Intel(R) Converged Security and Manageability Engine (CSME) Crypto Module for Tiger Point PCH, Mule Creek Canyon PCH, and Rocket Lake PCH

Hardware Version 4.0

Firmware Versions: 4.0, 4.1, 4.2, 4.3

FIPS 140-2 Non-Proprietary Security Policy

Document Version 1.4

Last update: 2023-05-01

Prepared by:

atsec information security corporation

9130 Jollyville Road, Suite 260

Austin, TX 78759

www.atsec.com

Table of Contents

Copyrights and Trademarks

1. Cryptographic Module Specification

1.1. Statement of Module Security Policy

This document is the non-proprietary FIPS 140-2 Security Policy of the Firmware-Hybrid Crypto Module. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 Firmware-Hybrid module.

The table below shows the security level claimed for each of the eleven sections that comprise the FIPS 140-2 standard:

FIPS 140-2 Section		Security Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services and Authentication	1
4	Finite State Model	1
5	Physical Security	1
6	Operational Environment	N/A
7	Cryptographic Key Management	1
8	EMI/EMC	1
9	Self-Tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	1
Overall Level		1

Table 1 - Security Levels

1.2. Module Overview

The Intel(R) Converged Security and Manageability Engine (CSME) Crypto Module for Tiger Point PCH, Mule Creek Canyon PCH, and Rocket Lake PCH (hereafter referred to as “the module”) is classified as a firmware-hardware hybrid module for FIPS 140-2 purpose. The module consists of both hardware and firmware. The hardware portion provided by the Offload and Crypto Subsystem (OCS) validated under CMVP certificate (#4025). The firmware portion is the CSME Crypto Driver. The IUT is embedded within an Intel Platform Controller Hub (PCH) chip. Figure 1 shows a typical PCH chip. The IUT was validated on the following PCH chips:

- Tiger Point PCH for Intel Tiger Lake-U (TGL-U)
- Rocket Lake PCH for Intel Rocket Lake-S (RKL-S)
- Tiger Point PCH for Intel Tiger Lake-H (TGL-H)
- Mule Creek Canyon PCH for Intel Elkhart Lake (EHL)

In this document, “CSME Crypto Driver”, “CSME firmware”, “CSME FW” and “CSME Crypto Driver firmware” are used interchangeably. They all refer to the firmware component of this hybrid module. “CSME” is used as a shorthand for the entire hybrid module including its firmware component and hardware component OCS.

Figure 1: Intel PCH

1.3. Module Components

The components of the hybrid cryptographic module are specified in the following table:

Component	Type	Version Numbers	Description
Converged Security Management Engine (CSME) Crypto Driver	Firmware	4.0 [1], 4.1 [2], 4.2 [3], 4.3 [4][*]	Firmware running on an internal proprietary Operating System to communicate with the hardware components of the module in the Intel PCH chipset.
Offload and Crypto Subsystem (OCS)	Hardware	4.0	This enables: Platform secure boot; Runtime crypto acceleration; Secured Global Secret Key storage. It also contains AES, RC4, SHA-1, 256, 224, 384, 512 and HMAC. This OCS is embedded within the Intel PCH chipset.
run_bist	File	N/A	The Crypto Driver will read this file from battery-backed non-volatile storage. This file tracks if the Crypto Driver has previously passed the full set of power-up self-tests.
fips_hmac	File	N/A	This is a file located in the file system of the internal customized proprietary OS to contain the HMAC-SHA-256 hash value for integrity check of the module.

Table 2 - Cryptographic Module Components

[*] Refer to Table 3 for the corresponding platform configuration.

The module has been tested on the following single-chip standalone platforms:

Platform	Processor	Operating System
Tiger Point PCH for Intel Tiger Lake-U [1]	Lakemont 3.7	Embedded customized proprietary OS version 15.0.20.1648
Rocket Lake PCH for Intel Rocket Lake-S [2]	Lakemont 3.7	Embedded customized proprietary OS version 15.0.22.1571
Tiger Point PCH for Intel Tiger Lake-H [3]	Lakemont 3.7	Embedded customized proprietary OS version 15.0.30.1716
Mule Creek Canyon PCH for Intel Elkhart Lake [4]	Lakemont 3.7	Embedded customized proprietary OS version 15.40.10.2204

Table 3 - Tested Platforms

Note: Per IG G.5, CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

1.4. Block Diagrams

The physical boundary of the module is the physical boundary of the Intel PCH chipset that contains the module. Consequently, the embodiment of the module is a single-chip standalone cryptographic module. The physical crypto boundary is represented by the dashed purple lines below.

