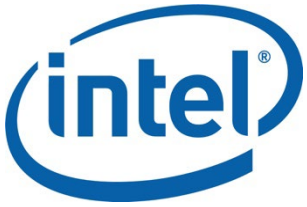

Intel Corporation

Crypto Module for Intel® Alder Point PCH Converged Security and Manageability Engine (CSME)

FIPS 140-3 Non-Proprietary Security Policy

Hardware Version: 4.6.0.0

Firmware Version: 5.2.0.0

Prepared by:

atsec information security corporation

4516 Seton Center Parkway, Suite 250

Austin, TX 78759

www.atsec.com

Table of Contents

1 General 6
1.1 Overview 6
1.2 Security Levels 6
2 Cryptographic Module Specification 7
2.1 Description 7
2.2 Tested and Vendor Affirmed Module Version and Identification 8
2.3 Excluded Components 9
2.4 Modes of Operation 9
2.5 Algorithms 9
2.6 Security Function Implementations 13
2.7 Algorithm Specific Information 16
2.8 RBG and Entropy 16
2.9 Key Generation 16
2.10 Key Establishment 17
2.11 Industry Protocols 17
3 Cryptographic Module Interfaces 18
3.1 Ports and Interfaces 18
4 Roles, Services, and Authentication 19
4.1 Authentication Methods 19
4.2 Roles 19
4.3 Approved Services 20
4.4 Non-Approved Services 24
4.5 External Software/Firmware Loaded 25
5 Software/Firmware Security 26
5.1 Integrity Techniques 26
5.2 Initiate on Demand 26
5.3 Additional Information 26
6 Operational Environment 27
6.1 Operational Environment Type and Requirements 27
7 Physical Security 28
7.1 Mechanisms and Actions Required 28
8 Non-Invasive Security 29
9 Sensitive Security Parameters Management 30

9.1 Storage Areas 30
9.2 SSP Input-Output Methods 30
9.3 SSP Zeroization Methods 30
9.4 SSPs 31
9.5 Transitions....34
10 Self-Tests....35
10.1 Pre-Operational Self-Tests....35
10.2 Conditional Self-Tests 35
10.3 Periodic Self-Test Information....37
10.4 Error States 38
11 Life-Cycle Assurance....39
11.1 Installation, Initialization, and Startup Procedures 39
11.2 Administrator Guidance....39
11.3 Non-Administrator Guidance....39
11.4 Design and Rules....40
11.5 Maintenance Requirements 40
11.6 End of Life 40
12 Mitigation of Other Attacks....41
12.1 Attack List....41

List of Tables

Table 1: Security Levels 6
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 8
Table 3: Tested Module Identification – Hybrid Disjoint Hardware 8
Table 4: Tested Operational Environments - Software, Firmware, Hybrid 8
Table 5: Modes List and Description 9
Table 6: Approved Algorithms 12
Table 7: Vendor-Affirmed Algorithms 12
Table 8: Non-Approved, Not Allowed Algorithms 13
Table 9: Security Function Implementations 16
Table 10: Entropy Sources 16
Table 11: Ports and Interfaces 18
Table 12: Authentication Methods 19
Table 13: Roles 19
Table 14: Approved Services 24
Table 15: Non-Approved Services 25
Table 16: Mechanisms and Actions Required 28
Table 17: Storage Areas 30
Table 18: SSP Input-Output Methods 30
Table 19: SSP Zeroization Methods 30
Table 20: SSP Table 1 32
Table 21: SSP Table 2 34
Table 22: Pre-Operational Self-Tests 35
Table 23: Conditional Self-Tests 37
Table 24: Pre-Operational Periodic Information 37
Table 25: Conditional Periodic Information 38
Table 26: Error States 38

List of Figures

Figure 1: Block Diagram 7
Figure 2: Alder Point PCH-S with Alder Lake-S....9
Figure 3: Alder Point PCH-M/P with Alder Lake M 9
Figure 4: Alder Point PCH-S with Raptor Lake S / Raptor Lake HX....9
Figure 5: Alder Point PCH-M/P with Raptor Lake P 9

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy of the Crypto Module for Intel® Alder Point PCH Converged Security and Manageability Engine (CSME). It has a one-to-one mapping to the [SP 800-140Br1] starting with section B.2.1 named “General” that maps to section 1 in this document and ending with section B.2.12 named “Mitigation of other attacks” that maps to section 12 in this document. This document also contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for a Security Level 2 module.

1.2 Security Levels

The table below shows the security level claimed for each of the security requirement area that comprise the FIPS 140-3 standard:

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

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The Crypto Module for Intel® Alder Point PCH Converged Security and Manageability Engine (CSME) is classified as a Hybrid Firmware module operating in a single-chip environment. In this document, “CSME”, and “module” are used interchangeably. They all refer to the Crypto Module for Intel® Alder Point PCH Converged Security and Manageability Engine (CSME).

The module consists of both hardware (AES, ECC, and HCU hardware cryptographic engines) and firmware providing interface to the engines.

Module Type: Firmware-hybrid
Module Embodiment: Single Chip

Cryptographic Boundary:

The module is a firmware hybrid module implemented in the physical embodiment of either a Alder Point PCH-S with Alder Lake S (referred to as ADL-S), Raptor Lake S (RPL-S) or Raptor Lake HX (RPL-HX) CPU or an Alder Point PCH-M/P with Alder Lake M (ADL-M) or Raptor Lake P (RPL-P) CPU. The Tested Operational Environment’s Physical Perimeter (TOEPP) is represented by the dashed purple lines in the block diagram shown below.

The module provides cryptographic services to operators through an application program interface (API). The cryptographic boundary consists of the CSME ROM, CSME Crypto Driver, the AES, ECC, and HCU hardware cryptographic engines along with the fips_hmac integrity file. The cryptographic boundary is represented by the dashed red lines in the block diagram shown below.

Tested Operational Environment’s Physical Perimeter (TOEPP):

The Tested Operational Environment’s Physical Perimeter (TOEPP) is represented by the dashed purple lines below.

Figure 1: Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
Intel(R) Converged Security and Manageability Engine Driver	5.2.0.0	N/A	HMAC-SHA2-256

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

Tested Module Identification – Hybrid Disjoint Hardware:

Model and/or Part Number	Hardware Version	Firmware Version	Processors	Features
Offload and Cryptography Subsystem with CSME ROM	4.6.0.0	N/A	Alder Lake S, Alder Lake M, Raptor Lake S, Raptor Lake HX, Raptor Lake P	N/A

Table 3: Tested Module Identification – Hybrid Disjoint Hardware

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
CSME OS running firmware version 16.1.25.2124	Alder Point PCH-S	Alder Lake S (ADL-S)	No	N/A	5.2.0.0
CSME OS running firmware version 16.1.25.2124	Alder Point PCH-M/P	Alder Lake M (ADL-M)	No	N/A	5.2.0.0
CSME OS running firmware version 16.1.25.2124	Alder Point PCH-S	Raptor Lake S (RPL-S)	No	N/A	5.2.0.0
CSME OS running firmware version 16.1.25.2124	Alder Point PCH-S	Raptor Lake HX (RPL-P)	No	N/A	5.2.0.0
CSME OS running firmware version 16.1.25.2124	Alder Point PCH-M/P	Raptor Lake P (RPL-HX)	No	N/A	5.2.0.0

