STMICROELECTRONICS

Trusted Platform Module ST33KTPM2XSPI / ST33KTPM2XI2C

FIPS 140-3 Non-Proprietary Security Policy Level 1

Date: 2024-03-06
Document Version: 01-02

NON-PROPRIETARY DOCUMENT

Table of Contents

1 GENERAL

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for the STMicroelectronics Trusted Platform Module ST33KTPM2XSPI / ST33KTPM2XI2C. It details how the module meets the requirements specified in [FIPS 140-3] for a Security Level1 module.

1.2 Security levels

Next table indicates the security levels reached by the security module.

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

Table 1 - Security Levels

2 CRYPTOGRAPHIC MODULE SPECIFICATION

ST33KTPM2XSPI / ST33KTPM2XI2C is a fully integrated security module implementing the revision 1.59 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM) version 2.0. It is designed to be integrated into personal computers and any other embedded electronic systems. TPM is primarily used for cryptographic keys generation, keys storage, keys management and secure storage for digital certificates.

The security module is a single chip cryptographic HW module as defined in **[FIPS 140-3]**. The single silicon chip is encapsulated in a hard, opaque, production grade integrated circuit (IC) package.

The cryptographic boundary is defined as the perimeter of the IC package. The security module supports both SPI and I^{2}C interfaces, compliant with the PC Client specification **[PTP 1.05]**. The HW and FW cryptographic boundaries are indicated in Figure 2 and Figure 4 of the current document.

2.1 Operating Environments

2.1.1 Module identification parameters

The operating environments covered by the FIPS 140-3 evaluation are summarized in the table below:

Model	Hardware [Part Number and Version]	Firmware Version	Distinguishing Features
ST33KTPM2XSPI	ST33K1M5T revC	9.256 (dec.) 0x00.09.01.00 (hex.)	SPI
ST33KTPM2XI2C			SPI or I^{2}C[1]

Table 2 - Cryptographic Module Tested Configuration

FW version can be read in the response to the command TPM2_GetCapability with property set to TPM_PT_FIRMWARE_VERSION_1.

The product is manufactured in one single package:

- UFQFPN32
  - Ultra-thin pitch Quad Flat No-lead 32-pin
  - 5 x 5 mm

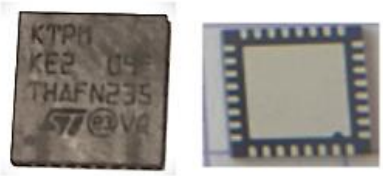

Figure 1 - UFQFPN32 package

2.1.2 Configurations

The security module is available in the configurations listed hereafter.

2.1.2.1 KE2

The current FIPS 140-3 level 1 security policy always applies (no mode lock requested) to this security module configuration.

[1] The interface is dynamically selected

	Module configuration
Module name / HW P/N	ST33KTPM2XSPI
Package	UFQFPN32
Interface	SPI
Marking	KE2
FW version	00.09.01.00 (9.256)
TPM2.0 revision	1.59
Libraries version	07.01.00.00 (HWINTF library) 05.01.00.00 (TPM2.0 library)

Table 3 - KE2 security module configuration

2.1.2.2 KE3

The current FIPS 140-3 level 1 security policy always applies (no mode lock requested) to this security module configuration. SPI or I^{2}C mode selection is done during the boot of the security module.

	Module configuration
Module name / HW P/N	ST33KTPM2XI2C
Package	UFQFPN 32
Interface	SPI / I^{2}C
Marking	KE3
FW version	00.09.01.00 (9.256)
TPM2.0 revision	1.59
Libraries version	07.01.00.00 (HWINTF library) 05.01.00.00 (TPM2.0 library)

Table 4 - KE3 security module configuration

2.2 Security functions

The security module supports the following cryptographic algorithms (both approved and non-approved). Algorithm certificate numbers for each approved algorithm are listed below. All algorithms, keys size or curve lengths listed below are part of services offered by the module.