The module provides cryptographic services to applications through an application program interface (API). The cryptographic logical boundary consists of the firmware component CSME Crypto Driver, the hardware component OCS along with the fips_hmac and run_bist files. The logical crypto boundary is represented by the dashed red lines below.

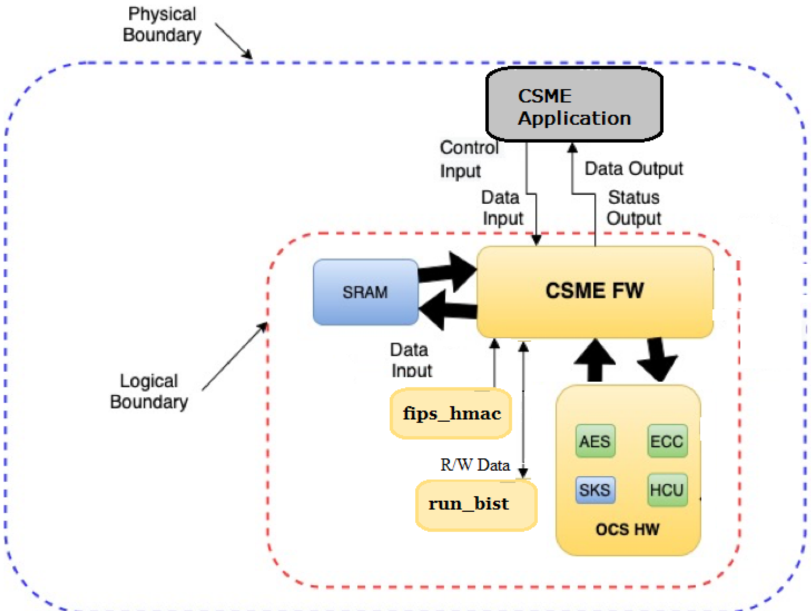

Figure 2 - Logical Block Diagram

1.5. Modes of operation

The module supports two modes of operation:

- In "FIPS mode" (the FIPS Approved mode of operation) only approved or allowed security functions with sufficient security strength can be used. Table 4 lists the FIPS approved functions.
- In "non-FIPS mode" (the non-Approved mode of operation) only non-approved security functions can be used. Table 5 lists the non-FIPS functions.

There is a difference between entering/exiting FIPS mode and enabling/disabling FIPS operations in BIOS. To enable FIPS operations, the module must first be configured as a FIPS 140-2 module from within the BIOS menu setting at power-on. The system will be reset in order to transition to/from FIPS operational mode after the setting is configured. Once FIPS operations are enabled in the BIOS settings, the module is initialized as a FIPS 140-2 validated module. This means the module will now implicitly be in FIPS mode if the user is requesting a FIPS algorithm. And the module is implicitly not in FIPS mode if the user is requesting a non-FIPS algorithm. If FIPS operations are not enabled within the BIOS settings, then the module is not a 140-2 validated module and cannot enter FIPS mode which means no FIPS services will be available. Critical security parameters (CSP) used or stored while in FIPS mode are not used in non-FIPS mode, and vice versa.

1.6. FIPS-Approved Algorithms

Table 4 below lists the FIPS Approved cryptographic algorithms provided by the OCS hardware with CAVP Certificate #A668 and the algorithms provided by the CSME firmware with CAVP Certificate #A667:

FIPS Approved Algorithm	Cryptographic Function	Standards	CAVP Certificates
AES Key Size: • 128-bit • 256-bit Modes: • CMAC[†] • CFB128 • OFB	Encryption Decryption	FIPS 197, SP800-38A SP800-38B	A667
AES Key Size: • 128-bit • 256-bit Modes: • ECB • CBC • CTR		FIPS 197, SP800-38A	A668
SHS • SHA-1 • SHA-224 • SHA-256 • SHA-384 • SHA-512	Hashing	FIPS 180-4	A668