Table 4: Tested Operational Environments - Software, Firmware, Hybrid

Figure 2: Alder Point PCH-S with Alder Lake-S

Figure 3: Alder Point PCH-M/P with Alder Lake M

Figure 4: Alder Point PCH-S with Raptor Lake S / Raptor Lake HX[1]

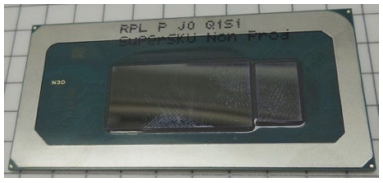

Figure 5: Alder Point PCH-M/P with Raptor Lake P

2.3 Excluded Components

There are no components within the cryptographic boundary excluded from the FIPS 140-3 requirements.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved mode	only approved or allowed security functions with sufficient security strength can be used	Approved	fips_indicator=FIPS_APPROVED_SEC_FUN
Non-approved mode	only non-approved security functions can be used	Non-Approved	fips_indicator=NULL

Table 5: Modes List and Description

Mode Change Instructions and Status:

The mode of operation is implicitly assumed contingent on the service that is being requested. In other words, if an approved service is requested, the module assumes the approved mode of operation (the FIPS mode); if a non-approved service is requested, the module assumes the non-approved mode of operation.

2.5 Algorithms

Approved Algorithms:

The module supports the following Approved cryptographic algorithms. Table 4 lists the algorithms validated by the CAVP Certificate:

[1] RPL-HX is the same exact HW as RPL-S but using a LGA socket.

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A3362	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38A
AES-CFB128	A3362	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38A
AES-CMAC	A3362	Direction - Generation, Verification Key Length - 128, 256 MAC Length - MAC Length: 128 Message Length - Message Length: 0-16384 Increment 8	SP 800-38B
AES-CTR	A3362	Direction - Decrypt, Encrypt Key Length - 128, 256 Payload Length - Payload Length: 128 Supports Counter larger than maximum value - No Incremental Counter - Yes Counter Tests Performed - Yes	SP 800-38A
AES-ECB	A3362	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38A
AES-GCM	A3362	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.2 Key Length - 128, 256 Tag Length - 104, 112, 120, 128, 96 IV Length - IV Length: 96 Payload Length - Payload Length: 128, 256, 8, 112 AAD Length - AAD Length: 128, 256, 16, 64	SP 800-38D
AES-OFB	A3362	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38A
Counter DRBG	A3362	Prediction Resistance - No Supports Reseed - No Mode - AES-256 Derivation Function Enabled - Yes Additional Input - Additional Input: 0 Entropy Input - Entropy Input: 256 Nonce - Nonce: 128 Personalization String Length - Personalization String Length: 0 Returned Bits - 128	SP 800-90A Rev. 1
ECDSA KeyGen (FIPS186-4)	A3362	Curve - P-256, P-384 Secret Generation Mode - Testing Candidates	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3362	Curve - P-256, P-384	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A3362	Component - No Curve - P-256, P-384 Hash Algorithm - SHA2-256, SHA2-384	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
ECDSA SigVer (FIPS186-4)	A3362	Component - No Curve - P-256, P-384 Hash Algorithm - SHA2-256, SHA2-384	FIPS 186-4
HMAC-SHA-1	A3362	MAC - MAC: 32, 48, 64 Key Length - Key Length: 8-4096 Increment 8	FIPS 198-1
HMAC-SHA2-224	A3362	MAC - MAC: 32, 48, 64 Key Length - Key Length: 8-4096 Increment 8	FIPS 198-1
HMAC-SHA2-256	A3362	MAC - MAC: 32, 48, 64 Key Length - Key Length: 8-4096 Increment 8	FIPS 198-1
HMAC-SHA2-384	A3362	MAC - MAC: 32, 48, 64 Key Length - Key Length: 8-4096 Increment 8	FIPS 198-1
HMAC-SHA2-512	A3362	MAC - MAC: 32, 48, 64 Key Length - Key Length: 8-4096 Increment 8	FIPS 198-1
KAS-ECC Sp800-56Ar3	A3362	Domain Parameter Generation Methods - P-256, P-384 Function - Full Validation, Key Pair Generation iutId - 1234567890abcdef Scheme - ephemeralUnified - KAS Role - Initiator, Responder KDF Methods - oneStepKdf - Auxiliary Function Methods - Auxiliary Function Name - SHA2-224 MAC Salting Methods - default Fixed Info Pattern - uPartyInfo||vPartyInfo||literal[0123456789abcdef] Fixed Info Encoding - Concatenation Key Length - 1024	SP 800-56A Rev. 3
KDF SP800-108	A3362	KDF Mode - Counter MAC Mode - HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512 Supported Lengths - Supported Lengths: 8-256 Increment 8 Fixed Data Order - Before Fixed Data Counter Length - 32 Supports Empty IV - No Requires Empty IV - No Custom Key In Length - 0	SP 800-108 Rev. 1
KTS-IFC	A3362	Function - keyPairGen IUT ID - 0123456789abcdef Modulo - 2048, 3072, 4096 Key Generation Methods - rsakpg1-basic Fixed Public Exponent - 010001 Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Hash Algorithms - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Supports Null Associated Data - No Associated Data Pattern - uPartyInfo||vPartyInfo	SP 800-56B Rev. 2

Algorithm	CAVP Cert	Properties	Reference
		Associated Data Encoding - concatenation Key Length - 256	
RSA Decryption Primitive (CVL)	A3362	Modulus Length - 2048	FIPS 186-4
RSA KeyGen (FIPS186-4)	A3362	Key Generation Mode - B.3.3 Modulo - 2048 Primality Tests - Table C.2 Info Generated By Server - Yes Public Exponent Mode - Fixed Fixed Public Exponent - 010001 Private Key Format - Standard	FIPS 186-4
RSA SigGen (FIPS186-4)	A3362	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-224	FIPS 186-4
RSA Signature Primitive (CVL)	A3362	Private Key Format - standard Public Exponent Mode - fixed Fixed Public Exponent - 010001	FIPS 186-4
RSA SigVer (FIPS186-4)	A3362	Signature Type - PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA-1 Public Exponent Mode - Fixed Fixed Public Exponent - 010001	FIPS 186-4
SHA-1	A3362	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4
SHA2-224	A3362	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4
SHA2-256	A3362	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4
SHA2-384	A3362	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4
SHA2-512	A3362	Message Length - Message Length: 8-65536 Increment 8	FIPS 180-4

Table 6: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
Cryptographic Key Generation (CKG)	Key Type:asymmetric	N/A	Key Generation Compliant with SP 800-133Rev2 section 4 example 1

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

N/A for this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

Non-Approved, Not Allowed Algorithms:

The module implements the following non-Approved algorithms **Not** Allowed in the Approved Mode of Operation listed in the table below.