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A2553	AES [SP 800-38A]	ECB, CFB128, OFB, CBC, CTR	128, 192, 256	Data encryption/decryption
A2547	DRBG [SP 800-90A]	HASH_based SHA2-256		Deterministic random bit generation
A2555	ECDSA [FIPS 186-4]	SHA2-256, SHA2-384, SHA3-256, SHA3-384	P-256, P-384	Digital signature generation
		SHA-1, SHA2-256, SHA2-384, SHA3-256, SHA3-384	P-256, P-384	Digital signature verification
		ECDSA KeyVer (FIPS 186-4)	P-256, P-384	Key verification
		Appendix B.4.1	P-256, P-384	Key generation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
-	ENT (P) [SP800-90B]			Entropy source[1]
A2551 **A2552**	HMAC [FIPS 198-1]	SHA-1, SHA2-256, SHA2-384, SHA3-256, SHA3-384	160, 256, 384	Message authentication
A2555	KAS [SP 800-56A Rev3][2] [SP 800-56C Rev1]	ECC (Full unified and One pass DH)	P-256, P-384	Key agreement scheme
A2550	KBKDF [SP 800-108]	CTR		Key derivation (based on HMAC)
A2554	KTS-IFC [SP800-56B Rev 2]	KTS-OAEP-basic	2048, 3072, 4096	Key generation and key transport
		RSADP Component (IG 2.4.B)	2048	Decryption primitive
A2554	RSA [FIPS 186-4]	SHA2-256, SHA2-384, RSASSA-PKCS-v1.5, RSASSA-PSS	2048, 3072, 4096	Digital signature generation
		SHA-1[3], SHA2-256, SHA2-384, RSASSA-PKCS-v1.5, RSASSA-PSS	1024[4], 2048, 3072, 4096	Digital signature verification
		Appendix C3.1	2048, 3072, 4096	Key generation
A2548	SHA3-256, SHA3-384 [FIPS 202]	SHA3-256, SHA3-384		Message digest
A2548 **A2549**	SHS [FIPS 180-4]	SHA-1, SHA2-256, SHA2-384		Message digest. SHA2-256 is also used as SP800-90B vetted conditioner

Table 5 - Approved Algorithms

Algorithm	Caveat	Use / Function
CKG [IG D.H]	Direct Generation of Symmetric Keys (Section 4 of [SP800-133 Rev2]).	Key generation[5]
RSA [FIPS 186-4]	Use of SHA3-256 or SHA3-384 hashing algorithms.	Digital signature generation
		Digital signature verification

Table 6 - Vendor Affirmed Approved Algorithms

[1] Seed or reseed SP800-90A approved DRBG with a minimum of 414 bits of entropy. Generate random numbers not dedicated to being used as cryptographic material.

[2] Per [IG] D.F Scenario 2 path (2), [56Ar3] compliant key agreement scheme where testing is performed end-to-end for the shared secret computation and a KDF compliant with oneStepKdf [56Cr1] without key confirmation.

[3] Legacy use only

[4] Legacy use only

[5] Symmetric keys and seeds used for generating the asymmetric keys are either generated by using KBKDF or DRBG methods. Methods are detailed per SSPs in Table 19 and Table 20.

Algorithm	Caveat	Use/Function
AES CFB	The AES CFB algorithm itself is Approved and awarded CAVP Cert. #A2553, but this usage employs a key that is non-compliant. The usage of AES CFB in this manner is entirely internal to the module and inaccessible to the operator. No security claimed per IG 2.4.A, Example Scenario #1.	Obfuscation of internally stored data
XOR	No security claimed per IG 2.4.A, Example Scenario #1.	Obfuscation of input or output data

Table 7 - Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed

Algorithm/Function	Use/Function
ECC BN P-256	Key generation, digital signature generation based on BN P-256 elliptic curve
ECC derived keys	Secret exchange or digital signature generation/verification
ECDAA	Key generation, digital signature generation
ECSchnorr	Key generation, digital signature generation and verification
HMAC	Key length < 112 bits for message authentication
RSA	1024-bit RSA digital signature generation
RSA with no padding mode (null scheme)	Key transport
RSAES-PKCS1-v1_5	Key transport
SHA-1	Digital signature generation

Table 8 - Non-approved Algorithms not Allowed in the Approved Mode of Operation

Name	Type	Description	SF Properties [O]	Algorithms	Algorithm Properties
KAS	KAS	Key establishment	SP 800-56A, Rev 3 Key length 128 bits IG D.F	KAS-ECC (Initiator, Responder), KPG, Full (Cert. #A2555)	P-256, P-384 fullUnified, onePassDH oneStepKDF
KTS	KTS	Key Transport	SP 800-38F IG D.G SSP establishment methodology provides 128 or 256 bits of encryption strength	KTS (AES Cert. #A2553 + HMAC Cert. #2551)	AES CFB Key size 128 or 256 bits.
KTS-RSA	KTS	Key Transport	SP 800-56B Rev 2 IG D.G KTS-OAEP-basic SSP establishment methodology provides between 112 and 150 bits of encryption strength	KTS-IFC (Cert. #A2554)	Key size 2048, 3072, or 4096

Table 9 - Security Function Implementations

2.3 Cryptographic boundary

A block diagram of the security module with its associated cryptographic boundary is provided in Figure 2.

