Cryptographic Module for Intel® vPro™ Platforms' Security Engine Chipset

Version 1.0

FIPS 140-2 Non-Proprietary Security Policy

Doc Version 1.2

Last update: 2016-07-18

Prepared by:
atsec information security corporation
9130 Jollyville Road, Suite 260
Austin, TX 78759
www.atsec.com

Contents

Copyrights and Trademarks

1. Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for version 1.0 of the Cryptographic Module for Intel® vPro™ Platforms' Security Engine Chipset. 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.

2. Cryptographic Module Specification

2.1. Module Overview

The Cryptographic Module for Intel® vPro™ Platforms' Security Engine Chipset (hereafter referred to as “the module”) is classified as a multiple-chip standalone firmware-hybrid module for FIPS 140-2 purpose. The Security Engine Chipset consists of both hardware and firmware. The hardware portion is the Converged Security Engine (CSE) and the firmware portion is the crypto driver process of the Management Engine (ME). The two portions form the logical cryptographic boundary and they combine to perform cryptographic functions within the Intel® vPro™ for applications executing on ME. The module is embodied in the Intel Platform Controller Hub (PCH) chipset shown in Figure 1 below.

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

Component	Type	Version Number	Description
Management Engine (ME) Crypto Driver	Firmware	1.0	It is a firmware running in an internal customized proprietary O/S to communicate with the hardware components of the module in the Intel PCH chipset. It includes the Triple-DES, SHA-224, SHA-384, SHA-512, HMAC, RSA, ECDSA, DRBG and EC Diffie-Hellman implementations.
Converged Security Engine (CSE)	Hardware	3.0	It includes AES, SHA-1 and SHA-256 security engines, and big number arithmetic support for implementing asymmetric algorithms in firmware. It is embedded within the Intel PCH chipset.
fips_sig	File	N/A	It is a file located in the file system of the internal customized proprietary O/S to contain the HMAC-SHA-256 hash value for integrity check of the module.

Table 1 - Cryptographic Module Components

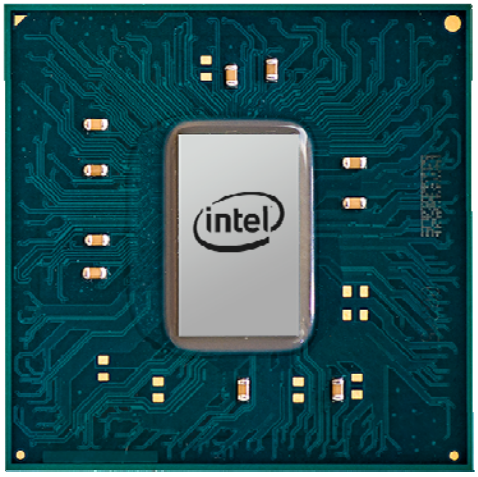

Figure 1 – Intel PCH chipset

The module has been tested on the following multichip standalone platform:

Platform	Processor	Operating System
Intel Sunrise Point PCH chipset	Embedded CPU with x86 instruction set (IA-32) executing the ME device firmware version 11.6.0.1102 CORPORATE SKU.	Embedded customized proprietary O/S dedicated to support the functionality of the ME.

Table 2 - 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.

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 3 - Security Levels

2.2. 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 multi-chip standalone cryptographic module.

The module provides cryptographic services to applications through an application program interface (API). The cryptographic logical boundary consists of the firmware component (i.e., ME Crypto Driver), the files for integrity check, and the hardware components (i.e., CSE). The logical block diagrams below shows the module, its interfaces with the operational environment and the delimitation of its logical boundary which are colored in **BLUE**:

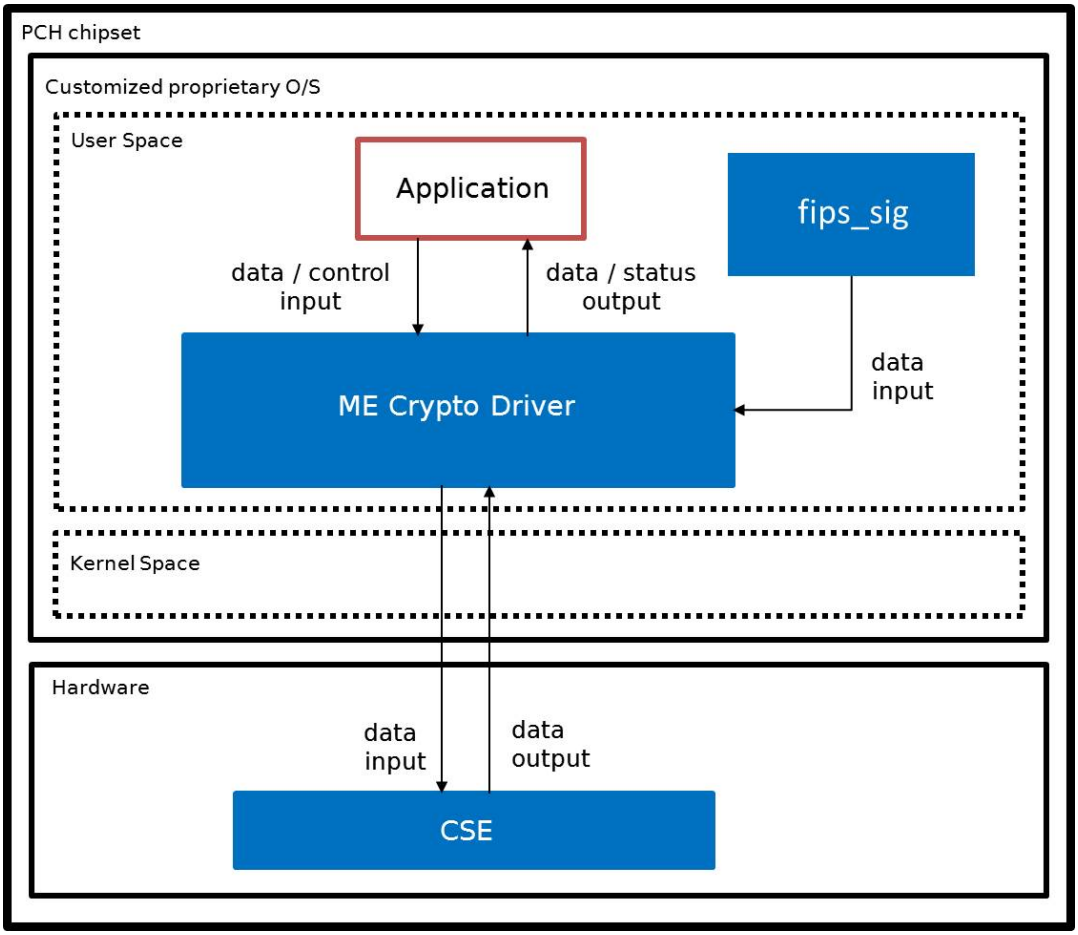

Figure 2 - Logical Block Diagram

2.3. 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.
- In "non-FIPS mode" (the non-Approved mode of operation) only non-approved security functions can be used.

When the module is power-up, the kernel of the operating environment obtains the HMAC value of the module for integrity check from the fips_sig file, and sends the HMAC value to the Crypto Driver. The module enters FIPS mode automatically after power-up tests succeed. If the module fails any power-up self-test, it will enter Error state and trigger ME reset to rerun the module and perform the power-up self-tests again. Once the module is operational, the mode of operation is implicitly assumed depending on the security function invoked and the security strength of the cryptographic keys.

Critical security parameters used or stored in FIPS mode are not used in non-FIPS mode, and vice versa.

The module supports the following Approved cryptographic algorithms:

Algorithms	Components	Standards	CAVS Certificates
3-key Triples-DES encryption and decryption with 168 bits key size and the following mode: • ECB • CBC • Counter (CTR)	ME Crypto Driver	SP 800-67, SP 800-38A	#2152
AES encryption and decryption with 128 and 256 bits key size and the following mode: • ECB • CBC • Counter (CTR)	CSE	FIPS 197, SP800-38A	#3923
SHA: • SHA-1 • SHA-256	CSE	FIPS 180-4	#3233
SHA: • SHA-224 • SHA-384 • SHA-512	ME Crypto Driver		#3232
HMAC with: • SHA-1 • SHA-256	ME Crypto Driver using CSE hash support	FIPS 198-1	#2548
HMAC with: • SHA-224 • SHA-384 • SHA-512	ME Crypto Driver		#2547
DRBG using AES-128 CTR_DRBG without Derivation Function and without Prediction Resistance	ME Crypto Driver	SP 800-90A	#1156