[†] MAC Length tested: 128 bits

FIPS Approved Algorithm	Cryptographic Function	Standards	CAVP Certificates
HMAC with: • SHA-1 • SHA-224 • SHA-256 • SHA-384 • SHA-512	Message Authentication Code	FIPS 198-1	A668
ECDSA Curves: • P-256 • P-384	Key Generation Key Verification	FIPS 186-4	A667
ECDSA Curves: • P-256 • P-384 with • SHA-224 • SHA-256 • SHA-384 • SHA-512	Digital Signature Generation and Verification		A668
DRBG Key Size: • 256-bit Modes: • Counter	Random Number Generation	SP800-90A	A667
ECDH Shared Secret Computation Ephemeral Unified: P-256, P-384	Key agreement scheme (KAS-SSC)	SP800-56A Rev3	A667
KBKDF KDF Mode: • Counter MAC Mode: • HMAC-SHA-1 • HMAC-SHA-224 • HMAC-SHA-256 • HMAC-SHA-384 • HMAC-SHA-512	Key Derivation Function	SP800-108	A667
RSA Modulus: • 2048 Mode: • B.3.3 • fixed public exponent	Key Generation	FIPS 186-4	A667

FIPS Approved Algorithm	Cryptographic Function	Standards	CAVP Certificates
RSA Modulus: • 2048 • 3072 • 4096 Mode: • PKCS#1 v1.5 • PKCSPSS	Digital Signature Generation		A667
RSA Modulus: • 1024 • 2048 • 3072 Mode: • PKCS#1 v1.5 • PKCSPSS	Digital Signature verification	FIPS 186-4	A667
RSA Modulus: • 2048 • 3072 • 4096	Key Transport using RSA-OAEP	SP800-56B	Vendor Affirmed
Password-Based Key Derivation version 2 (PBKDFv2) with HMAC-SHA-1 and HMAC-SHA-256	CSME Crypto Driver using OCS hash support	SP 800-132	Vendor Affirmed
SP 800-56C KDF using HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512	Key Derivation Function (KDA)	SP 800-56C	Vendor Affirmed
Cryptographic Key Generation (CKG) for asymmetric keys	RSA and ECDSA Key Generation	SP800-133	Vendor Affirmed
ENT (P)	Random Number Generation	SP800-90B	N/A

Table 4 – FIPS approved Cryptographic Algorithms provided by CSME Crypto Driver and OCS

The module implements the following non-Approved algorithms:

Non-Approved Algorithm	Use
AES GCM	Authenticated Data Encryption and Decryption
RC4, SM4	Data Encryption and Decryption
MD5, SM3	Message Digest
HMAC-MD5	Message Authentication Code
HMAC with keys less than 112 bits	Message Authentication Code
RSA with 1024 bits modulus size	Key Generation
RSA with MD5, SM3 or SHA-1	Digital signature generation
RSA using MD5 or SM3	Digital Signature Verification
AES key wrapping with KW	Key Transport
RSA key wrapping using MD5 hash	
RSA key wrapping using PKCS#1 v1.5 padding scheme with any key size	
Trusted Computing Group (TCG) digital signature scheme using elliptic curves: EC Schnorr, EC commit computation, EC DAA	Digital Signature Generation and Verification
Barreto Naehrig, SECG P256k1, Brainpool384 or SM2	shared secret computation
	Key Pair Generation and Public Key Verification
	Signature Generation and Verification
ECDH using One-Pass DH	shared secret computation or Key Agreement

Table 5 - non-approved algorithms

Regarding to the services available in FIPS mode of operation and non-FIPS mode of operation, please refer to Table 7 - Services in FIPS mode of operation and Table 8 - Services in non-FIPS mode of operation in 3.2 Services.

2. Cryptographic Module Ports and Interfaces

As the module is a firmware-hybrid, the data should enter and exit the module via the logical interface through firmware. The logical interfaces are the application program interface (API) through which applications request services from the firmware (i.e., CSME Crypto Driver). The hardware interfaces are the interface for data in and out of the hardware components of the module. The following table summarizes the four logical interfaces:

FIPS 140-2 Interface	Logical Interface	Physical Interface
Data Input	API input parameters pointing to data stored in RAM fips_hmac file, run_bist file.	DMA
Data Output	API output parameters pointing to RAM locations for storing output data.	DMA
Control Input	API function calls.	IOSF
Status Output	API return codes.	IOSF

Table 6 - Ports and Interfaces

3. Roles, Services and Authentication

3.1. Roles

The module supports the following roles:

- **User Role:** Performs all the services (in both FIPS mode and non-FIPS mode), except for the module installation and configuration.
- **Crypto Officer role:** performs module initialization.

The User and Crypto Officer roles are implicitly assumed by the entity accessing the module services.

