1; ECDSA (signature generation and verification; provides 128 bits of security strength)

Table 5 – Non-Approved but Allowed Cryptographic Algorithms

2.2 Module Interfaces

Table 6 describes the main interfaces of the module:

Physical Interface	Description / Use
25 Gigabit Ethernet Port	Main session interface for cryptographic services
IPMI / Gigabit Ethernet Port	Provides power on / off, and query status
Power Interface	Accept and provide power to the module

Table 6 – Interface Descriptions

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 140-2 defined logical interfaces: data input, data output, control input, and status output. The logical interfaces and their mapping are provided in the following table:

FIPS 140-2 Logical Interface	Module Physical Interface
Data Input	25 Gigabit Ethernet Port
Data Output	25 Gigabit Ethernet Port
Control Input	25 Gigabit Ethernet Port IPMI / Gigabit Ethernet Port
Status Output	25 Gigabit Ethernet Port IPMI / Gigabit Ethernet Port
Power	Power Interface

Table 7 – Logical Interface / Physical Interface Mapping

2.3 Roles, Services, and Authentication

Operators of the module may assume the following three roles:

Roles	Description	Authentication
KMS Front End Role (KMS-FE)	The KMS front end hosts perform actions on behalf of customers of AWS KMS.	RSA 2048, 4096 or ECDSA P384
KMS Coordinator Role (KMS-C)	Non-public facing KMS hosts perform actions on behalf of KMS administrators in the Administrator Role.	RSA 2048, 4096 or ECDSA P384
Administrator Role (Admin)	Employees of AWS who are authorized to manage the module.	RSA 2048, 4096 or ECDSA P384

Table 8 – Roles and Authentication

For FIPS purposes, the KMS Coordinator and Administrator roles serve as the Cryptographic Officer role per FIPS 140-2 requirements. The KMS-Front End role serves as the User role per FIPS 140-2 requirements.

The module supports identity-based authentication, and the respective services for each role are described in the following sections. The module does not support a Maintenance role.

Services supported by the module may also be referred to as APIs in this document.

The module supports authentication using RSA with 2048 and 4096 bit keys, and 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.

The list of services supported by the module are listed in Table 9, Table 10, and Table 11. Unless otherwise specified, access to services can be configured to require one or more members of one or more roles listed in Table 8. 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.

All unauthenticated services listed do not affect the security of the information protected by the module and are allowed per IG 3.1.

2.3.1 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.

The possibility of a random authentication succeeding within a one-minute period is 2^{-80} 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, assuming a rate of 2^{32} attempts per minute, is not significant enough to allow a random attempt to succeed within a one-minute period.

2.3.2 Cryptographic Services and Descriptions

For all cryptographic services in this section, all key/CSP input and output are encrypted using the HSM-to-Operator Session Key (HOSK) using 256-bit AES GCM. The use of the HOSK provides transport security between the HSM and other KMS Operators (as defined in Section 2.3 above).

HSM Service (API)	Roles	Description
Create	KMS Front End, KMS Coordinator, Administrators	Generates and encrypts either an HSM Backing Key (HBK) or an Import Wrapping Key (IWK) private key. **Key/CSP Input:** None **Key/CSP Output:** The Create API returns either: • A HSM Backing Key encrypted with the active Domain Key (DK_n), or • An Import Wrapping Key (IWK) key pair: a. The IWK private key is encrypted with the active Domain Key (DK_n). b. The IWK public key. **Key/CSP Generated:** • HSM Backing Key • IWK public and private keys **Key/CSP Read Access:** • Active Domain Key (DK_n) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The Create API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.

HSM Service (API)	Roles	Description
ImportKey	KMS Front End, KMS Coordinator, Administrators	Decrypts a Customer Supplied Key (CSK) and re-encrypts it with the active Domain Key (DK_n) **Key/CSP Input:** • The private key of an Import Wrapping Key Pair (IWK) encrypted with the active or a recent iteration of domain key (DK_n or DK_{n-1}) • Customer Supplied Key (CSK), encrypted with the public key of the Import Wrapping Key. This may use the wrapping methods as defined in SP 800-56B, using the ephemeral Import Wrapping Envelope Key (IWEK). **Key/CSP Output:** The Customer Supplied Key, encrypted with the current active domain key (DK_n) **Key/CSP Read Access:** • Active or a recent iteration of Domain Key (DK_n or DK_{n-1}) used to encrypt the IWK private key • HSM-to-Operator Session Key (HOSK) **Additional Information:** The ImportKey API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
RefreshKey	KMS Front End, KMS Coordinator, Administrators	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). **Key/CSP Input:** HBK or CSK encrypted with a recent iteration of a Domain Key (DK_{n-1}) **Key/CSP Output:** HBK or CSK encrypted with the active domain key (DK_n) **Key/CSP Read Access:** • Active or a recent iteration of Domain Key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The RefreshKey API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Encrypt	KMS Front End, KMS Coordinator, Administrators	Encrypt an arbitrary set of bytes using the DEK derived from the provided HBK or CSK. **Key/CSP Input:** A HBK or CSK encrypted with the active or a recent iteration of domain key (DK_n or DK_{n-1}) **Key/CSP Output:** N/A (encrypted ciphertext) **Key/CSP Read Access:** • Active or a recent iteration of Domain Key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK)