Algorithms	Components	Standards	CAVS Certificates
RSA Signature Generation based on PKCS#1 v1.5 with 2048 and 3072 bits modulus size and: • SHA-384 • SHA-512 RSA Signature Generation based on PSS with 2048 and 3072 bits modulus size and: • SHA-224 • SHA-384 • SHA-512 **Note**: Only the modulus size 2048 is supported by the CSE big number arithmetic.	ME Crypto Driver using CSE big number arithmetic	FIPS 186-4	#2022
RSA Signature Verification based on PKCS#1 v1.5 with 1024, 2048, and 3072 bits modulus size and: • SHA-384 • SHA-512 RSA Signature Verification based on PSS with 1024, 2048, and 3072 bits modulus size and: • SHA-224 • SHA-384 • SHA-512 **Note**: Only the modulus size 1024 and 2048 are supported by the CSE big number arithmetic.			
RSA X9.31 Key Generation with 2048 bits modulus size		FIPS 186-4 Appendix B.3.3	
RSA Signature Generation based on PKCS#1 v1.5 and PSS with 2048 and 3072 bits modulus size and: • SHA-256 **Note**: Only the modulus size 2048 is supported by the CSE big number arithmetic.	ME Crypto Driver using CSE big number arithmetic and hash support	FIPS 186-4	#2003
RSA Signature Verification based on PKCS#1 v1.5 and PSS with 1024, 2048, and 3072 bits modulus size and: • SHA-1 • SHA-256 **Note**: Only the modulus size 1024 and 2048 are supported by the CSE big number arithmetic.			

Algorithms	Components	Standards	CAVS Certificates
Section 5.7.1.2 ECC cofactor Diffie-Hellman (ECC CDH) primitive with P-256 curve	ME Crypto Driver using CSE big number arithmetic	SP 800-56A	CVL #779
ECDSA Key Pair Generation and Public Key Verification with P-256 curve	ME Crypto Driver using CSE big number arithmetic	FIPS 186-4	#871
ECDSA Signature Verification with P-256 curve and: • SHA-224			
ECDSA Signature Generation with P-256 curve and: • SHA-224			CVL #798
ECDSA Signature Verification with P-256 curve and: • SHA-1 • SHA-256	ME Crypto Driver using CSE big number arithmetic and hash support		#872
ECDSA Signature Generation with P-256 curve and: • SHA-256			CVL #799
Password-Based Key Derivation version 2 (PBKDFv2) with HMAC-SHA-1 and HMAC-SHA-256	ME Crypto Driver using CSE hash support	SP 800-132	Vendor Affirmed
RSA-based Key Transport with OAEP and 2048 and 3072 bits modulus size	ME Crypto Driver	SP 800-56B	Vendor Affirmed

Table 4 – Validated Cryptographic Algorithms

Note: The module only uses the general purpose AES engine from the CSE for AES cryptographic operations (i.e., encryption and decryption). Any other AES engine from the hardware component of the module is considered as dead path since it is not callable via the interface of ME crypto driver.

The module implements the following non-Approved algorithms:

- MD5
- HMAC-MD5
- RC4
- DES
- 2-key Triple-DES
- RSA key generation, digital signature, key wrapping with non-compliant modulus size and/or non-Approved hashing
- RSA key wrapping with encryption and decryption primitives

Regarding to the services available in FIPS mode of operation and non-FIPS mode of operation, please refer to Table 6 - Services in FIPS mode of operation and Table 7 - Services in non-FIPS mode of operation in 4.2 Services.

3. 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., ME 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	Hardware Interface
Data Input	API input parameters pointing to data stored in RAM, fips_sig 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 5 - Ports and Interfaces

4. Roles, Services and Authentication

4.1. Roles

The module supports the following roles:

- **User role**: performs all services (in both FIPS mode and non-FIPS mode of operation), except 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.