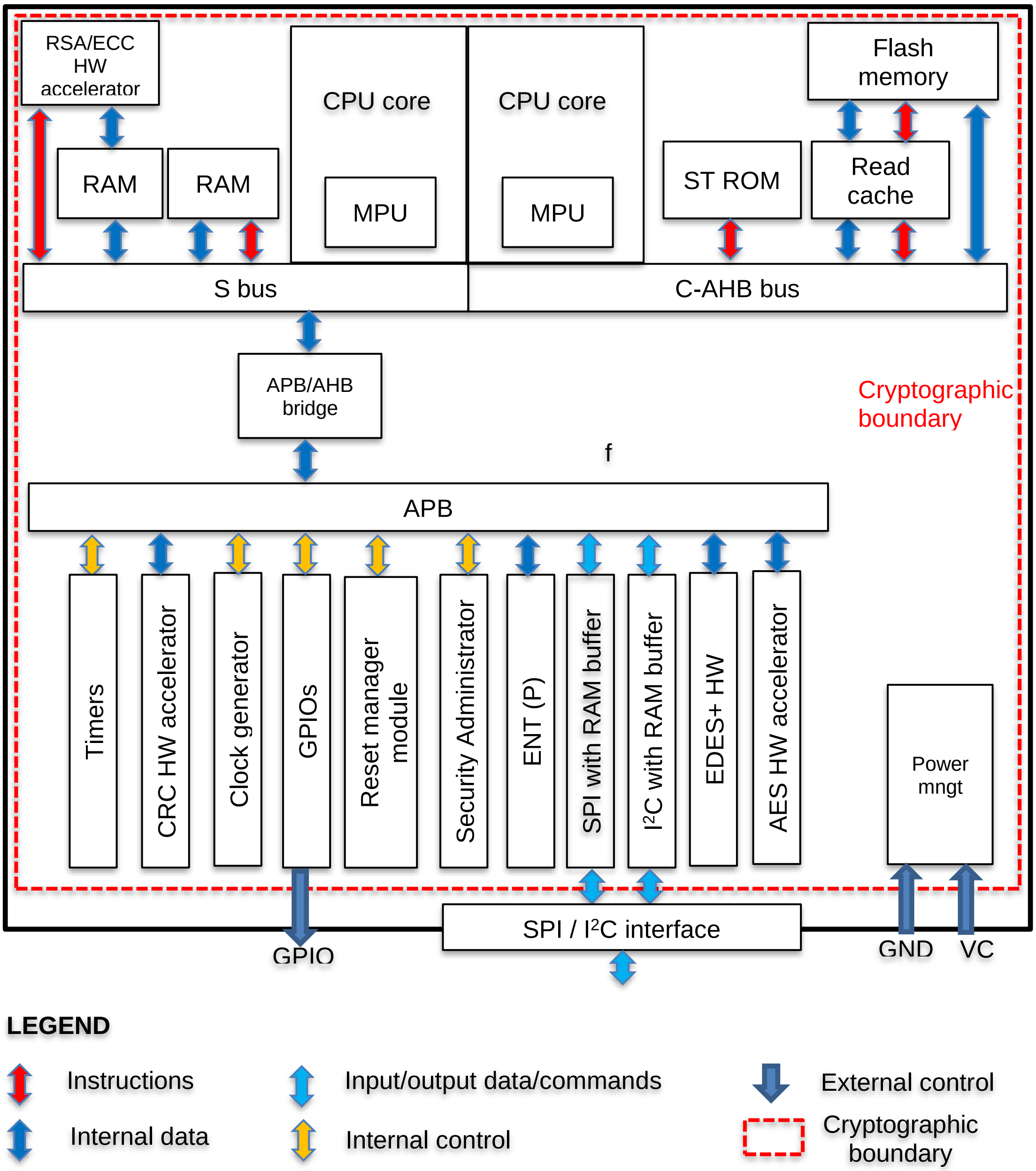

Figure 2 - HW block diagram

Module is composed of:

- Two CPU cores, each including a MPU (Memory Protection Unit)
- Memories (RAMs, Flash and ROM) that store data or FW
- HW accelerators for CRC (16 and 32-bits), symmetric cryptographic operations (AES) and asymmetric cryptographic operations (RSA/ECC)
- A clock generator and timers
- ENT (P)
- SPI and I^{2}C[1] master/slave blocks
- An administration block dedicated to chip security configuration and alarms detection

[1] I^{2}C block is not used by the ST33KTPM2XSPI module configuration

2.4 Overall security design

1. The Module provides one operator role: the Cryptographic Officer.
2. The Module, evaluated at FIPS 140-3 Level 1, does not claim to provide authentication.
3. The Module allows the operator to initiate power-up self-tests by power cycling or resetting the Module.
4. Power up self-tests do not require any operator action.
5. Data output is inhibited during key generation, self-tests, zeroization, firmware loading, and error states.
6. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the Module.
7. The Module does not support concurrent operators.
8. The Module does not support a maintenance interface or role.
9. The Module does not support manual key entry method.
10. The Module does not have any proprietary external input/output devices used for entry/output of data.
11. The Module does not output intermediate key values.
12. The Module does not provide bypass services or ports/interfaces.

3 CRYPTOGRAPHIC MODULE INTERFACES

3.1 Pinout description

The pin layouts for the ST33KTPM2XSPI / ST33KTPM2XI2C with the UFQFPN32 package in Figure 3. The security module supports both SPI and I²C physical interfaces but only one interface is configured during TPM boot. The interface configured remains active until the next module reset.

3.1.1 *UFQFPN32 configuration*

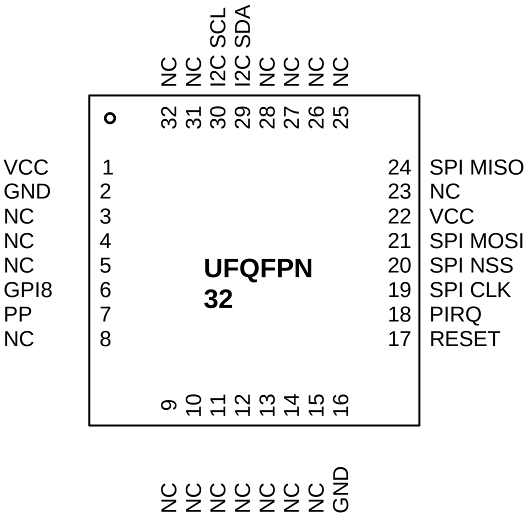

Figure 3 - UFQFPN32 Pinout Diagram

Next table gives a description of the products pins.

Signal	Type	Description
VCC	Input	**Power supply**. This pin must be connected to 1.8V or 3.3V DC power rail supplied by the motherboard.
GND	Input	GND has to be connected to the main motherboard ground.
RESET	Input	Reset used to re-initialize the device
I2C SCL / GPIO5	Input or Input/Output	I²C serial clock (Open drain with no weak pull-up resistor) or GPIO if SPI interface is selected
I2C SDA / GPIO6	Input/Output	I²C serial data (Open drain with no weak pull-up resistor) or GPIO if SPI interface is selected
PIRQ	Output	IRQ used by TPM to generate an interrupt
SPI CLK / GPIO1	Input or Input/Output	SPI serial clock (output from master) or GPIO if I²C interface is selected
SPI NSS / GPIO2	Input or Input/Output	SPI slave select (active low; output from master) or GPIO if I²C interface is selected
SPI MISO / GPIO0	Output or Input/Output	SPI Master Input, Slave Output (output from slave) or GPIO if I²C interface is selected
SPI MOSI / GPIO3	Input or Input/Output	SPI Master Output, Slave Input (output from master) or GPIO if I²C interface is selected
GPI8	Input	GPI default to low. The level of this pin on the rising edge of the RESET signal is used to determine the physical interface to use (high level corresponds to SPI configuration and low-level to I²C)
PP	Input	**Physical presence**, active high, internal pull-down. Used to indicate Physical Presence to the TPM.
NC	-	**Not Connected:** connected to the die but not usable. May be left unconnected. Internal pull-down.