Name	Use and Function
MD5	Message Digest generation using MD5 algorithm
SM2	SM2 digital signature generation/verification
SM3	SM3 message digest generation
SM4	SM4 data encryption and decryption
HMAC-MD5	HMAC-MD5 MAC generation
ECC Commit Computation	ECC commit computation used in ECSCHNORR and ECDAA Signature Generation
ECDAA	ECDAA digital signature generation/verification
ECSCHNORR	ECSCHNORR digital signature generation/verification
AES-GCM	AES-GCM data encryption using an externally generated IV
HMAC	HMAC MAC generation using with keys less than 112 bits
ECDSA	ECDSA key-pair generation using secp256k1 curve
ECDHE Shared Secret Computation	ECDHE shared secret computation using brainpoolp384r1 curve
RSA	RSA key-pair generation using 1024 bits modulus size; RSA digital signature generation using MD5, SM3 or SHA-1 hash algorithm; RSA digital signature verification using MD5 or SM3 hash algorithm; RSA key encapsulation and decapsulation using MD5 or SHA-1; RSA key encapsulation and decapsulation using 1024-bit modulus; RSA key decapsulation using 3072 or 4096-bit modulus

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
AES-CBC	BC-UnAuth	AES Encryption / AES Decryption		AES-CBC: (A3362)
AES-CFB128	BC-UnAuth	AES Encryption / AES Decryption		AES-CFB128: (A3362)
AES-CMAC	MAC	AES Message Authentication Code		AES-CMAC: (A3362)

Name	Type	Description	Properties	Algorithms
		Generation and Verification		
AES-CTR	BC-UnAuth	AES Encryption / AES Decryption		AES-CTR: (A3362)
AES-ECB	BC-UnAuth	AES Encryption / AES Decryption		AES-ECB: (A3362)
AES-GCM	BC-Auth	AES Encryption / AES Decryption	IV Generation:Internal (Mode 8.2.2)	AES-GCM: (A3362)
AES-OFB	BC-UnAuth	AES Encryption / AES Decryption		AES-OFB: (A3362)
Counter DRBG	DRBG	Random Number Generation		Counter DRBG: (A3362)
ECDSA KeyGen	AsymKeyPair-KeyGen	ECDSA key-pair Generation		ECDSA KeyGen (FIPS186-4): (A3362)
ECDSA KeyVer	AsymKeyPair-KeyVer	ECDSA public-key verification		ECDSA KeyVer (FIPS186-4): (A3362)
ECDSA SigGen	DigSig-SigGen	ECDSA digital signature generation		ECDSA SigGen (FIPS186-4): (A3362)
ECDSA SigVer	DigSig-SigVer	ECDSA digital signature verification		ECDSA SigVer (FIPS186-4): (A3362)
HMAC-SHA-1	MAC	HMAC Message Authentication Code Generation		HMAC-SHA-1: (A3362)
HMAC-SHA2-224	MAC	HMAC Message Authentication Code Generation		HMAC-SHA2-224: (A3362)
HMAC-SHA2-256	MAC	HMAC Message Authentication Code Generation		HMAC-SHA2-256: (A3362)
HMAC-SHA2-384	MAC	HMAC Message Authentication Code Generation		HMAC-SHA2-384: (A3362)
HMAC-SHA2-512	MAC	HMAC Message Authentication Code Generation		HMAC-SHA2-512: (A3362)
KAS-ECC	KAS-Full	ECDH key agreement	Compliance:IG D.F Scenario 2 (path 2) Caveat:Key establishment methodology providing between 128 and 192	KAS-ECC Sp800-56Ar3: (A3362)

Name	Type	Description	Properties	Algorithms
			bits of key strength Key confirmation:No Key Derivation:One-Step KDF using HMAC-SHA-384	
KBKDF	KBKDF	Key Based Key Derivation		KDF SP800-108: (A3362)
RSA encapsulation	KTS-Encap	RSA encapsulation using KTS-IFC	Scheme:KTS-OAEP-basic Modulus Size:2048, 3072, 4096 Standard:SP800-56Brev2 Caveat:Key encapsulation methodology providing between 112 to 150 bits of key strength	KTS-IFC: (A3362)
RSA decapsulation	KTS-Decap	RSA decapsulation using KTS-IFC	Scheme:KTS-OAEP-basic Modulus Size:2048, 3072, 4096 Standard:SP800-56Brev2 Caveat:Key decapsulation methodology providing 112 bits of key strength	KTS-IFC: (A3362)
RSA Decryption Primitive (CVL)	KTS-Decap	RSA decapsulation primitive decryption	Scheme:Basic Modulus Size:2048 Standard:SP800-56Brev2 Caveat:Key decapsulation methodology providing 112 bits of key strength	RSA Decryption Primitive: (A3362)
RSA KeyGen	AsymKeyPair-KeyGen CKG	RSA key-pair generation		RSA KeyGen (FIPS186-4): (A3362) Cryptographic Key Generation (CKG): ()
RSA SigGen	DigSig-SigGen	RSA digital signature generation		RSA SigGen (FIPS186-4): (A3362)
RSA SigVer	DigSig-SigVer	RSA digital signature verification		RSA SigVer (FIPS186-4): (A3362)
RSA SigVer (Legacy)	DigSig-SigVer	RSA digital signature verification for legacy use	Compliance:FIPS 140-3 IG C.M Legacy Algorithms Caveat:When using 1024-bit modulo or	RSA SigVer (FIPS186-4): (A3362)

Name	Type	Description	Properties	Algorithms
			SHA-1 hash for legacy signatures	
RSA Signature Primitive (CVL)	DigSig-SigGen	RSA digital signature generation primitive		RSA Signature Primitive: (A3362)
SHS	SHA	Message Digest		SHA-1: (A3362) SHA2-224: (A3362) SHA2-256: (A3362) SHA2-384: (A3362) SHA2-512: (A3362)

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

AES- GCM IV is constructed in accordance with SP800-38D in compliance with IG C.H scenario 2. GCM IV generation uses an approved CTR_DRBG that is internal to the module's boundary and the IV length is at least 96 bits (per SP 800-38D).

The module does not establish SSPs using an approved key transport scheme (KTS). However, it does offer approved authenticated algorithms that can be used by an external operator/application as part of an approved KTS. See section 2.10 Key Establishment.

Algorithms designated as "Legacy" can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M.

2.8 RBG and Entropy

N/A for this module.

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Intel DRNG 4 Entropy Source	Physical	FC DRNG MTL SoC-M Step C0 - Intel Core Ultra Series 1	128	128-bits per 128-bits	Hardware entropy source with CBC-MAC conditioning component (Cert. #A5254) located within the tested execution environment's physical perimeter.

Table 10: Entropy Sources

The module provides a SP800-90Arev1 compliant CTR_DRBG (Cert. #A3362) with AES-256 with derivation function and without prediction resistance as the approved Random Number Generator. The CTR_DRBG is implemented in the firmware (i.e., CSME Crypto Driver) and provides between 128 and 65536 bits of output data per each request.

The module uses the output of the SP 800-90B and IG D.J compliant ENT (P) as the entropy source for seeding the CTR_DRBG inside the module. The entropy source provides 0.5 bits of entropy per bit. The entropy contained in the entropy input string to the DRBG is 256 bits.

2.9 Key Generation

The module implements asymmetric key generation services compliant to SP800-133rev2 Cryptographic Key Generation (CKG, vendor affirmed). The key generation methods are specified in the *Security Function Implementations* table.

The module does not offer a dedicated service for generating symmetric keys.

2.10 Key Establishment

The module provides an approved [SP800-56Arev3] EC Diffie-Hellman Key Agreement Scheme. The key agreement scheme is compliant with IG D.F scenario 2 path (2). The CAVP testing was performed end-to-end, using the Ephemeral Unified Model with approved domain parameters (i.e., P-256 and P-384 curves and SHA-224, SHA-256, SHA-384, and SHA-512 auxiliary function) resulting in a KAS-ECC Cert. #A3362.