HSM Service (API)	Roles	Description
		Additional Information: The Encrypt API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Decrypt	KMS Front End, KMS Coordinator, Administrators	Decrypts ciphertext using the DEK derived from the provided HBK or CSK. Optionally encrypts the plaintext result using the provided QCDEK. **Key/CSP Input:** • A HBK or CSK encrypted with a Domain Key (DK_n) • Ciphertext or encrypted Customer Data Key (CDK) • (optional) Customer Data Encryption Public Key (QCDEK) **Key/CSP Output:** • Arbitrary data or CDK encrypted using the HOSK • (optional) Arbitrary data or CDK encrypted using the Customer Data Envelope Key (CDEnK) and QCDEK if QCDEK is provided **Key/CSP Generated:** (optional) Customer Data Envelope Key (CDEnK) if QCDEK is provided **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The Decrypt API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
ReEncrypt	KMS Front End, KMS Coordinator, Administrators	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. **Key/CSP Input:** • 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) **Key/CSP Output:** N/A (encrypted ciphertext) **Key/CSP Read Access:** • Active or a recent iteration of Domain Key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK)

HSM Service (API)	Roles	Description
		Additional Information: The ReEncrypt API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Sign	KMS Front End, KMS Coordinator, Administrators	Performs an ECDSA or RSA sign operation, or HMAC operation using the provided HBK or CSK **Key/CSP Input:** HBK or CSK encrypted with the active domain key (DK_n) **Key/CSP Output:** None **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The Sign API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Verify	KMS Front End, KMS Coordinator, Administrators	Performs an ECDSA or RSA verify, or HMAC operation using the provided HBK or CSK **Key/CSP Input:** HBK or CSK encrypted with the active domain key (DK_n) **Key/CSP Output:** None **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The Verify API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
EncryptRandomBytes	KMS Front End, KMS Coordinator, Administrators	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). **Key/CSP Input:** HBK or CSK encrypted by the active domain key (DK_n) **Key/CSP Output:** A number of random bytes that may be used as Customer Data Keys (CDK) encrypted by the HBK or CSK. **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The EncryptRandomBytes API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.

HSM Service (API)	Roles	Description
GenerateAndEncryptRandomBytes	KMS Front End, KMS Coordinator, Administrators	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 3 forms. **Key/CSP Input:** • HBK or CSK encrypted by the active domain key (DK_n) • Customer Data Encryption Public Key (QCDEK) **Key/CSP Output:** • 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. • A number of random bytes that may be used as Customer Data Keys (CDK) encrypted using the Customer Data Envelope Key (CDEnK) and QCDEK **Key/CSP Generated:** Customer Data Envelope Key (CDEnK) **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The GenerateAndEncryptRandomBytes API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
GenerateDataKeyPair	KMS Front End, KMS Coordinator, Administrators	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 3 forms. **Key/CSP Input:** • HBK or CSK encrypted by the active domain key (DK_n) • (optional) Customer Data Encryption Public Key (QCDEK) **Key/CSP Output:** • 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 • (optional) An asymmetric Customer Data Key (CDK) private key encrypted using the Customer Data Envelope Key (CDEnK) and QCDEK **Key/CSP Generated:** (optional) Customer Data Envelope Key (CDEnK) if QCDEK is provided **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1})

HSM Service (API)	Roles	Description
		• HSM-to-Operator Session Key (HOSK) **Additional Information:** The GenerateDataKeyPair API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
GenerateDataKeyPairWithoutPlaintext	KMS Front End, KMS Coordinator, Administrators	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). **Key/CSP Input:** • HBK or CSK encrypted by the active domain key (DK_n) **Key/CSP Output:** • An asymmetric Customer Data Key (CDK) private key encrypted by the HBK or CSK **Key/CSP Read Access:** • Active or a recent iteration of domain key (DK_n or DK_{n-1}) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The GenerateDataKeyPairWithoutPlaintext API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Generate	KMS Front End, KMS Coordinator, Administrators	Generate a specified number of random bytes, up to 1024 bytes. The random bytes can be optionally encrypted with the Customer Data Envelope Key (CDEnK) and Customer Data Encryption Public Key (QCDEK). **Key/CSP Input:** (optional) Customer Data Encryption Public Key (QCDEK) **Key/CSP Output:** None (random bytes optionally encrypted using CDEnK and QCDEK) **Key/CSP Generated:** (optional) Customer Data Envelope Key (CDEnK) if QCDEK is provided **Key/CSP Read Access:** • HSM-to-Operator Session Key (HOSK) **Additional Information:** The Generate API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
GetParametersForReplication	KMS Front End, KMS Coordinator, Administrators	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). **Key/CSP Input:** None