Table 10 - UFQFPN32 pins definition

3.2 Ports and interfaces

The physical port of the security module is the SPI bus or I^{2}C Bus. The logical interfaces and their mapping to physical ports of the module are described below:

Physical port	Logical interface	Data that passes over the port/interface
SPI_NSS / SPI_CLK / SPI_MOSI / RESET / PP	Control input interface	Control parts of the TPM commands provided to the security module. It concerns all bytes of a command except plaintext data, ciphertext data and SSPs (entered with the data input interface).
I2C_SCL / I2C_SDA / RESET / PP		
SPI_NSS / SPI_CLK / SPI_MISO / PIRQ	Control output interface	Control parts of the TPM responses output by the security module. It concerns all bytes of a response except plaintext data, ciphertext data and SSPs (output with the data output interface) and except the responseCode of a response (output with the status output interface)
I2C_SCL / I2C_SDA / PIRQ		
SPI_NSS / SPI_CLK / SPI_MISO / PIRQ	Status output interface	Status output by the security module (responseCode parameter of a response)
I2C_SCL / I2C_SDA		
SPI_NSS / SPI_CLK / SPI_MOSI	Data input interface	Data (plaintext data, ciphertext data and SSPs) provided to the security module as part of an input processing command.
I2C_SCL / I2C_SDA		
SPI_NSS / SPI_CLK / SPI_MISO	Data output interface	Data (plaintext data, ciphertext data and SSPs) output by the security module as part of the response to a processing command.
I2C_SCL / I2C_SDA		
VCC / GND	Power interface	Power interface of the security module

Table 11 - Ports and Interfaces

Here are some details concerning the ports and interfaces of TPM:

1. Control and data inputs are multiplexed over the same physical interface. Control and data are distinguished by properly parsing input TPM command parameters according to input structures description, indicated for each command in **[TPM2.0 Part3]**[1].
2. Status, data and control output are multiplexed over the same physical interface. Status, data and control are distinguished by properly setting output TPM response parameters according to output structures description, indicated for each command in **[TPM2.0 Part3]**.
3. The logical state machine and the command structure parsing of the module prevent from using input data externally from the “data input path” and prevent from outputting data externally from the “data output path”.
4. While performing key generation or key zeroization (no manual key entry on TPM), the output data path is logically disconnected while the output status path remains connected to report any possible failure during command processing. Generally, the output data path is only connected when TPM outputs response containing data.
5. To prevent the inadvertent output of CSPs in plaintext form on TPM2_Duplicate, the two following independent internal actions are performed:
    a. Verification of the encryptedDuplication attribute of the key to be duplicated
    b. Verification of the handle of the new parent of the key to be duplicated

    encryptedDuplication attribute must be set to 0 and new handle must be set to the null handle to authorize outputting the private part of the key in plaintext form.
6. The logical state machine and command structure of the module guarantees the inhibition of all data output via the data output interface whenever an error state exists and while doing self-tests.

The status output interface remains active during the error state to output the status of the security module with the service TPM2_GetCapability and TPM2_GetTestResult.

[1] Some commands only deal with control input and status output parameters

4 ROLES, SERVICES AND AUTHENTICATION

This chapter gives details about the roles managed by TPM.

4.1 Roles

Services proposed by TPM are accessible under the roles defined in the table below. The list of services accessible by each role is indicated in Table 14.

Role	Service	Input	Output
Crypto officer (CO)	This role performs the cryptographic initialization of the security module and executes the management functions. This role also covers the use of the general security services provided by the cryptographic module.	Any valid inputs and outputs for commands are usable (refer to **[TPM2.0 Part3]**).	

Table 12 - Roles, Service Commands, Input and Output

The security module does not provide a maintenance role or maintenance interface and does not support concurrent operators. The CO role is implicitly selected by the TPM operator on service execution.

4.2 Authentication

In the context of this FIPS 140-3 Level 1 evaluation, there is no authentication mechanism claimed to control access of the security module. The authorization mechanisms (password, HMAC and policy) provided by the TPM2.0 standard are available and protected as sensitive parameters but are not employed to satisfy FIPS 140-3 requirements. Crypto officer role is implicitly assumed by the operator when using services corresponding to that role.