3.2. Services

The module provides services to users that assume one of the available roles. All services are described in detail in the user documentation. The following table lists the Approved services and the non-Approved but allowed services in FIPS mode of operation, the roles that can request the service, the Critical Security Parameters involved and how they are accessed:

Service	Role	Key/CSP	Access
AES symmetric encryption and decryption	User	AES 128-bit and 256-bit keys	Read
RSA key pair generation	User	RSA public-private keys with modulus size 2048.	Create
RSA digital signature generation based on PKCS#1 v1.5 and PSS scheme		RSA public-private keys with modulus size 2048, 3072, and 4096 bits.	Read
RSA digital signature verification based on PKCS#1 v1.5 and PSS scheme		RSA public-private keys with modulus size 1024, 2048, 3072, and 4096 bits.	Read
RSA-based Key Transport using RSA-OAEP (SP 800-56B Section 9)	User	RSA public-private keys with modulus size 2048, 3072, and 4096 bits.	Read
ECDSA key pair generation	User	ECDSA public-private keys with P-256 and P-384 curve	Create
ECDSA public key verification			Read
ECDSA signature generation			Read
ECDSA signature verification			Read
ECDH Shared Secret Computation (KAS-SSC)	User	ECDSA public-private keys with P-256 and P-384 curve, shared secret	Read, Create
Key agreement scheme using ECDH Shared Secret Computation (KAS-SSC) and SP 800-56C KDF (KDA)	User	ECDSA public-private keys with P-256 and P-384 curve, shared secret, derived key	Read, Create
Message digest generation	User	None	None
MAC generation and verification	User	At least 112 bits HMAC key	Read
Random Number Generation	User	DRBG seed (consisting of Entropy Input String), V and Key	Read, Update
SKS Key Storing (plaintext)	User	HMAC and AES keys	Read, Write
SKS Key Storing (encrypted)	User	AES Master Key, HMAC and AES keys	Read, Write

Service	Role	Key/CSP	Access
Password-based Key Derivation Function	User	Password, at least 112 bits derived keying material	Read, Create
SP 800-56C based Key Derivation Function	User	Shared secret, derived key	Read, Create
KBKDF using Counter and HMAC Mode	User	Key derivation key, at least 112 bits derived keying material	Read, Create
Show status	User	None	None
Self-Tests	User	None	None
Zeroization	User	All CSPs	Zeroize
Module Initialization	Crypto Officer	None	None

Table 7 - Services in FIPS mode of operation

The following table lists the services only available in non-FIPS mode of operation:

Service	Role
AES-GCM with external IV	User
Message digests using MD5 or SM3	User
MAC generation using HMA-MD5	User
MAC generation using less than 112 bits HMAC key	User
Symmetric encryption / decryption using RC4 or SM4	User
RSA key generation with 1024-bit modulus size	User
RSA signature generation with MD5, SM3 or SHA-1 for any modulus size	User
RSA signature verification with MD5, or SM3 for any modulus size	User
AES key wrapping with KW	User
RSA key wrapping using MD5 hash	User
RSA key wrapping using PKCS#1 v1.5 padding scheme with any key size	User
Trusted Computing Group (TCG) digital signature scheme using elliptic curves: EC Schnorr, EC commit computation, EC DAA	User
shared secret computation with Barreto Naehrig, SECG P256k1, Brainpool384 and SM2 curves	User
Key Pair Generation and Public Key Verification with Barreto Naehrig, SECG P256k1, Brainpool384 and SM2 curves	User
Signature Generation and Verification with Barreto Naehrig, SECG P256k1, Brainpool384 and SM2 curves	User
EC Diffie-Hellman shared secret computation or Key Agreement using One Pass DH	User

Table 8 - Services in non-FIPS mode of operation

3.3. Operator Authentication

The module does not implement authentication. The role is implicitly assumed based on the service requested.

4. Physical Security

The module is a firmware-hybrid module that operates on a single-chip standalone platform which conforms to the Level 1 requirements for physical security. The hardware portion of the cryptographic module is a production grade component. The cryptographic module is used in a commercial off the shelf (COTS) computer device. The computer device is comprised of production grade components with standard passivation (e.g. a sealing coat applied over the chip circuitry to protect it against environmental and other physical damage) and a production grade enclosure that completely surrounds the cryptographic module.

5. Operational Environment

5.1. Applicability

The module operates in a limited operational environment per FIPS 140-2 level 1 specifications. The operator cannot modify the firmware component of the module. The module runs on an internal customized proprietary OS within the Intel PCH chipset.

5.2. Policy

The operating system is restricted to a single operator; concurrent operators are explicitly excluded. The application that requests cryptographic services is the single user of the module.