HSM Service (API)	Roles	Description
		Key/CSP Output: • Public Replication Agreement Key ($QRAK_1$) • Private Replication Agreement Key ($dRAK_1$) encrypted by the active domain key (DK_n) **Key/CSP Generated:** • Replication Agreement Key Pair ($dRAK_1$, $QRAK_1$) **Key/CSP Read Access:** • 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 $dRSK_{n-1}$) **Additional Information:** The GetParametersForReplication API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters. The API also signs all output with the Private Replication Signing Key ($dRSK_n$ or $dRSK_{n-1}$).
WrapKeyFor Replication	KMS Front End, KMS Coordinator, Administrators	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). **Key/CSP Input:** • Public Replication Agreement Key ($QRAK_1$) • HBK encrypted by the active domain key (DK_n) **Key/CSP Generated:** • Replication Agreement Key Pair ($dRAK_2$, $QRAK_2$) **Key/CSP Output:** • Public Replication Agreement Key ($QRAK_2$) • Customer Replicated Key (CRK) encrypted by the Replication Wrapping Key (RWK) **Key/CSP Read Access:** • 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 $QRSK_{n-1}$) **Additional Information:** The WrapKeyForReplication API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters. The API also verifies input using the Public Replication Signing Key ($QRSK_n$ or $QRSK_{n-1}$), and signs all output with the Private Replication Signing Key ($dRSK_n$ or $dRSK_{n-1}$).

HSM Service (API)	Roles	Description
ImportReplicatedKey	KMS Front End, KMS Coordinator, Administrators	This API combines two Replication Agreement Key ($dRAK_1$ and $QRAK_2$) using the Diffie-Hellmann key exchange to produce a shared secret and derive a Replication Wrapping Key (RWK). The RWK is used to decrypt the Customer Replicated Key (CRK), obtaining an HBK, which is then re-encrypted using the Domain Key (DK_n). **Key/CSP Input:** • 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) **Key/CSP Output:** • HBK encrypted by the active domain key (DK_n) **Key/CSP Read Access:** • 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 $QRSK_{n-1}$) **Additional Information:** The ImportReplicatedKey API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters. The API also validates input using the Public Replication Signing Key ($QRSK_n$ or $QRSK_{n-1}$).

Table 9 - Cryptographic Services and Descriptions

2.3.3 Configuration Services and Descriptions

HSM Service (API)	Roles	Description
CreateDomain	KMS Front End, KMS Coordinator, Administrators	Creates a new domain token for a new domain, but does not join the HSM to the domain yet. **Key/CSP Input:** • List of Operator Signature Public Keys (QOS) **Key/CSP Generated:** • HSM Signature Key Pair (HSK) • HSM Agreement Key Pair (HAK) • Initial Domain Key (DK_0) • Replication Signing Key ($dRSK_0$, $QRSK_0$) **Key/CSP Output:** 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

HSM Service (API)	Roles	Description
		• Encrypted Initial Domain Key (DK_0) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_0$) • Public Replication Signing Key ($QRSK_0$) **Key/CSP Read Access:** None
IngestDomain	KMS Front End, KMS Coordinator, Administrators	Joins a domain or receive an updated domain token. **Key/CSP Input: 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 (DK_n) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Generated:** None **Key/CSP Output:** The unmodified input Domain Token **Key/CSP Read Access:** • HSM Signature Public Key (QHSK) of a known member of the domain • HSM Agreement Private Key (dHAK) • Operator Signature Public Keys (QOS) **Key/CSP Write Access:** • 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 • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Additional Information:** When using the IngestDomain API to set up the first domain member, the operator(s) must meet the quorum configuration in the to-be-ingested domain. When using the IngestDomain API to ingest subsequent domains, the operator(s) must meet the quorum configuration in both the first domain, and in the to-be-ingested domain.

HSM Service (API)	Roles	Description
ForgetDomain	KMS Front End, KMS Coordinator, Administrators	Deletes domain information as it pertains to a particular domain on the module including all Domain Keys (DK_n, DK_{n-1}), effectively leaving the domain. **Key/CSP Input: 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 (DK_n) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Output:** The unmodified input Domain Token **Key/CSP Read Access:** • Domain Key (DK_n) • Operator Signature Public Keys (QOS) **Key/CSP De-Referenced:** • 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
GetDomain	KMS Front End, KMS Coordinator, Administrators	Retrieves the current version of the domain token for a specified domain. **Key/CSP Input:** None **Key/CSP Output:** 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) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Read Access:** • Domain Key (DK_n) • Operator Signature Public Keys (QOS)

ChangeDomain	KMS Front End, KMS Coordinator, Administrators	Modifies the current state of an operational domain.
		