4.2. 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
Symmetric encryption and decryption	User	AES 128 and 256 bit keys	Read
		3-key Triple-DES key	
RSA key 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
RSA-based Key Wrapping using RSA Encryption and Decryption Primitives (SP 800-56B Section 7.1)			
RSA key validation	User	RSA prime numbers and private key	Read
ECDSA key generation	User	ECDSA public-private keys with P-256 curve	Create
ECDSA public key verification			Read
ECDSA signature generation			Read
ECDSA signature verification			Read
Message digest generation	User	None	None
MAC generation and verification	User	At least 112 bits HMAC key	Read

Service	Role	Key/CSP	Access
Random Number Generation	User	DRBG seed (consisting of Entropy Input String), V and Key	Read, Update
Secure Key Storage (SKS) To load the 128 or 256 bits key directly from SKS in the register for AES or HMAC operations.	User	AES 128, 256 bit keys, HMAC 128, 256 bit keys, AES master key	Read, Write
Password-based Key Derivation Function with SHA-1 and SHA-256	User	Password, at least 112 bits derived keying material	Read, Create
EC Diffie-Hellman Key Agreement Primitive	User	ECDSA public-private keys with P-256 curve, shared secret	Read, Create
Show status	User	None	None
Self-Tests	User	According to IG 7.4, the keys only used to perform Power-Up Tests are not considered CSPs.	None
Zeroization	User	All CSPs	Zeroize
Module Initialization	Crypto Officer	None	None

Table 6 - Services in FIPS mode of operation

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

Service	Role		Access
Message digests using MD5	User	None	Create
MAC generation using MD5	User	HMAC key	Read
MAC generation using less than 112 bits HMAC key	User	Less than 112 bits HMAC key	Read
Symmetric encryption / decryption using RC4	User	40 to 2048 bits key	Read
Symmetric encryption / decryption using DES	User	56 bit keys	Read
Symmetric encryption / decryption using 2-key Triple-DES	User	112 bit keys	Read
RSA key generation with 1024 bits modulus size	User	RSA public-private keys with modulus size 1024 bits	Create
RSA signature generation, key wrapping with any modulus size rather than 2048, 3072 and 4096 bits	User	RSA public-private keys with any modulus size rather than 2048, 3072 and 4096 bits	Read
RSA signature generation with MD5 or SHA-1 for any modulus size	User	RSA public-private keys with any modulus size	Read

Table 7 - Services in non-FIPS mode of operation

4.3. Operator Authentication

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

5. Physical Security

The Cryptographic Module for Intel® vPro™ Platforms' Security Engine Chipset is a firmware-hybrid module that operates on a multi-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 must be used in a commercial off the shelf (COTS) computer device. The computer device shall be comprised of production grade components with standard passivation (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.

6. Operational Environment

6.1. Applicability

The module operates in a limited operational environment per FIPS 140-2 level 1 specifications. The module runs on an internal customized proprietary O/S within the Intel PCH chipset.

6.2. 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.

7. 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.	Called memset_secure() to replace zero in the memory.
Triple-DES keys	The key is passed into the module via API input parameters.	Stored as plaintext in the RAM.	Called memset_secure() to replace zero in the memory.
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.	Called memset_secure() to replace zero in the memory.
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.	Called memset_secure() to replace zero in 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.	Called memset_secure() to replace zero in the memory.
Shared Secret	The shared secret is generated in the EC Diffie-Hellman key agreement function.	Stored as plaintext in the RAM.	Called memset_secure() to replace zero in the memory.
Entropy Input String for DRBG seed	Obtained from hardware DRBG outside of the module’s logical 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.
Password	The password is passed into the module via API input parameters.	Stored as plaintext in the RAM.	Called memset_secure() to replace zero in the memory.
AES master key	The key is passed into the module via API input parameters.	Stored as plaintext in the SKS	Zeroized by replacing new values.

Table 8 - Life cycle of Keys and Critical Security Parameters (CSP)

The following sections describe how CSPs, in particular cryptographic keys, are managed during its life cycle.

7.1. Random Number Generation

The module employs a Deterministic Random Bit Generator (DRBG) based on [SP800-90A] for the creation of asymmetric keys, and for providing a Random Number Generation service to calling applications.

The module implements the CTR_DRBG with AES-128 without derivation function and without prediction resistance. The CTR_DRBG is implemented in the firmware (i.e., ME Crypto Driver) and provides between 128 and 65536 bits of output data per each request.

The module uses the output of the hardware DRBG as the entropy source for seeding the CTR_DRBG. The hardware DRBG is implemented outside of the module's logical boundary within the Intel PCH chipset physical boundary. The module collects 256 bits of data from the hardware DRBG for generating the initial seed during initialization of the CTR_DRBG, and reseeding which occurs less than 2^{48} times of DRBG service request.