The module supports SP800-56Brev2 Key Transport using KTS-OAEP-basic. The module does not establish any SSPs for itself. Instead, the module provides this functionality as a service. RSA-KTS encapsulation is approved with the key sizes of 2048, 3072 and 4096 bits in approved mode while RSA-KTS un-encapsulation is approved only with a key size of 2048 bits in the approved mode. Guidance meeting the SP800-56Brev2 assurances can be found in section 11.3 Non-Administrator Guidance.

2.11 Industry Protocols

This section is not applicable to this module.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

The cryptographic module is defined as a Firmware-Hybrid module. The logical interfaces are the application program interface (API) through which operators request services from the module (i.e., CSME Crypto Driver). The module does not implement a control output interface. The following table summarizes the four logical interfaces:

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	All data (except control data entered via the control input interface) that is input to and processed by a cryptographic module
N/A	Data Output	All data (except status data output via the status output interface and control data output via the control output interface) that is output from a cryptographic module
N/A	Control Input	Control data used to control the operation of a cryptographic module
N/A	Status Output	All status data used to indicate the status of a cryptographic module
Provided by the underlying SoC/PCH	Power	External electrical power that is input to a cryptographic module

Table 11: Ports and Interfaces

4 Roles, Services, and Authentication

4.1 Authentication Methods

Method Name	Description	Security Mechanism	Strength Each Attempt	Strength per Minute
User Authentication	RSASSA-PSS signature algorithm with 3072-bit modulus	RSA SigVer (FIPS186-4) (A3362)	1/2^128	Less than or equal to 60,000,000 * 1/2^128

Table 12: Authentication Methods

The module implements role-based operator authentication to authenticate the User Role. The authentication mechanism is based on the RSASSA-PSS signature algorithm with 3072-bit modulus (Cert. #A3362). The Crypto Officer role is not authenticated. The Crypto Officer role can only perform the initialization service, which is a non-authenticated service and does not affect the security of the module per IG 4.1.A.

An operator with the User role is the CSME firmware application (Figure 1). There can be only one CSME application running and interfacing with the module during the module's operation, enforced by the design of the module. Thus, the module does not support concurrent operators.

The manufacturer utilizes their unique RSA private key to sign the CSME application images. The private signing keys are kept in a HSM (Hardware Security Module) within an Intel secured facility and remain unknown to both the module and the CSME application. The public key and the signature are provided as part of the firmware manifest. The hash of the public key is also hardcoded in the module. During runtime, the public key in the provided firmware manifest is first hashed, then compared with the hash stored in module. If the hash matches, the module proceeds to assert whether the signature of the CSME application verifies, using the public key. If the signature verification succeeds, the CSME application firmware is authenticated and hence can be loaded and executed. The module does not maintain authentication after computing platform power loss (power-off, reset, etc.).

The authentication process occurs within the physical perimeter of the module, and thus it is not visible outside of this perimeter. The authentication data is essentially composed of public data (signature and public key); therefore, their disclosure does not affect the security of the authentication mechanism.

Strength of Authentication

The digital signature verification authentication mechanism using RSA PSS with 3072-bit modulus provides an encryption strength of 128 bits. The strength of this mechanism is equivalent to the probability of correctly guessing the private signing key, and this probability is $1/2^{128}$ (or 2.94e-39).

If attempts are made to authenticate an operator by guessing the private key and presenting the corresponding signature of the CSME firmware, we may suppose a rate of 1µs per attempted authentication (i.e., per guess of the private key and respective signature). This rate would allow 60,000,000 consecutive attempts per minute. The probability of successfully authenticating at this rate is less than or equal to 60,000,000 * $1/2^{128}$ (≤1.7632e-31).

4.2 Roles

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

Table 13: Roles

The module does not support a maintenance role The User and Crypto Officer roles are assumed by the entity accessing the module services contingent on the authentication, as described in Section 4.3. The Crypto Officer and User Roles are separated by function. The Crypto Officer is only available during initialization of the module as instructed in Section 11.1. After the module is operational, only the User Role is available to request services of the module.

4.3 Approved Services

The module provides services to the operator that assumes one of the authorized roles, User role after operator is authenticated or CO role which is not authenticated but can only perform initialization service. All services are described in detail in the user documentation. For Approved services, a fips_indicator flag is enabled in the crypto_ioctl_status_t structure and is set to FIPS_APPROVED_SEC_FUN when a function is an approved service. After the module completes the requested service, it will report the status as an output parameter indicating whether the service was Approved (i.e., set to “1”).

The following table lists the Approved services and the non-Approved but allowed services in approved mode of operation, the roles that can perform the service, the Critical Security Parameters involved and how they are accessed:

The access rights to keys and SSPs have the following interpretation:

G = **Generate**: The module generates or derives the SSP.

R = **Read**: The SSP is read from the module (e.g., the SSP is output).

W = **Write**: The SSP is updated, imported, or written to the module.

E = **Execute**: The module uses the SSP in performing a cryptographic operation.