		Key/CSP Input:
		• A Domain Token containing:
		  ○ List of Operator Signature Public Keys (QOS)
		  ○ List of HSM Signature Public Keys (QHSK) of all members of the domain

HSM Service (API)	Roles	Description
		○ List of HSM Key Agreement Public Keys (QHAK) of all members of the domain ○ Encrypted Domain Keys (DK_n) ○ Encrypted 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) **Key/CSP Generated:** • Domain Key Encrypting Key (DKEK) • HSM Ephemeral Agreement Key (dE, QE) **Key/CSP Output:** 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 (DK_n) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Read Access:** Domain Key (DK_n), HSM Agreement Key (HAK), HSM Signature Key (HSK)

HSM Service (API)	Roles	Description
Initialize	All / unauthenticated	Initializes the HSM by generating the HSM Signature Key and HSM Agreement 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 InitializeAndCreateDomain API, the Initialize API will return an error as the HSM cannot be Initialized again without a reboot. **Key/CSP Input:** 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) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Generated:** • HSM Signature Key (HSK) • HSM Agreement Key (HAK) • HSM Session Key Encryption Key (HSKEK) **Key/CSP Output:** None **Key/CSP Read Access:** None **Key/CSP Write Access:** • HSM Signature Key (HSK) • HSM Agreement Key (HAK) • HSM Session Key Encryption Key (HSKEK) • Operator Signature Public Keys (QOS) **Additional Information:** The Initialize API is unauthenticated. Initialize will fail if the HSM is already initialized by a call to either the Initialize or InitializeAndCreateDomain API.
InitializeAndCreateDomain	All / unauthenticated	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. **Key/CSP Input:** • List of Operator Signature Public Keys (QOS) **Key/CSP Generated:** • HSM Signature Key Pair (HSK)

HSM Service (API)	Roles	Description
		• HSM Agreement Key Pair (HAK) • HSM Session Key Encryption Key (HSKEK) • Initial Domain Key (DK_0) **Key/CSP Output:** 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) • Encrypted Domain Key Encryption Key (DKEK) • Encrypted Private Replication Signing Key ($dRSK_n$) • Public Replication Signing Key ($QRSK_n$) **Key/CSP Read Access:** None **Key/CSP Write Access:** • HSM Signature Key (HSK) • HSM Agreement Key (HAK) • HSM Session Key Encryption Key (HSKEK) • List of Operator Signature Public Keys (QOS) **Additional Information:** The InitializeAndCreateDomain API is unauthenticated. The InitializeAndCreateDomain API will fail if the HSM is already initialized by a call to either the Initialize or InitializeAndCreateDomain API.
Attest	KMS Front End, KMS Coordinator, Administrators	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. **Key/CSP Input:** None **Key/CSP Output:** • HSM Signature Public Key (QHSK) • HSM Agreement Public Key (QHAK) **Key/CSP Read Access:** • 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) **Additional Information:** If an optional attestation challenge is included in the request, the Attest API requires the use of the HSM-to-Operator Session Key (HOSK) to encrypt all input and output parameters.

HSM Service (API)	Roles	Description
GetAttestationChallenge	KMS Front End, KMS Coordinator, Administrators	The GetAttestationChallenge API is used by operators to retrieve a token that can be used to validate the identity of another HSM. **Key/CSP Input:** None **Key/CSP Output:** None. **Key/CSP Read Access:** • Active or a recent iteration of domain key (DKn or DKn-1) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The GetAttestationChallenge API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
GetAttestationIdentity	KMS Front End, KMS Coordinator, Administrators	The GetAttestationIdentity API is used by operators to retrieve information to attest the identity of the HSM. **Key/CSP Input**: None **Key/CSP Output**: None **Key/CSP Read Access:** • HSM Session Key Encryption Key (HSKEK) • HSM-to-Operator Session Key (HOSK) **Additional Information:** The GetAttestationIdentity API validates the HSM-to-Operator Session Key (HOSK) to authenticate the call originates from an authenticated operator. The HOSK is also used to encrypt all input and output parameters.
Wipe	All / unauthenticated	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. **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** None **Key/CSP De-Referenced:** • HSM Signature Key Pair (HSK) • HSM Agreement Key Pair (HAK) • HSM Session Key Encryption Key (HSKEK) **Additional Information:** This call is unauthenticated.