6. Cryptographic Key Management

The following table summarizes the Keys and Critical Security Parameters (CSPs) that are used by the cryptographic services implemented in the module:

Name	Generation / Entry	Storage	Zeroization
AES keys	The key is passed into the module via API input parameters. The key can also be loaded from the SKS directly to the register.	Stored as plaintext in the RAM or SKS.	Keys in RAM can be zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory. Keys stored in SKS can be zeroized when by setting cse_zeroing_en bit to '1' in control register.
HMAC keys	The key is passed into the module via API input parameters. The key can also be loaded from the SKS directly to the register.	Stored as plaintext in the RAM or SKS.	Keys in RAM can be zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory. Keys stored in SKS can be zeroized when by setting cse_zeroing_en bit to '1' in control register.
RSA key pair	The prime number is generated using SP 800-90A DRBG. The RSA public-private keys are generated using FIPS 186-4 RSA Key Generation method. The key pair is passed into the module via API input parameters.	Stored as plaintext in the RAM.	Keys in RAM can be zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory.
ECDSA key pair	The ECDSA public-private keys are generated using FIPS 186-4 ECDSA Key Generation method and the random value used in key generation is generated using SP 800-90A DRBG. The key pair is passed into the module via API input parameters.	Stored as plaintext in the RAM.	Keys in RAM can be zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory.
Shared Secret	The shared secret is generated in the EC Diffie-Hellman key agreement function.	Stored as plaintext in the RAM.	Keys in RAM can be zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory.
Entropy Input String for DRBG seed	Obtained from the hardware ENT outside of the module's logical boundary but within its physical boundary	Stored as plaintext in the RAM.	Zeroized during the power cycle of the module.
DRBG internal V and Key	Generated internally in the DRBG.	Stored as plaintext in the RAM.	Zeroized during the power cycle of the module.
PBKDF Password	The password is passed into the module via API input parameters.	Stored as plaintext in the RAM.	zeroized when power is reset or by explicitly calling

PBKDF derived key	Generated using PBKDF algorithm	Stored as plaintext in the RAM.	memset_secure() to zeroize the memory.
Key derivation key	Passed into the module via API input parameters.	Stored as plaintext in the RAM.	zeroized when power is reset or by explicitly calling memset_secure() to zeroize the memory.
KBKDF derived key	Generated using KBKDF algorithm	Stored as plaintext in the RAM.	
SP 800-56C KDF derived key	Generated using 800-56C KDF algorithm	Stored as plaintext in the RAM.	
AES master key	The key is passed into the module via API input parameters.	Stored as plaintext in the SKS	zeroized when by setting cse_zeroing_en bit to '1' in control register.

Table 9 - Life cycle of Keys and Critical Security Parameters (CSP)

6.1. Random Number Generation

The module implements the CTR_DRBG with AES-256 with derivation function and without prediction resistance. 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 7.18 compliant ENT as the entropy source for seeding the CTR_DRBG inside the module. This entropy source is implemented outside the module's logical boundary within the physical boundary.

The module collects 576 bits of data from the entropy source for generating the initial seed during initialization of the CTR_DRBG, and reseeding which occurs less than 2^{28} times of DRBG service request. The module performs the Intel Online Health Test (OHT) on the entropy source to ensure that the amount of entropy is sufficient for a security strength of 256 bits.

6.2. Key Generation

For generating RSA and ECDSA keys the module implements asymmetric key generation services compliant with [SP800-133] and [SP800-90A]. The random value used in asymmetric key generation is obtained from the DRBG. In accordance with FIPS 140-2 IG D.12, the cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys as per SP800-133 (vendor affirmed). The module does not offer a dedicated service for generating keys for symmetric algorithms or for HMAC. NOTE: FIPS approved RSA Key Generation will only use public key exponent e = $2^{16} + 1$

6.3. Key Establishment/Key Derivation

The module supports the [800-56ARev3] EC Diffie-Hellman shared secret computation with P-256 and P-384 curves and includes SP800-56C one step KDF using HMAC. Using these algorithms, the module provides SP [800-56ARev3] Key agreement scheme using EC Diffie-Hellman Shared Secret Computation (KAS-SSC) and SP 800-56C (KDA) per IG D.8 scenario X1 path (2).

The module also supports the [SP800-56B] RSA Key Transport using OAEP with the modulus size of 2048, 3072 and 4096 bits in FIPS mode. The modulus size of 1024 bits is only available in non-FIPS mode.

CAVEAT:

- EC Diffie-Hellman key establishment methodology provides 128 or 192 bits of encryption strength.
- RSA key wrapping provides between 112 and 150 bits of encryption strength.