Z = **Zeroise**: The module zeroises the SSP.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
AES data encryption	Data Encryption using Advanced Encryption Standard algorithm	1	Key and Plaintext Data	Ciphertext Data	AES-CBC AES-CFB128 AES-CTR AES-ECB AES-GCM AES-OFB	User Role - AES keys: W,E,Z
AES data decryption	Data Decryption using Advanced Encryption Standard algorithm	1	Key and Ciphertext Data	Plaintext Data	AES-CBC AES-CFB128 AES-CTR AES-ECB AES-GCM AES-OFB	User Role - AES keys: W,E,Z
AES message authentication code generation	Message Authentication Code Generation using Advanced Encryption Standard algorithm	1	Key and Message	Message Authentication Code	AES-CMAC	User Role - AES keys: W,E,Z
AES message authentication code verification	Message Authentication Code Verification using Advanced Encryption Standard algorithm	1	Key and Message Authentication Code	True or False	AES-CMAC	User Role - AES keys: W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
HMAC message authentication code generation	Message Authentication Code Generation using Hashed Message Authentication Code algorithm	1	Key and Message	Message Authentication Code	HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	User Role - HMAC keys: W,E,Z
RSA key-pair generation	Asymmetric Key Pair Generation	1	Key Size	RSA Key Pair	Counter DRBG RSA KeyGen	User Role - Module generated RSA public key: G,R - Module generated RSA private key (Including intermediate keygen values): G
RSA public key validation	Public Key Validation of RSA public key	1	RSA Public Key	True or False	None	User Role - RSA public key: W,E,Z
RSA digital signature generation	Digital Signature Generation using RSA	1	RSA Private Key and Message	Digital Signature	RSA SigGen	User Role - RSA private key: W,E,Z
RSA digital signature generation primitive	Digital Signature Generation on a pre-hashed message using RSA	1	RSA Private Key, Message and Digest	Digital Signature	RSA Signature Primitive (CVL)	User Role - RSA private key: W,E,Z
RSA digital signature verification	Digital Signature Verification using RSA	1	RSA Public Key and Digital Signature	True or False	RSA SigVer	User Role - RSA public key: W,E,Z
RSA digital signature verification (Legacy)	Digital Signature Verification using RSA for legacy use	1	RSA Public Key and Digital Signature	True or False	RSA SigVer (Legacy)	User Role - RSA public key: W,E,Z
RSA encapsulation	Encapsulation using KTS-IFC	1	RSA Public Key and Plaintext Data	Ciphertext Data	RSA encapsulation	User Role - Module generated RSA private key (Including intermediate keygen values): E,Z
RSA decapsulation	Decapsulation using KTS-IFC	1	RSA Private Key and Ciphertext Data	Plaintext Data	RSA decapsulation	User Role - RSA public key: W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
RSA Decryption Primitive (CVL)	RSA decryption primitive as defined in SP800-56BRev2 Section 7.1.2.1	1	RSA Private Key and Encrypted Key	Plaintext Key	RSA Decryption Primitive (CVL)	User Role - RSA private key: W,E,Z
ECDSA key-pair generation	Asymmetric Key Pair Generation	1	EC curve	ECDSA Key Pair	Counter DRBG ECDSA KeyGen	User Role - ECDSA/ECDH (ECC) public key (including intermediate keygen values): G,R - ECDSA/ECDH (ECC) private key (including intermediate keygen values): G,R
ECDSA public key validation	Public Key Validation of ECDSA public key	1	ECDSA Public Key	True or False	ECDSA KeyVer	User Role - ECDSA/ECDH (ECC) public key (including intermediate keygen values): W,E,Z
ECDSA digital signature generation	Digital Signature Generation using ECDSA	1	ECDSA Private Key and Message	Digital Signature	ECDSA SigGen	User Role - ECDSA/ECDH (ECC) private key (including intermediate keygen values): W,E,Z
ECDSA digital signature verification	Digital Signature Verification using ECDSA	1	ECDSA Public Key and Digital Signature	True or False	ECDSA SigVer	User Role - ECDSA/ECDH (ECC) public key (including intermediate keygen values): W,E,Z
ECDH key agreement	Diffie-Hellman Key Agreement using Elliptic Curve Cryptography	1	EC Curve, Party U's ephemeral private key, and Party V's ephemeral public key	Derived Key	KAS-ECC	User Role - ECDSA/ECDH (ECC) public key (including intermediate keygen values): W,E - ECDSA/ECDH (ECC) private key (including intermediate keygen values): W,E - Shared Secret: G

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- SP 800-56C KDF derived key: G,R,Z
SHS message digest generation	Message Digest Generation using Secure Hash Standard algorithm	1	Message	Message Digest	SHS	User Role
Random Number Generation	Deterministic Random Number Generation	1	Entropy input string and nonce	Random Numbers	Counter DRBG	User Role - Entropy Input String: W,E,Z - DRBG Seed: W,E,Z - DRBG internal state: (V and Key values): W,E,Z
Key Derivation	Derive key using KBKDF in counter mode	1	Key Derivation Key	Derived Key	KBKDF	User Role - KBKDF key derivation key: W,E,Z - KBKDF derived key: R,W,E,Z
Show Module Version	Output Module Name and Module Hardware and Firmware Version Numbers	N/A	None	Module Base Name + Module Version Number	None	Unauthenticated
Show Status	Outputs Operational/Error Status of the Module	N/A	None	Operational/Error status	None	Unauthenticated
On Demand Self-Test (Pre-operational Integrity Test and Conditional Algorithm Self-Test)	Performs On Demand Self- Test	N/A	None	Pass/Fail status	None	User Role
Zeroization	Zeroizes all CSPs	N/A	None	None	None	User Role - AES keys: Z - HMAC keys: Z - Module generated RSA public key: Z - Module generated RSA private key (Including intermediate keygen values): Z - RSA public key: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- RSA private key: Z - ECDSA/ECDH (ECC) public key (including intermediate keygen values): Z - ECDSA/ECDH (ECC) private key (including intermediate keygen values): Z - Shared Secret: Z - SP 800-56C KDF derived key: Z - KBKDF key derivation key: Z - KBKDF derived key: Z - Entropy Input String: Z - DRBG Seed: Z - DRBG internal state: (V and Key values): Z
Module Initialization	Configure BIOS to Initialize in FIPS Validated Configuration	N/A	BIOS settings	None	None	Crypto Officer Role

Table 14: Approved Services

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Data Encryption using AES-GCM with externally generated IV	Data Encryption using AES-GCM with externally generated IV	AES-GCM	User
Data Encryption and Decryption using SM4	Data Encryption and Decryption using SM4	SM4	User
Message digests using MD5 or SM3	Message digests using MD5 or SM3	MD5 SM3	User
MAC generation using HMAC-MD5	MAC generation using HMAC- MD5	HMAC-MD5	User
MAC generation using less than 112 bits HMAC key	MAC generation using less than 112 bits HMAC key	HMAC	User
Key generation using RSA with 1024 bits modulus size	Key generation using RSA with 1024 bits modulus size	RSA	User
Key generation using ECDSA using secp256k1 curve	Key generation using ECDSA using secp256k1 curve	ECDSA	User

Name	Description	Algorithms	Role
Digital signature generation using RSA with MD5, SM3 or SHA-1 for any modulus size	Digital signature generation using RSA with MD5, SM3 or SHA-1 for any modulus size	RSA	User
Digital signature verification using RSA with MD5, or SM3 for any modulus size	Digital signature verification using RSA with MD5, or SM3 for any modulus size	RSA	User
Digital signature generation using ECSCHNORR, ECDAA, or SM2 algorithm	Digital signature generation using ECSCHNORR, ECDAA, or SM2 algorithm	SM2 ECDAA ECSCHNORR	User
Digital signature verification using ECSCHNORR, ECDAA, or SM2 algorithm	Digital signature verification using ECSCHNORR, ECDAA, or SM2 algorithm	SM2 ECDAA ECSCHNORR	User
Key Transport (key encapsulation and decapsulation) using RSA-OAEP with MD5 or SHA-1	Key Transport (key encapsulation and decapsulation) using RSA-OAEP with MD5 or SHA-1	RSA	User
Key Transport (key encapsulation and decapsulation) using RSA-OAEP with 1024-bit modulus	Key Transport (key encapsulation and decapsulation) using RSA-OAEP with 1024-bit modulus	RSA	User
Key Transport (key decapsulation) using RSA-OAEP with 3072 or 4096-bit modulus	Key Transport (key decapsulation) using RSA-OAEP with 3072 or 4096-bit modulus	RSA	User
Shared secret computation using ECDHE with brainpoolp384r1 curve	Shared secret computation using ECDHE with brainpoolp384r1 curve	ECDHE Shared Secret Computation	User
ECC Commit Computation using ECSCHNORR and ECDAA (used in ECSCHNORR and ECDAA Signature Generation)	ECC Commit Computation using ECSCHNORR and ECDAA (used in ECSCHNORR and ECDAA Signature Generation)	ECC Commit Computation ECDAA ECSCHNORR	User

Table 15: Non-Approved Services

4.5 External Software/Firmware Loaded

The module loads the firmware at each boot from external memory. Prior to loading the firmware, a firmware load test is performed whereby the RSA signature over the firmware is verified. If the signature verifies successfully, the firmware is loaded and the module becomes operational. In the event the firmware load test fails, the firmware is not loaded and the module enters the Error State.