HSM Service (API)	Roles	Description
GetInitialDomainName	All / unauthenticated	Retrieves the initial domain name from an initialized HSM that is used as part of the domain creation bootstrap process. **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** No access to Keys/CSPs. **Additional Information:** This call is unauthenticated.
DeactivateAndReboot (This service also performs the self-tests to run after the module is rebooted)	All / unauthenticated	The DeactivateAndReboot API returns the HSM to the factory state and reboots after verifying the HSM Signature Key and HSM Agreement Key have been deleted by the Wipe API. **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** No access to Keys/CSPs. **Additional Information:** This call is unauthenticated.
NegotiateSessionKey	One member from any role	Uses a set of identity keys to securely negotiate a session key that can be used between a KMS host and any HSM in the domain. The NegotiateSessionKey API will return encrypted versions of the HSM-Operator Session Key (HOSK) in 2 forms. **Key/CSP Input:** • Operator Ephemeral Agreement Public Key (QOEAK) **Key/CSP Generated:** • HSM Ephemeral Agreement Key Pair (dE, QE) • HSM-Operator Session Key (HOSK) **Key/CSP Output:** • Encrypted HSM-Operator Session Key (HOSK) encrypted with the Domain Key (DK) 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 Key (QE) and the Operator Ephemeral Agreement Public Key (QOEAK). • HSM Ephemeral Agreement Public Key (QE) **Key/CSP Read Access:** • Operator Signature Public Key (QOS) • HSM Signature Key (dHSK) **Key/CSP Write Access:** • HSM-Operator Session Key (HOSK)

HSM Service (API)	Roles	Description
UpdateHostConfiguration	KMS Front End, KMS Coordinator, Administrators	Allows updates of non-security-relevant host configuration. **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** Operator Signature Public Key (QOS)

Table 10 - Configuration Services and Descriptions

2.3.4 Audit Log Services and Descriptions

Service (API)	Role	Description
ListLogs	KMS Front End, KMS Coordinator, Administrators	Returns a list of audit log file names. **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** Operator Signature Public Key (QOS)
GetLog	KMS Front End, KMS Coordinator, Administrators	Retrieves specified audit log files **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** Operator Signature Public Key (QOS)
DeleteLog	KMS Front End, KMS Coordinator, Administrators	Deletes specified audit log file **Key/CSP Input:** None **Key/CSP Output:** None **Key/CSP Read Access:** Operator Signature Public Key (QOS)

Table 11 – Audit Log Services and Descriptions

2.3.5 Show Status

The module supports the following APIs to return status information.

Status Service (API)	Description
Ping	Returns “healthy” if the module is initialized and has ingested a domain Returns “failure” otherwise **Additional Information:** This call is unauthenticated.
Fips	Returns “healthy” if the module is operating in FIPS mode Returns “failure” if the module is not operating in FIPS mode **Additional Information:** This call is unauthenticated.

Table 12 - Status Services and Descriptions

In addition, an operator with access to the serial console can obtain hardware status information such as temperature, fan speed, etc.

2.3.6 Zeroization

Zeroization is accomplished by powering off the module.

2.4 Physical Security

The module is a multiple-chip standalone module and conforms to Level 3 requirements for physical security. The module's production-grade enclosure is made of a hard metal with no removable cover. Attempts at removal or penetration of the enclosure will have a high probability of causing serious damage to the module (i.e. the module will not function). The baffles installed by AWS satisfy FIPS 140-2 requirements for module opacity.

2.5 Operational Environment

The module operates in a non-modifiable operational environment.

The module meets Federal Communications Commission (FCC) Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for home use as defined by 47 Code of Federal Regulations, Part 15, Subpart B.

2.6 Cryptographic Key Management

2.6.1 Critical Security Parameters

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

Keys and CSPs	Key Description	Algorithm and Key Size	Generation	Input / Output Method	Storage
HSM Backing Key (HBK)	HSM Backing Keys (HBK) can be of two different forms – symmetric or asymmetric keys. The first form is a 256-bit master key, from which specific-use keys may be derived using the SP800-108 CTR key derivation function. The HBK or keys derived from the HBK are used to encrypt CDKs. The second form is an asymmetric private key. The HBK maps to the Customer Master Key construct exposed in the public AWS KMS API.	AES GCM 256 bits RSA: 2048, 3072, or 4096 bits ECDSA: curves P-256, P-384, P-521, or secp256k1	Internally using DRBG, or imported from another member of a Domain	**Input:** Encrypted with the Domain Key using AES GCM. **Output:** Encrypted with the Domain Key using AES GCM.	Volatile memory only
Customer Data Key (CDK)	Customer data keys are exported by the EncryptRandomBytes, GenerateAndEncryptRandomBytes, GenerateDataKeyPair, and GenerateDataKeyPairWithoutPlaintext APIs. Customer Data Keys (CDK) can be of two different forms – symmetric and asymmetric keys. The use of CDKs are unknown to the module. The customer may obtain the CDK by sending the encrypted CDK to KMS to decrypt under an authenticated and authorized request.	For symmetric keys, random bits length specified by customer (in the range of 8 bits to 65536 bits). For RSA: 2048, 3072, or 4096 bits For ECDSA: curves P-256, P-384, P-521, or secp256k1	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. **Output:** Encrypted in 2 forms by the GenerateAndEncryptRandomBytes and GenerateDataKeyPair APIs: • Encrypted with the DEK derived from an HBK or CSK; and • 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.	Volatile memory only