4.3 Services

All services are accessible under the roles defined in Table 12and no specific access rights are considered to operate with keys and SSPs. Full services inputs and outputs are defined in **[TPM2.0 Part3]**. Next table indicates how mandatory services required in §7.4.3.1 of **[ISO/IEC 19790]** are mapped to security module's services:

Mandatory service requested from [ISO/IEC 19790]	Corresponding services from the security module
Show module's versioning information	TPM2_GetCapability
Show status	TPM2_GetTestResult
Perform self-tests	TPM2_SelfTest TPM2_IncrementalSelfTest
Perform approved security functions	See approved services listed in Table 14
Perform zeroization	TPM2_Clear, TPM2_ChangePPS, TPM2_ChangeEPS, TPM2_FlushContext, TPM2_EvictControl

Table 13 - Mapping between services

The security module does not implement any bypass capability, nor self-initiated cryptographic output capability.

4.3.1 Approved services list

Next table lists all approved services supported by the TPM. The indicator is accessible with the TPM2_GetCapability (capability = TPM_CAP_VENDOR_PROPERTIES) command by using the sub-capability TPM_SUBCAP_VENDOR_TPMA_MODES = 0x7.

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs[1]	Indicator[2]
TPM2_Startup	Set-up the TPM after a power cycle.	None	ppSeed, epSeed, spSeed, phProof, ehProof, shProof, drbgState	CO	G	Approved
			nullSeed, nullProof, contextKey, drbgSeed		G, Z	
TPM2_Shutdown (I)	Prepare the TPM for a power cycle.	None	None	CO	N/A	Non-security relevant
TPM2_SelfTest (I)	Self-tests execution	SHS, SHA3, ENT, HMAC, AES, DRBG, KBKDF, KAS, RSA (signature generation, verification) ECC (signature generation, verification)	None	CO	N/A	Approved
TPM2_IncrementalSelfTest (I)	Incremental self-tests execution	SHS, SHA3, ENT, HMAC, AES, DRBG, KBKDF, KAS, RSA (signature generation, verification), ECC (signature generation, verification)	None	CO	N/A	Approved
TPM2_GetTestResult (I)	Get self-tests result	None	None	CO	N/A	Non-security relevant
TPM2_StartAuthSession (I/E/D)	Session command	SHS, SHA3, HMAC, AES, DRBG, KBKDF, KTS-RSA, KAS, KDA, CKG	sesHmacKey, sesSymKey	CO	G, W	Approved
			sesSalt		E, Z	
			objSens, objAuth, nvAuth, platformAuth, endorsementAuth, ownerAuth, lockoutAuth, seqAuth		E	
TPM2_PolicyRestart (I)	Policy session restart	None	None	CO	N/A	Non-security relevant
TPM2_Create (I/E/D)	Object creation		objSeed, objSens, objPub	CO	G, R, E	Approved

[1] **G** = generate, **R** = read, **W** = write, **E** = execute, **Z** = zeroize

[2] Approved, non-approved or non-security relevant.

		SHS, SHA3, HMAC, AES, DRBG, KBKDF, CKG, RSA (signature generation, verification, key generation), ECC (signature generation, verification, key generation)	objSymKey, objHmacKey		G, E	
			drbgState		W, E	
			objAuth		W	
			nullProof, phProof, ehProof, shProof		E	
TPM2_Load (I/E/D)	Object loading	SHS, SHA3, HMAC, AES, KBKDF	objSens, objSeed	CO	W, E	Approved
			objPub, objAuth		W	
			objSymKey, objHmacKey		G, W, E	
TPM2_LoadExternal (I/E/D)	External object loading	None	objPub, objSens, objAuth	CO	W	Approved
TPM2_ReadPublic (I)	Read public part of a loaded object	None	objPub	CO	R	Approved
TPM2_ActivateCredential (I/E/D)	Enables the association of a credential with an object	SHS, SHA3, HMAC, AES, KBKDF, KTS-RSA, KAS, CKG	objSens	CO	E	Approved
			creSeed		E, Z	
			creSymKey, creHmacKey		G, E, Z	
TPM2_MakeCredential (I/E/D)	Allows the TPM to perform the actions required of a Certificate Authority	SHS, SHA3, HMAC, AES, KBKDF, KTS-RSA, KAS, CKG	objPub	CO	E	Approved
			creSeed		G, R, E, Z	
			creSymKey, creHmacKey		G, E, Z	
TPM2_Unseal (I/E/D)	Returns the data in a loaded Sealed Data Object	None	objSens	CO	R	Approved
TPM2_ObjectChangeAuth (I/E/D)	Changes the authorization secret for a TPM-resident object	SHS, SHA3, HMAC, AES, KBKDF, CKG	drbgState, objAuth	CO	W	Approved
			objSeed		R, E	
			objSymKey, objHmacKey		E	
			objSens		R	
TPM2_CreateLoaded (I/E/D)	Creates an object and loads it in the TPM	SHS, SHA3, HMAC, AES, DRBG, KBKDF, CKG, RSA (signature generation, verification, key generation), ECC (signature generation, verification, key generation)	objPub	CO	R, E	Approved
			nullSeed, ppSeed, epSeed, spSeed, nullProof, phProof, ehProof, shProof, ekRsa, ekEcc, shProofForReseed		E	
			objSeed, objSymKey, objHmacKey, tdrbgState		G, E	