5 Software/Firmware Security

5.1 Integrity Techniques

A firmware integrity test is performed on the runtime image of the module. The HMAC-SHA256 implemented in the module is used as an approved algorithm for the integrity test. If the test fails, the module enters an error state where no cryptographic services are provided, and data output is prohibited i.e., the module is not operational.

The OCS ROM component of the module is a non-reconfigurable memory (specifically masked ROM), which is exempt from the requirements of integrity test. The vendor performed memory degradation testing to assert that the memory will not degrade before 10 (ten) years of manufacture date, thus complying with the requirements of IG 5.A.

5.2 Initiate on Demand

The on-demand integrity self-tests can be invoked by the user performing reboot, the device which will cause pre-operational and conditional self-tests to run.

5.3 Additional Information

The Converged Security Management Engine (CSME) Driver i.e., module’s firmware, is made up of a single component, provided in the form of binary executable code. The firmware wholly contains the executable form without further compilation.

6 Operational Environment

6.1 Operational Environment Type and Requirements

The module operates in a non-modifiable operational environment per FIPS 140-3 security level 2 specifications. The operator cannot modify the firmware component of the module. The module runs on an internal customized proprietary OS (i.e., CSME OS) within the Intel SoC.

Type of Operational Environment: Non-Modifiable

7 Physical Security

7.1 Mechanisms and Actions Required

Mechanism	Inspection Frequency	Inspection Guidance
Hard tamper-evident coating	Determined by the operator	Observe the coating surrounding the chip for any signs of damage

Table 16: Mechanisms and Actions Required

The module is a hybrid firmware module that operates on a single-chip standalone platform which conforms to the Level 2 requirements for physical security. The single chip cryptographic module is a production grade component that include standard passivation (e.g., a sealing coat applied over the chip circuitry to protect it against environmental and other physical damage). The layering process which is used to embed the die into the PCB of the single chip computing platform also provides opacity that prevents viewing internal construction within the visible spectrum. The single chip enclosure prevents accessing of the module's hardware components without leaving physical tamper evidence.

8 Non-Invasive Security

This module does not implement any non-invasive security mechanism and therefore this section is Not Applicable (N/A).

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Stored as plaintext	Static

Table 17: Storage Areas

The symmetric keys and HMAC keys are provided to the module via API input parameters and are zeroized by the module before they are released in the memory. Asymmetric public and private keys are provided to the module via API input parameters and are destroyed by the module before they are released in the memory.

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API input parameters	External Source	RAM	Plaintext	Manual	Electronic	
API output parameters	RAM	External Source	Plaintext	Manual	Electronic	

Table 18: SSP Input-Output Methods

The module does not support manual key entry or intermediate key generation key output. SSPs entered into the module are electronically entered in plain text form. SSPs are output from the module in plain text form if required by the calling application.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Power Cycling	Powers down RAM	Power to RAM is lost causing all values stored in memory to be zeroized.	Invoked after device shut down
Automatic Zeroization	Performed at the end of every service utilizing SSPs	Overwrites memory occupied by keys with "zeros" before de-allocating the memory.	Automatically by the module

Table 19: SSP Zeroization Methods

The memory occupied by SSPs is stored in RAM during runtime, allocated by regular memory allocation operating system calls. Every service of the module performs a zeroization operation as the last step before exiting from the function. The zeroization operation overwrites the memory occupied by keys with “zeros” before de-allocating the memory with the regular memory de-allocation operating system call. In addition, the RAM is volatile so zeroization of all SSPs can be performed by power-cycling (i.e., reset) or power-off the computing platform.

Additionally, while zeroization is in progress, data output is inhibited. Once a SSP is zeroized, it is no longer retrievable. The module uses an implicit indicator to express the completion of the zeroization operation. Zeroization is performed by the module through the course of executing its services. The module implicitly indicates the success of the zeroization by accepting the proceeding request. If the module accepts the next service request, this implicitly indicates that the zeroization of the previous service request successfully completed.

Lastly, temporary SSPs are zeroized when they are no longer needed.

9.4 SSPs

Keys residing in internal storage can only be accessed using the defined API. Memory and process space is protected from unauthorized access by the operating system. Only the process that creates or imports keys can use or export them.

According to FIPS 140-3, Sensitive Security Parameters (SSPs) consist of Critical Security Parameters (CSPs) and Public Security Parameters (PSPs). The following table summarizes all CSPs, and PSPs employed by the cryptographic module:

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES keys	Keys used by AES	128-bits and 256-bits - 128-bits or 256-bits	Symmetric - CSP			AES-CBC AES-CFB128 AES-CMAC AES-CTR AES-ECB AES-GCM AES-OFB
HMAC keys	Keys used by HMAC	Minimum of 112-bits - Minimum of 112-bits	Symmetric - CSP			HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512
Module generated RSA public key	public key for RSA generated by the module	2048-bits - 112-bits	Asymmetric - PSP	RSA KeyGen		
Module generated RSA private key (Including intermediate keygen values)	private key for RSA generated by the module	2048-bits - 112-bits	Asymmetric - CSP	RSA KeyGen		RSA decapsulation RSA Decryption Primitive (CVL) RSA SigGen RSA Signature Primitive (CVL)
RSA public key	public key for RSA	1024-bits, 2048-bits, 3072-bits or 4096-bits - 80-bits, 112-bits, 128-bits or 150-bits	Asymmetric - PSP			RSA encapsulation RSA SigVer
RSA private key	private key for RSA	2048-bits, 3072-bits or 4096-bits - 112-bits, 128-bits or 150-bits	Asymmetric - CSP			RSA SigGen RSA Signature Primitive (CVL)
ECDSA/ECDH (ECC) public key (including intermediate keygen values)	public key for ECDSA and ECDH	P-256, P-384 - 128-bits or 192-bits	Asymmetric - PSP	ECDSA KeyGen		ECDSA KeyVer ECDSA SigVer KAS-ECC

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
ECDSA/ECDH (ECC) private key (including intermediate keygen values)	private key for ECDSA and ECDH	P-256, P-384 - 128-bits or 192-bits	Asymmetric - CSP	ECDSA KeyGen		ECDSA KeyVer ECDSA SigGen KAS-ECC
Shared Secret	Shared secret established by ECDH	128-bits or 192-bits - 128-bits or 192-bits	Symmetric - CSP		KAS-ECC	
SP 800-56C KDF derived key	Key derived by 56C KDF	128-bits or 192-bits - 128-bits or 192-bits	Symmetric - CSP		KAS-ECC	
KBKDF key derivation key	Key used for KBKDF key derivation	Contingent on key derivation key and length of output - Contingent on key derivation key and length of output	Symmetric - CSP			KBKDF
KBKDF derived key	Key derived by KBKDF	Contingent on key derivation key and length of output - Contingent on key derivation key and length of output	Symmetric - CSP	KBKDF		
Entropy Input String	Random bits produced by the entropy source	256-bits - 256-bits	Entropy - CSP			Counter DRBG
DRBG Seed	Seed used by DRBG	256-bits - 256-bits	DRBG - CSP	Counter DRBG		Counter DRBG
DRBG internal state: (V and Key values)	The CTR DRBG working state. Contains the current V and Key values	256-bits - 256-bits	DRBG - CSP	Counter DRBG		Counter DRBG