Keys and CSPs	Key Description	Algorithm and Key Size	Generation	Input / Output Method	Storage
Data Encryption Key (DEK)	A DEK is a per-message key derived from an HBK or CSK using the SP 800-108 KDF in counter mode using HMAC with SHA256.	AES GCM 256 bits	Derived internally using SP 800-108 CTR KDF	**Input:** N/A **Output: N/A**	Volatile memory only
Domain Key (DK)	A Domain Key is shared among all the members of a Domain and is used to encrypt HBKs, CSKs HOSKs	AES GCM 256 bits	Internally using DRBG, or imported from another member of a Domain	**Input:** DK encrypted with the DKEK may be imported to other members of a Domain **Output:** DK encrypted with the DKEK may be exported to other members of a Domain	Volatile memory only
Domain Key Encryption Key (DKEK)	A Domain Key Encryption Key is generated on a Host and is used for encrypting the current set of domain keys when sharing of the domain state between HSM hosts.	AES GCM 256 bits	Internally using DRBG or externally by another member of a Domain	**Input:** The DKEK is encrypted with the shared secret generated from the HSM's Key Agreement Key (QHAK) and another HSM's Ephemeral Key Agreement Key (dE). **Output:** The DKEK is encrypted with the shared secret generated from the HSM's Key Agreement Key (dHAK) and another HSM's Ephemeral Key Agreement Key (QE).	Volatile memory only
HSM Agreement Key Pair (dHAK, QHAK)	Every initialized HSM has a locally generated Elliptic Curve Diffie-Hellman agreement key pair used to encrypt/decrypt DKEKs between HSMs.	Elliptic Curve Diffie-Hellman agreement key pair on the curve secp384r1 (NIST-P384)	Internally using DRBG	**Input: N/A** **Output:** The public key (QHAK) is exported in plaintext	Volatile memory only
HSM Ephemeral Agreement Key Pair (dE, QE)	These keys are generated in two cases: (i) to establish a HSM-to-HSM encryption key to transport DKEKs in domain tokens; (ii) to establish HSM-to-operator session keys to protect sensitive HSM-operator communications.	Elliptic curve Diffie-Hellman keys on the curve secp384r1 (NIST-P384)	Internally using DRBG	**Input:** N/A **Output:** The public key (QE) is exported in plaintext	Volatile memory only
HSM Session Key Encryption Key (HSKEK)	Encrypts the HSM-Operator Session Key (HOSK) for the following operations: Initialize, InitializeAndCreateDomain, Attest, GetAttestationIdentity, and Wipe.	AES GCM 256 bits	Internally using DRBG	**Input:** N/A **Output:** N/A	Volatile memory only

Keys and CSPs	Key Description	Algorithm and Key Size	Generation	Input / Output Method	Storage
HSM Signature Key Pair (dHSK, QHSK)	Every initiated HSM has a locally generated Elliptic Curve Signature key pair used to sign data created on the HSM.	Elliptic Curve Signature key pair on the curve secp384r1 (NIST-P384)	Internally using DRBG	**Input:** N/A **Output:** The public key (QHSK) is exported in plaintext	Volatile memory only
HSM-Operator Session Key (HOSK)	Operator services establish an AES-256-GCM session key with the HSM to protect communication between operator services and HSMs in the same domain.	AES GCM 256 bits.	Internally using DRBG, or imported from an HSM that is a member of the same domain	**Input:** The HOSK is encrypted with the domain key (DK). **Output:** The HOSK is encrypted in two forms. The first form is encrypted with either the domain key (DK) or the HSM Session Key Encryption Key (HSKEK) using AES GCM. 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 (dE,QE) and the Operator Ephemeral Agreement Public Key (dOEAK, QOEAK).	Volatile memory only
Import Wrapping Key (dIWK, QIWK)	The public key is used by customers of KMS to wrap their CSK for import via the public AWS KMS API.	RSA 2048, 3072, and 4096 bits	Internally using DRBG or imported from another member of a Domain	**Input:** The private key (dIWK) is encrypted with the Domain Key (DK) using AES-GCM. **Output:** the private key (dIWK) is encrypted with the Domain Key (DK) using AES-GCM. The public key (QIWK) is exported in plaintext.	Volatile memory only
Import Wrapping Envelope Key (IWEK)	Key generated by a customer of KMS outside the AWS KMS system. Ephemeral key-wrapping-key used to encrypt CSKs for the ImportKey API when AES-KWP is used per SP 800-56B.	AES KWP 256 bits	Externally by AWS KMS customers	**Input:** IWEK is encrypted using Import Wrapping Key (QIWK) when used with the ImportKey API when the customer imports a CSK into the AWS KMS system. **Output:** N/A	Volatile memory only