The module implements Password-Based Key Derivation version 2 (PBKDFv2) as defined in [SP800-132]. The PBKDFv2 function is provided as a service and returns the key derived from the provided password to the caller. The module supports HMAC-SHA-1 and HMAC-SHA-256 for PBKDF. Option 1 described in section 5.4 of [SP800-132] is provided to protect data using the derived key. The caller shall observe all requirements and should

consider all recommendations specified in SP800-132 with respect to the strength of the generated key, including the quality of the password, the quality of the salt as well as the number of iterations. The security guidance of the PBKDFv2 function is described in section 9.1.
Note: The keys derived from passwords, as shown in SP 800-132, may only be used in storage applications.

6.4. Key Entry / Output

The module does not support manual key entry or intermediate key generation key output. In addition, the module does not produce key output in plaintext format outside its physical boundary.

6.5. Key / CSP Storage

The symmetric keys and HMAC keys are provided to the module via API input parameters and are destroyed 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. The module provides Secure Key Storage (SKS) services. The keys stored in the SKS are in plaintext or encrypted with AES-CBC[§] and a 256-bit key. The keys stored in SKS are directly loaded into the register for AES or HMAC operations at the hardware level and cannot be retrieved by the CSME Crypto Driver or calling application.

6.6. Key / CSP Zeroization

The memory occupied by keys is stored in static arrays or allocated by regular memory allocation operating system calls. The firmware of the module (i.e., CSME Crypto Driver) calls the memset_secure() functions to overwrite the memory occupied by keys with “zeros” before deallocating the memory with the regular memory deallocation operating system call.

At the hardware level, two Memory-Mapped I/O (called “MMIO” in short) registers are accessible from the CSME Crypto Driver and OCS’s Kernel to store these keys temporarily: HCU_KEY and AES_KEY. These two registers are write-only (i.e., user cannot read the keys from the registers) and they are mapped to the CSME Crypto Driver only (i.e., no other process is able to access these registers); therefore, the keys are protected by the hardware architecture before the key zeroization occurs. The keys are zeroized during the power-cycle of the module or replacing by other new keys. All other keys in the hardware components for the cryptographic operations are provided via the SKS which is wired hardware-internally to the cipher engines.

[§] AES-CBC is allowed for encryption of stored keys per FIPS 140-2 IG 7.16

7. EMI/EMC

The module cannot be certified by the FCC as it is not a standalone device. It is a sub-chip embedded into the platform(s) listed in Table 3. The platform listed there are also not standalone devices, but rather intended to be used within a COTS device which would undergo standard FCC certification for EMI/EMC.

According to 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, the module is not subject to EMI/EMC regulations because it is a subassembly that is sold to an equipment manufacturer for further fabrication. That manufacturer is responsible for obtaining the necessary authorization for the equipment with the module embedded prior to further marketing to a vendor or to a user.

8. Self-Tests

The module performs power-up self-tests and conditional tests to ensure the correctness of the cryptographic algorithm implementations within the module boundary. If any self-test fails, the module enters Error state where the Error state triggers a reset of the module. This is followed by the execution of the power-up self-tests. No data output and cryptographic operation are allowed in Error state.

Pursuant with IG 9.11 requirements, the set of power-up self-tests performed by CSME Crypto Driver Firmware (Table 10) will *not* be run if the run_bist file indicates that the full set tests had previously passed. Instead the only tests executed at power up are the integrity test and the self-test performed by OCS Hardware (Table 11). The CSME Crypto Driver will read the run_bist file from battery-backed non-volatile storage. This file tracks if the Firmware has previously passed the full set of power-up self-tests.

If any self-test fails, the module enters Error state by calling the CSME reset to restart the module and rerun the power-up self-test to recover from Error state. No data output and cryptographic operation are allowed in Error state.

8.1. Power-Up Tests

The module performs power-up tests automatically when the module is loaded into memory; power-up tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected. While the module is executing the power-up tests, services are not available, and input and output are inhibited. The module does not return control to the calling application until the power-up tests are completed. Once the power-up tests are completed successfully, the module will be operational.

8.1.1. Integrity Tests

The integrity of the module is verified by comparing an HMAC-SHA-256 value calculated at run time with the HMAC value stored in the fips_hmac file that was computed at build time. If the HMAC values do not match, the test fails, and the module enters the Error state.