Table 20: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES keys	API input parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	
HMAC keys	API input parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Module generated RSA public key	API output parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG internal state: (V and Key values):Derived From Module generated RSA private key (Including intermediate keygen values):Paired With
Module generated RSA private key (Including intermediate keygen values)		RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG internal state: (V and Key values):Derived From Module generated RSA public key:Paired With
RSA public key	API input parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	RSA private key:Paired With
RSA private key	API input parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	RSA public key:Paired With
ECDSA/ECDH (ECC) public key (including intermediate keygen values)	API input parameters API output parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG internal state: (V and Key values):Derived From ECDSA/ECDH (ECC) private key (including intermediate keygen values):Paired With
ECDSA/ECDH (ECC) private key (including intermediate keygen values)	API input parameters API output parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG internal state: (V and Key values):Derived From ECDSA/ECDH (ECC) private key (including intermediate keygen values):Paired With
Shared Secret		RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	ECDSA/ECDH (ECC) public key (including intermediate keygen values):Established by ECDSA/ECDH (ECC) private key (including intermediate keygen values):Established by SP 800-56C KDF derived key:Derives
SP 800-56C KDF derived key	API output parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	Shared Secret:Derived From
KBKDF key derivation key	API input parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	KBKDF derived key:Derives
KBKDF derived key	API output parameters	RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	KBKDF key derivation key:Derived From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Entropy Input String		RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG Seed:Derives
DRBG Seed		RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	Entropy Input String:Derived From DRBG internal state: (V and Key values):Derives
DRBG internal state: (V and Key values)		RAM:Plaintext	Until no longer needed	Power Cycling Automatic Zeroization	DRBG Seed:Derived From Module generated RSA public key:Derives Module generated RSA private key (Including intermediate keygen values):Derives ECDSA/ECDH (ECC) public key (including intermediate keygen values):Derives ECDSA/ECDH (ECC) private key (including intermediate keygen values):Derives

Table 21: SSP Table 2

9.5 Transitions

The SHA-1 algorithm as implemented by the module will be non-approved for all purposes, starting January 1, 2031.

10 Self-Tests

10.1 Pre-Operational Self-Tests

The module performs pre-operational firmware integrity tests automatically when the computing platform is powered on, and the module is loaded into memory. The module's OCS ROM first performs an HMAC-SHA-256 conditional cryptographic algorithm self-test (CAST) and after successfully passing, the module performs an integrity test of the module's firmware component (Converged Security Management Engine (CSME) Driver) by computing an HMAC-SHA-256 value of the binary and comparing it with the value stored in the module that was computed at build time. If the HMAC values do not match, the test fails, and the module enters the Error state. While the module is performing pre-operational firmware integrity test, the module's data output is inhibited.

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A3362)	HMAC-SHA2-256 with 256-bit key	Message Authentication Code (MAC)	SW/FW Integrity	Successful boot	Performed on system startup

Table 22: Pre-Operational Self-Tests

10.2 Conditional Self-Tests

The module performs a conditional cryptographic algorithm self-test (CAST) on all Approved cryptographic algorithms supported in the approved mode of operation and per IG 10.3.A. The conditional cryptographic algorithm self-tests are performed before the first use of the related algorithm: First the AES ECB, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512, and KAS-SSC are self-tested in hardware, prior to the integrity test of the firmware component, and then the rest of the CASTs are automatically performed after the integrity test completes successfully (and before the module enters the operational state[2]). If any of the cryptographic algorithm self-test fails, the module will enter the Error State, wherein all data output is inhibited. The pre-operational and conditional algorithm tests performed are shown in the following table:

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A3362)	128-bit key	KAT	CAST	Module is operational	Encryption/Decryption	Upon first invocation of service that uses the algorithm
AES-GCM (A3362)	128-bit key	KAT	CAST	Module is operational	Encryption	Upon first invocation of service that uses the algorithm
AES-CMAC (A3362)	128-bit key	KAT	CAST	Module is operational	Signature Generation (Encryption)	Upon first invocation of service that uses the algorithm
HMAC-SHA-1 (A3362)	SHA-1	KAT	CAST	Module is operational	Message authentication	Upon first invocation of service that uses the algorithm
HMAC-SHA2-256 (A3362)	SHA2-256	KAT	CAST	Module is operational	Message authentication	Upon first invocation of service that uses the algorithm

[2] Operational State is the state where the operator can request services of the module.

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC-SHA2-512 (A3362)	SHA2-512	KAT	CAST	Module is operational	Message authentication	Upon first invocation of service that uses the algorithm
SHA-1 (A3362)	SHA-1	KAT	CAST	Module is operational	Message Digest	Upon first invocation of service that uses the algorithm
SHA2-256 (A3362)	SHA2-256	KAT	CAST	Module is operational	Message Digest	Upon first invocation of service that uses the algorithm
SHA2-512 (A3362)	SHA2-512	KAT	CAST	Module is operational	Message Digest	Upon first invocation of service that uses the algorithm
ECDSA SigGen (FIPS186-4) (A3362)	P-256 curve and SHA-256 message digest	KAT	CAST	Module is operational	Digital Signature Generation	Upon first invocation of service that uses the algorithm
ECDSA SigVer (FIPS186-4) (A3362)	P-256 curve and SHA-256 message digest	KAT	CAST	Module is operational	Digital Signature Verification	Upon first invocation of service that uses the algorithm
Counter DRBG (A3362)	256-bit key	KAT	CAST	Module is operational	Deterministic Random Bit Generation	Upon first invocation of service that uses the algorithm
RSA SigGen (FIPS186-4) (A3362)	PKCS#1 v1.5, 2048 modulus and SHA-256 message digest	KAT	CAST	Module is operational	Digital Signature Generation	Upon first invocation of service that uses the algorithm
RSA SigVer (FIPS186-4) (A3362)	PKCS#1 v1.5, 2048 modulus and SHA-256 message digest	KAT	CAST	Module is operational	Digital Signature Verification	Upon first invocation of service that uses the algorithm
KTS-IFC (A3362)	2048 modulus	KAT	CAST	Module is operational	Encryption/Decryption	Upon first invocation of service that uses the algorithm
KDF SP800-108 (A3362)	HMAC-SHA2-384	KAT	CAST	Module is operational	Key Derivation	Upon first invocation of service that uses the algorithm
KAS-ECC Sp800-56Ar3 (A3362)	P-256 curve, One-Step KDF using HMAC-SHA-384	KAT	CAST	Module is operational	Shared Secret Computation and Key Derivation	Upon first invocation of service that uses the algorithm
ECDSA KeyGen (FIPS186-4) (A3362)	P-256, P-384	PCT	PCT	Key pair returned to caller	Key Generation	Performed every time ECC key pair is generated
RSA KeyGen (FIPS186-4) (A3362)	2048 modulus	PCT	PCT	Key pair returned to caller	Key Generation	Performed every time RSA key pair is generated

Table 23: Conditional Self-Tests

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A3362)	Message Authentication Code (MAC)	SW/FW Integrity	On demand	Manually

Table 24: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A3362)	KAT	CAST	On demand	Manually
AES-GCM (A3362)	KAT	CAST	On demand	Manually
AES-CMAC (A3362)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A3362)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A3362)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A3362)	KAT	CAST	On demand	Manually
SHA-1 (A3362)	KAT	CAST	On demand	Manually
SHA2-256 (A3362)	KAT	CAST	On demand	Manually
SHA2-512 (A3362)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A3362)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A3362)	KAT	CAST	On demand	Manually
Counter DRBG (A3362)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A3362)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A3362)	KAT	CAST	On demand	Manually
KTS-IFC (A3362)	KAT	CAST	On demand	Manually
KDF SP800-108 (A3362)	KAT	CAST	On demand	Manually
KAS-ECC Sp800-56Ar3 (A3362)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA KeyGen (FIPS186-4) (A3362)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-4) (A3362)	PCT	PCT	On demand	Manually

Table 25: Conditional Periodic Information

The on-demand and periodic self-tests can be invoked by the user performing reboot, the device which will cause pre-operational and conditional self-tests to run.