Keys and CSPs	Key Description	Algorithm and Key Size	Generation	Input / Output Method	Storage
Customer Supplied Key (CSK)	Key generated by a customer of KMS outside the AWS KMS system. Customer Supplied Keys (CSK) can be of two different forms – symmetric and asymmetric keys. The first form is a 256-bit master key, from which specific-use DEKs may be derived using the SP800-108 CTR key derivation function. The CSK or DEKs derived from the CSK are used to encrypt CDKs. The second form is an asymmetric private key. The CSK maps to the Customer Master Key construct exposed in the public AWS KMS API.	AES GCM: 256 bits RSA: 2048, 3072, or 4096 bits ECDSA: curves P-256, P-384, P-521, or secp256k1.	Externally by AWS KMS customers	**Input:** CSK is encrypted using Import Wrapping Key (QIWK) (and, optionally, the ephemeral ImportWrapping 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. **Output:** CSK encrypted by a Domain Key (DK).	Volatile memory only
DRBG (CTR AES)	Entropy input (length dependent on security strength)	SP 800-90A CTR DRBG V (128 bits) AES key (256)	Internally by ENT	**Input:** Directly from the internal ENT **Output:** N/A	Volatile memory only
Replication Signing Key Pair ($dRSK_n$, $QRSK_n$)	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.	ECDSA on curve P-384	Internally using DRBG, or imported from another member of a Domain	**Input:** $dRSK_n$ encrypted with the DKEK may be imported to other members of a Domain; $QRSK_n$ may be imported by an operator **Output:** $dRKS_n$ encrypted with the DKEK may be exported to other members of a Domain; $QRSK_n$ may be exported in plaintext	Volatile memory only
Replication Agreement Key ($dRAK_k$, $QRAK_k$)	Key used for key agreement to derive a Replication Wrapping Key (RWK)	ECDH on curve P-384	Internally using DRBG, or imported from a member of a different Domain	**Input:** $QRAK_k$ may be obtained in plaintext from another HSM; $dRAK_k$ may be obtained encrypted with the domain key (DK_n) from another HSM **Output:** $QRAK_k$ may be exported in plaintext; $dRAK_k$ may be exported encrypted with the domain key (DK_n)	Volatile memory only

Keys and CSPs	Key Description	Algorithm and Key Size	Generation	Input / Output Method	Storage
Replication Wrapping Key (RWK)	Key used to encrypt an HBK	AES GCM: 256 bits	Internally derived from a Public Replication Agreement Key ($QRAK_1$) and a Private Replication Agreement Key ($dRAK_2$)	**Input:** N/A **Output:** N/A	Volatile memory only
Customer Replication Key (CRK)	Customer key that is being transmitted between two HSMs. Customer Replication Keys (CRK) may be of two types: symmetric or asymmetric. CRKs are encrypted with Replication Wrapping Keys (RWK).	AES GCM: 256 bits RSA: 2048, 3072, or 4096 bits ECDSA: curves P-256, P-384, P-521, or secp256k1	Internally from an HBK encrypted with a domain key (DK_n)	**Input:** CRK may be obtained by decrypting an HBK using a domain key (DK_n) and re-encrypting it using a Replication Wrapping Key (RWK) **Output:** CRK is exported encrypted with a Replication Wrapping Key (RAK)	Volatile memory only
Customer Data Envelope Key (CDEnK)	Ephemeral key used to encrypt plaintext data	AES CBC and GCM: 256 bits	Internally using DRBG	**Input:** N/A **Output:** CDEnK is encrypted using Customer Data Encryption Public Key (QCDEK) when used with the Generate,GenerateAndEncryptRandomBytes, GenerateDataKeyPair, and Decrypt APIs.	Volatile memory only

Table 13 – Module Keys/CSPs

2.6.2 Public Keys

Table 14 shows the list of Public Keys used within the module with associated private keys that only exist outside of the module. All Public Keys are generated outside of the module.

Public Key	Key Description	Algorithm and Key Size	Input / Output Method	Storage
Operator Ephemeral Agreement Public Key (QOEAK)	Operators establish a session key (HSM-Operator Session Keys) using an Elliptic Curve Diffie-Hellman key exchange on the curve secp384r1 (NIST-P384).	Elliptic Curve Diffie-Hellman (EC DH) ephemeral key agreement on the curve secp384r1 (NIST-P384)	**Input:** When an operator calls the NegotiateSessionKey service. **Output:** N/A	Volatile memory