8.1.2. Cryptographic algorithm tests

The module performs self-tests on all FIPS-Approved cryptographic algorithms supported in the approved mode of operation, using the known answer tests (KAT) and pair-wise consistency test (PCT), shown in the following table:

Algorithm	Power-Up Tests
AES	• KAT AES-CMAC Generation (Encryption) • KAT AES-CMAC Verification (Decryption)
RSA	• KAT RSA PKCS#1 v1.5 with SHA-256 signature generation • KAT RSA PKCS#1 v1.5 with SHA-256 signature verification
	• KAT KTS-OAEP encrypt • KAT KTS-OAEP decrypt
DRBG	• KAT CTR_DRBG
SP800-108 KDF	• KAT CTR mode
ECDH Shared Secret Computation (KAS-SSC)	• KAT Shared Z Computation
SP 800-56C KDF	• KAT for one-step KDF using HMAC-SHA-384

Table 10- Self-Tests implemented by CSME Crypto Driver

Algorithm	Power-Up Tests
AES	• KAT AES-ECB Encryption • KAT AES-ECB Decryption
HMAC	• KAT HMAC-SHA-1 • KAT HMAC-SHA-512
SHS	• KAT SHA-1 is covered in the KAT for HMAC-SHA-1 as allowed with IG 9.1 • KAT SHA-224 is not required per IG 9.4 • KAT SHA-256 is covered in the Integrity Test which is allowed with IG 9.3 • KAT SHA-384 is not required per IG 9.4 • KAT SHA-512 is covered in the KAT for HMAC-SHA-512 as allowed with IG 9.1
ECDSA	• PCT ECDSA signature generation and signature verification
ENT (SP800-90B)	• OHT

Table 11- Self-Tests implemented by OCS

For the KAT, the module calculates the result and compares it with the known value. If the answer does not match the known answer, the KAT has failed, and the module enters the Error state. For the PCT, if the signature generation or verification fails, the module enters the Error state.

8.2. On-Demand self-tests

The on-demand self-tests can be invoked by the user performing these two steps:

1. Reboot and then disable FIPS in BIOS
2. Reboot, then re-enable FIPS in BIOS which will cause POST/KAT to run

During the execution of the on-demand self-tests, services are not available, and no data output or input is possible.

8.3. Conditional Tests

The module performs conditional tests on the cryptographic algorithms, using the pair-wise consistency test (PCT), Continuous Random Number Generator Test (CRNGT), and Intel Online Health Test (OHT), shown in the following table:

Algorithm	Test
ECDSA key generation	• PCT signature generation and verification using SHA-256
RSA key generation	• PCT signature generation and verification using SHA-256
ENT (SP800-90B)	• OHT
	• CRNGT

Table 12 - Conditional Tests

8.4. Entropy Health Tests

FIPS 140-2 section 4.9.2 states that a Continuous Random Number Generator Test (CRNGT) shall be used on the output of any random number generator. The CSME Crypto Driver Firmware of this module has implemented such a test which verifies that each block of data generated by the ENT is not a duplicate of the previous block of data.

In addition, Intel has designed a proprietary Online Health Test (OHT) for the ENT that can detect when:

1. Some value is consecutively repeated more times than expected, given the assessed entropy per sample of the source.
2. Some value becomes much more common in the sequence of noise source outputs than expected, given the assessed entropy per sample of the source.

After each reset, the OHT is automatically started and runs continuously until power-off or the next reset. A constant stream of 256-bit samples is outputted from the entropy source into the OHT where it tracks the entropy health.

The OHT is designed to have the same functionality and health coverage as the following health tests required by NIST SP 800-90B:

- Start-Up Tests
- Repetitive Counter Test (RCT)
- Adaptive Proportion Test (APT)

The OHT was designed to detect repeating patterns from the ENT. The OHT accomplishes this by continuously monitoring the statistical arrival rate of short bit patterns. As the bit patterns arrive, they are counted and then tested to see if the total pattern number lay within the expected binomial distribution. The final output from the ENT provides full entropy of 256 bits.

9. Guidance

9.1. Operator’s Guidance

The following security guidance for User role is described below:

- There is a difference between entering/exiting FIPS mode and enabling/disabling FIPS operations in BIOS. To enable FIPS operations, the module must first be configured as a FIPS 140-2 module from within the BIOS menu setting at power-on. The system will be reset in order to transition to/from FIPS operational mode after the setting is configured. If FIPS operations are not enabled within the BIOS settings, then the module is not a 140-2 validated module and cannot enter FIPS mode which means no FIPS services will be available.
- Once FIPS operations are enabled in the BIOS settings, the module is initialized as a FIPS 140-2 validated module following power-on. This means the module will now implicitly be in FIPS mode if the user is requesting a FIPS service. And the module is implicitly not in FIPS mode if the user is requesting a non-FIPS service. The operator of the module can call the API function crypto_drv_fips_mode_status() to check if the module is an FIPS 140-2 validated module. If API call returns 1 if the module is an FIPS 140-2 validated module; it returns 0 if the module is not an FIPS 140-2 validated module.
- When the module switches between FIPS and non-FIPS mode or vice versa the following must be considered by the user:
  - The DRBG engine must be reseeded.
  - The CSPs and keys shall not be shared between the modes.