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error State	No further communication is possible with the module until the module is reset.	Pre-operational firmware integrity test failure Conditional self-test failures other than PCT	Device reset	An error code is provided

Table 26: Error States

The module implements one error state. If any of the self-tests described in sections above fails, the module indicates the error indicator associated with the specific error by invoking the crypto_fips_error_handler() function and causes the module to enter the error state. In the error state, no cryptographic services are provided, and data output is prohibited. When the module is in the error state, the only method to recover is to reset the computing platform which results in the module reperforming the pre-operational firmware integrity test and the conditional cryptographic algorithm self-tests. The module will only enter the operational state after successfully passing both.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The firmware component of the module is distributed as part of the CSME Device Driver firmware. Firmware is released on VIP site https://platformsw.intel.com. Only Original Equipment Manufacturers (OEM) with signed Intel agreements can download this firmware.

The module is contained within one of the platforms listed in Section 2.2. These Intel platforms are a tightly coupled component of 12th Generation Intel® Core™ chipsets. These platforms can be bundled with CPU as a kit, or outside the CPU packages as a discreet component mounted on the Printed Circuit Board (PCB). Intel requires their Original Equipment Manufacturer (OEM) partners that create, market, and sell these systems to meet the brand validation requirements and testing to ensure they have been designed and constructed with the proper components including CPU and Intel chipsets. Intel's brand validation tool would detect any mismatch of CPU and chipset for any system being designed.

Intel manages and implements security best practices throughout every step of their supply chain and works closely with their partners (i.e., Original Design Manufacturer and Original Equipment Manufacturer) to ensure that they meet Intel's requirements for secure supply chain processes as specified in partner contract agreements. Therefore, end customers can be assured that any system had been designed and tested to conform to Intel's requirements will always have an Intel PCH that contains the module.

11.2 Administrator Guidance

The following security guidance for Crypto Officer role is described below:

- To enable the module for use in a FIPS validated configuration, the Crypto Officer must first perform initialization of the module. If FIPS operations are not enabled within the BIOS settings, then the module is not a 140-3 validated module and cannot enter the approved mode which means no FIPS services will be available. The Crypto Officer shall perform the following steps to initialize the module.
  - Power on the Host Platform and enter the BIOS menu setting.
  - Enter "FIPS mode" submenu.
  - Set "FIPS Mode Select" to <Enabled>.
  - Save and exit the BIOS menu.

The Host Platform will the power-cycle (i.e., reset) and proceed to boot. Once "FIPS Mode Select" is Enabled, the CSME OS will set the crypto_fips_en file which serves the control input the module and initializing it as a FIPS 140-3 validated module following power-on.

11.3 Non-Administrator Guidance

The following security guidance for User role is described below:

- The User of the module can call the API function crypto_drv_fips_mode_status() to check if the module is an FIPS 140-3 validated module. If the call returns 1, the module is an FIPS 140-3 validated module; it returns 0 if the module is not an FIPS 140-3 validated module. Note, this is not the service indicator; the service indicator is provided as described in Section 4.3.

- SP800-56B Assurances
  The following statements explain the SP800-56Brev2 assurances found in its Section 5:
  - Section 5.1 – The module uses an approved hash function (SHS, Cert. #A3362) for mask generation during RSA-OEAP encryption.
  - Section 5.2 – N/A, the module does not implement key confirmation.
  - Section 5.3 – The module uses an approved random bit generator (CTR_DRBG, Cert. #A3362) when generating random values.
  - Section 5.4 and Section 5.5 – N/A, the module does not implement a key agreement scheme (i.e., KAS1).
  - Section 5.6 – N/A, the module does not implement key confirmation.

In addition, the following statements explain the SP800-56Brev2 assurances found in its Section 6 (specifically SP800-56Brev2 *Section 6.4 Required Assurances*):

- Prior to the use of the key pair in a key-establishment transaction, assurances required by the key-pair owner are obtained in the following way (Note: only keys generated following this guidance shall be compliant to SP800-56Brev2):
  1) The entity requesting the RSA key unwrapping (decapsulation)service from the module, shall only use an RSA private key that was generated by an active FIPS validated module that implements FIPS 186-5 compliant RSA key generation service and performs the key pair validity and the pairwise consistency as stated in section 6.4.1.1 of the SP 800-56BRev2. Additionally the entity shall renew these assurances over time by using any method described in section 6.4.1.5 of the SP 800-56BRev2.
  2) For use of an RSA key wrapping (encapsulation) service in the context of key transport per IG D.G,
     - the entity using the module, shall verify the validity of the peer's public key using the public key validation service of the module.
     - the entity using the module, shall confirm the peer's possession of private key by using any method specified in section 6.4.2.3 of the SP 800-56BRev2.

  Only after the above assurances are successfully met, shall the entity use the peer's public key to perform the RSA key wrapping (encapsulation) service of the module.

- The RSA Decryption Primitive (CVL) shall only be used within the context of a SP 800-56Brev2 KTS.

11.4 Design and Rules

This section is N/A for this module.

11.5 Maintenance Requirements

This section is N/A for this module.

11.6 End of Life

The module automatically performs secure sanitization at the conclusion of any service performed by the module. Since the module does not retain any persistent SSPs, the procedures for secure sanitization of the cryptographic module are inherently met.

12 Mitigation of Other Attacks

12.1 Attack List

The module provides mechanism to protect against RSA timing attacks. The OCS's (i.e., module's hardware component) big-number arithmetic can perform the modular exponentiation operations in constant time. This means, the time taken is only dependent on the size of operands and not dependent on the value of the operands. During modular exponentiation, an extra mathematical step is needed when the processed bit of the key is 1 compared to a zero bit. The OCS's big-number arithmetic implements a "dummy" step when processing a zero bit from the key such that this processing time is identical to the processing time of a set bit. Using this approach, an observer is unable to determine the number of set and unset bits from observing the timing behavior of the modular exponentiation operation. The CSME Crypto Driver takes advantage of this feature by enabling the functionality in the OCS for private key operations. This implies that the computation time using the private key is constant, hence mitigating timing attacks.

The OCS's AES block cipher in OCS supports an implementation that is resistant to DPA (Differential Power Analysis) attacks. The mechanism implemented is based on masking AES inputs at every stage with a pseudo-random mask. The seed for the pseudorandom mask generator is programmable.

The OCS's ECC has protection against known DPA Attacks. This is achieved by randomizing the inputs so there is no correlation to the power consumed and ECC operations. This is done by transforming inputs from one coordinate system (Affine) to another coordinate system (Randomized Jacobian).