Public Key	Key Description	Algorithm and Key Size	Input / Output Method	Storage
Operator Signature Public Key (QOS)	Both service operators and human operators have an identity signing key used to authenticate to the HSM.	Elliptic Curve Signature (EC DSA) on the curve secp384r1 (NIST-P384), RSA 2048, or RSA 4096 bits	**Input:** The public key (QOS) is imported in plaintext when an administrator calls InitializeAndCreateDomain, CreateDomain, and ChangeDomain. They are also imported by APIs that accepts a Domain Token. **Output:** The public keys are exported from the HSM in plaintext by APIs that exports a Domain Token.	Volatile memory
Customer Data Encryption Public Key (QCDEK)	A public key provided by an operator or customer for the module to encrypt plaintext data.	RSA 2048, 3072, or 4096 bits	**Input:** The public key (QCDEK) is optionally provided when an operator calls Generate,GenerateAndEncryptRandomBytes, GenerateDataKeyPair, and Decrypt. **Output:** N/A	Volatile memory

Table 14 – Public Keys

2.7 Self-Tests

FIPS 140-2 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 CSPs are zeroized and the module becomes unusable.

2.7.1 Power-On Self-Tests

Power-on 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 power-on self-tests:

Type	Test
Integrity Check	• 256 bit error detection code (EDC) on all module components
Known Answer Tests	• AES (Encryption and decryption in ECB mode. Key size: 128 bit) • AES GCM / GMAC (Generation and verification. Key size: 128 bit) • ECC KAS (ECDH) (Primitive Z test. Parameter set: EC P-256) • ECDSA (signature generation and verification. Curve: P-256) • RSA (Signature generation and verification, key transport SP800-56B per IG D.9. Key size: 2048 bits) • HMAC (Generation and verification with SHA-256, SHA-512) • SHS (SHA-1, SHA-256, SHA-512) • SP 800-90 CTR_DRBG • SP 800-108 CTR KDF (HMAC-SHA-256) • KDA (OneStep KDF) (SHA-256)
Other tests	• Vendor defined health tests, per SP 800-90B Section 4.5 (run over 65,536 consecutive samples)

Table 15 – Power-On Self-Tests

Each module performs all power-on self-tests automatically when the module is initialized. All power-on self-tests must be passed before a User/Crypto Officer can perform services. The Power-on self-tests can be run on demand by rebooting the module in FIPS-Approved Mode of Operation.

With the exception of the Integrity Check, the failure of any power-on self-test will cause the module to an error state where the module will restart. If the Integrity Check fails, the module will enter into an alternate error state where the module will power off.

2.7.2 Conditional Self-Tests

Conditional self-tests are tests that run during operation of the module. 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 the FIPS-Approved mode of operation. Each module performs the following conditional self-tests:

Type	Description
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
Continuous RNG Tests	• Performed on output of entropy source
DRBG Health Tests	• Performed on DRBG, per SP 800-90A Section 11.3.
SP 800-90B Health Tests	• Vendor defined health tests

Table 16 – Conditional Self-Tests

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

2.7.3 Critical Function Tests

Known answer tests as described in Table 15 are automatically executed by the module every 24 hours. If any of the tests fail, the module will enter into an error state and no services can be accessed by the operator.

2.7.4 On-Demand Self-Tests

On-demand self-tests can be performed by rebooting the module which will perform the power-on self-tests as described in Section 2.7.1.

2.8 Mitigation of Other Attacks

The module does not mitigate other attacks.

3 Guidance and Secure Operation

The module only supports FIPS-mode of operation. Beyond initial setup, no specific technical steps are required to configure FIPS-mode of operation.

3.1 Crypto Officer Guidance

Only authorized AWS employees may assume the Administrator (Crypto Officer) role.

The following section provides a high-level overview to configure the HSM. Members of the Administrator role (Crypto Officer) must follow the AWS internal guidance published in the Operation Guidance for the AWS Key Management Service.

3.1.1 Module Inspection

The module must be inspected to verify that no attempts have been made to open the module. The module must be inspected upon initial delivery and after the module reboots due to unscheduled/unexpected power events.

If evidence of a tamper is discovered, the module shall be removed from operation immediately.

3.1.2 Initial Configuration

When setting up the first HSM member of a new domain, call the InitializeAndCreateDomain service with the list of operators' Operator Signature Public Keys, their respective roles and the access policy for each service (API) in accordance with AWS internal guidance and procedures.

Services / APIs that modify an HSM's domain membership or configuration must be configured to require a quorum of two Crypto Officers except when adding an HSM that was previously a member of the domain.

When setting up subsequent members of an existing domain, the administrator first retrieves domain information from an existing domain member using the GetDomain service. The Initialize service can then be used to initialize the new HSM with the configuration of the existing domain.

Ensure each HSM is operating in FIPS mode by calling the Fips status API.

3.2 User Guidance

3.2.1 General Guidance

No additional guidance is required to maintain FIPS mode of operation. The only users of the HSM are the front-end hosts of the AWS Key Management Service.