The following security guidance for the User role is described below:

- The security guidance of the PBKDFv2 function is described below:
  - The length of the derived keying material shall be at least 112 bits long.
  - The length of the salt generated by an Approved DRBG shall be at least 128 bits long.
  - The iteration count shall be selected as large as possible, at least 1000 is recommended.
  - The password of 10 characters is considered the minimum password length. This provides a probability of randomly guessing the password as 1/10^10 (given digits 0-9 as the worst-case scenario, although lowercase and uppercase letters are also possible character classes).
  - Easily accessed personal information shall not be used directly as a password.
- One RSA key pair shall be used for one cryptographic purpose, such as digital signature and key transport. New RSA key pair is required for different cryptographic purpose.

9.2. Delivery Procedure

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 are able to download this firmware.

The module is contained within one of the platforms listed in Table 2. These Intel platforms are a tightly coupled component of 10th 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.

10. Mitigation of Other Attacks

The module provides mechanism to protect against the RSA timing attack. The OCS's big number arithmetic is able to 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 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 ECC has protection against known Differential Power Attack (DPA) which 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).

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
API	Application Program Interface
CAVS	Cryptographic Algorithm Validation System
CBC	Cipher Block Chaining
CMVP	Cryptographic Module Validation Program
COTS	Commercial Off The Shelf
CRNGT	Continuous Random Number Generator Test
CSE	Converged Security Engine
CSP	Critical Security Parameter
CTR	Counter Mode
DMA	Direct Memory Access
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
FIPS	Federal Information Processing Standards Publication
HMAC	Hash Message Authentication Code
IG	Implementation Guidance
IOSF	Intel® On-Chip System Fabric
KAS	Key Agreement Schema
KAT	Known Answer Test
MAC	Message Authentication Code
ME	Management Engine
MMIO	Memory-Mapped I/O
NIST	National Institute of Science and Technology
OAEP	Optimal Asymmetric Encryption Padding
OEM	Original Equipment Manufacturers
PBKDF	Password-Based Key Derivation Function
PCH	Platform Controller Hub
PCT	Pair-wise Consistency Test
PSS	Probabilistic Signature Scheme
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SHA	Secure Hash Algorithm
SKS	Secure Key Storage
SKU	Stock Keeping Unit

Appendix B. References

FIPS140-2 **FIPS PUB 140-2 - Security Requirements For Cryptographic Modules**
May 2001
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

FIPS140-2_IG **Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program**
September 15, 2015
http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf

FIPS180-4 **Secure Hash Standard (SHS)**
March 2012
http://csrc.nist.gov/publications/fips/fips180-4/fips 180-4.pdf

FIPS186-4 **Digital Signature Standard (DSS)**
July 2013
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

FIPS197 **Advanced Encryption Standard**
November 2001
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

FIPS198-1 **The Keyed Hash Message Authentication Code (HMAC)**
July 2008
http://csrc.nist.gov/publications/fips/fips198 1/FIPS-198 1_final.pdf

PKCS#1 **Public Key Cryptography Standards (PKCS) #1: RSA Cryptography**
Specifications Version 2.1
February 2003
http://www.ietf.org/rfc/rfc3447.txt

SP800-38A **NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques**
December 2001
http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

SP800-56ARev3 **NIST Special Publication 800-56A Revision 3 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**
April 2018
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf

SP800-56B **NIST Special Publication 800-56B Revision 1 - Recommendation for Pair Wise Key Establishment Using Integer Factorization Cryptography**
September 2014
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br1.pdf

SP800-90A **NIST Special Publication 800-90A Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators**
June 2015
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf

SP800-90B **NIST Draft Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation**
August 2012
http://csrc.nist.gov/publications/drafts/800-90/draft-sp800-90b.pdf

SP800-131A **NIST Special Publication 800-131A - Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths**
January 2011
http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf

SP800-132 **NIST Special Publication 800-132 - Recommendation for Password-Based Key Derivation - Part 1: Storage Applications**
December 2010
http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf

SP800-133 **NIST Special Publication 800-133 - Recommendation for Cryptographic Key Generation**
December 2012
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133.pdf