The module performs Repetition Count Test (RCT) on the output of the hardware DRBG to ensure that the hardware DRBG is not degraded.

7.2. Key Generation

For generating RSA and ECDSA keys the module implements asymmetric key generation services compliant with [SP800-133] and [SP800-90A].

The module does not offer a dedicated service for generating keys for symmetric algorithms or for HMAC. However, the module offers a DRBG compliant to [SP800-90A] to allow a caller to obtain random numbers which can be used as key material for symmetric algorithms or HMAC.

7.3. Key Establishment/Key Derivation

The module supports the [SP800-56A] EC Diffie-Hellman Key Agreement primitive with P-256 curve. The caller may use the module's DRBG to generate an EC Diffie-Hellman private key.

The module also supports RSA key wrapping using encryption and decryption primitives with the modulus size of 2048, 3072 and 4096 bits in FIPS mode, as well as 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 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.

7.4. 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.

7.5. 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 256 bits master 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 ME Crypto Driver or calling application.

The HMAC key used for integrity test is stored in the fips_sig file and relies on the operating system for protection.

7.6. 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., ME 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-Map I/O (called “MMIO” in short) registers are accessible from ME Crypto Driver to store the 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 ME 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.

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 by calling the ME 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_sig 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 ECB, encrypt • KAT AES ECB, decrypt
Triple DES	• KAT Triple-DES ECB, encrypt • KAT Triple-DES ECB, decrypt
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
HMAC	• KAT HMAC-SHA-1 and HMAC-SHA-512
ECDSA	• PCT ECDSA (NIST P-256) signature generation • PCT ECDSA (NIST P-256) signature verification
RSA	• KAT RSA 2048-bit key PKCS#1 v1.5 with SHA-256 signature generation • KAT RSA 2048-bit key PKCS#1 v1.5 with SHA-256 signature verification
DRBG	• KAT CTR_DRBG

Algorithm	Power-Up Tests
ECC Diffie-Hellman	Primitive “Z” Computation KAT: • KAT for ECC Cofactor Diffie-Hellman primitive (NIST P-256)

Table 9- Self-Tests

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 is 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 is invoked by powering-off and reloading the module which cause the module to run the power-up tests again. 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 Repetition Count Test (RCT), shown in the following table:

Algorithm	Test
ECDSA key generation	• PCT signature generation and verification
RSA key generation	• PCT signature generation and verification
DRBG	• CRNGT is not required per IG 9.8
DRBG seeding	• RCT is allowed per IG 9.8

Table 10 - Conditional Tests

9. Guidance

9.1. Operator's Guidance

The following security guidance for User role is described below:

- 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 operator of the module can call the API call of 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.
- Once the operator enables FIPS in the configuration settings of the operating environment, the module is initialized as an FIPS 140-2 validated module. The operator cannot disable FIPS unless the module is uninstalled and reinstalled by the factory.
- The security guidance of the PBKDFv2 function are 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 100 is recommended;
  - The password shorter than 10 characters are usually considered to be weak;
  - 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 ME 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 hardware component of the module is contained within the Intel Sunrise Point Platform Controller Hub (PCH). The Intel Sunrise Point PCH is a tightly coupled component of 6th Generation Intel® Core™ and 6th Generation Intel® Core™ vPro™ platforms. The Intel Sunrise Point PCH 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 vPro™ brand systems to meet the brand validation requirements and testing to ensure the vPro™ systems have been designed and constructed with the proper components including CPU and Intel Sunrise Point PCH. Intel's brand validation tool would detect any mismatch of CPU and PCH for any vPro™ 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 with a vPro™ badge had been designed and tested to conform to Intel's requirements and

6th generation vPro™ systems will always have the Intel Sunrise Point PCH that contains the module.

10. Mitigation of Other Attacks

The module provides mechanism to against the RSA timing attack.

The CSE'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 CSE'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 ME Crypto Driver takes advantage of this feature by enabling the aforementioned functionality in the CSE for private key operations. This implies that the computation time using the private key is constant, hence mitigating timing attacks.

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
CVL	Component Validation List
DES	Data Encryption Standard
DMA	Direct Memory Access
DSA	Digital Signature Algorithm
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
O/S	Operating System
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

SP800-56A **NIST Special Publication 800-56A Revision 2 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**
May 2013
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800 56Ar2.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

SP800-67 **NIST Special Publication 800-67 Revision 1 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
January 2012
http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.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