			objSens		G, R, E	
			drbgState		W, E	
TPM2_Duplicate (I/E/D)	Duplicates a loaded object so that it may be used in a different hierarchy	SHS, SHA3, HMAC, AES, DRBG, KBKDF, KTS-RSA, KAS, CKG	dupSeed, dupInSymKey, dupOutSymKey, dupOutHmacKey	CO	G, E, Z	Approved
			objSens, objAuth		R	
			drbgState		W, E	
			objPub		E	
TPM2_Rewrap (I/E/D)	Rewraps a duplicated object with a new parent key	SHS, SHA3, HMAC, AES, KBKDF, KTS-RSA, KAS, CKG	objSens	CO	W, E	Approved
			dupOutSymKey, dupOutHmacKey		G, E, Z	
			dupInpSymKey		W, Z	
			drbgState, objPub		E	
			dupSeed		W, E, Z	
TPM2_Import (I/E/D)	Allows an object to be encrypted using the symmetric encryption values of a Storage Key	SHS, SHA3, HMAC, AES, KBKDF, KTS-RSA, KAS, CKG	drbgState	CO	E	Approved
			objSens, objPub		W, E	
			objAuth		W	
			dupSeed, dupInSymKey		E, Z	
			dupOutSymKey, dupOutHmacKey		W, E, Z	
TPM2_RSA_Encrypt (I/E/D)	Performs RSA encryption	KTS-RSA	objPub	CO	E	Approved
TPM2_RSA_Decrypt (I/E/D)	Performs RSA decryption	KTS-RSA	objSens	CO	E	Approved
TPM2_ECDH_KeyGen (I/E/D)	Shared secret value computation using KAS	KAS	drbgState	CO	W, E	Approved
			ephSensEccKey		G, E, Z	
			ephPubEccKey		G, R, Z	
			objPub		E	
TPM2_ECDH_ZGen (I/E/D)	Shared secret value recovery using KAS	KAS	objSens	CO	E	Approved
			ephPubEccKey		W, E, Z	

TPM2_ECC_Parameters (I)	Returns the parameters of an ECC curve identified by its TCG-assigned curveID	None	None	CO	N/A	Non-security relevant
TPM2_EncryptDecrypt (I/E)	Symmetric encryption or decryption	AES	objSens	CO	E	Approved
TPM2_EncryptDecrypt2 (I/E/D)	Symmetric encryption or decryption	AES	objSens	CO	E	Approved
TPM2_Hash (I/E/D)	Performs a hash operation on data	SHS, SHA3	nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_HMAC (I/E/D)	Performs a HMAC operation on data	HMAC	objSens	CO	E	Approved
TPM2_GetRandom (I/E)	Outputs random bytes from a DRBG	DRBG	drbgState	CO	W, E	Approved
TPM2_StirRandom (I/D)	Reseed the state of a DRBG	ENT(P), DRBG	drbgSeed	CO	W, E, Z	Approved
			drbgState		W, E	
TPM2_HMAC_Start (I/D)	Starts an HMAC sequence	HMAC	seqAuth	CO	W	Approved
			objSens		E	
TPM2_HashSequenceStart (I/D)	Starts a hash or an event sequence	SHS, SHA3	seqAuth	CO	W	Approved
TPM2_SequenceUpdate (I/D)	Adds data to a hash or HMAC sequence	SHS, SHA3, HMAC	objSens	CO	E	Approved
TPM2_SequenceComplete (I/E/D)	Adds last part of data to a hash or HMAC sequence and returns the result	SHS, SHA3, HMAC	nullProof, phProof, ehProof, shProof, objSens	CO	E	Approved
			seqAuth		Z	
TPM2_EventSequenceComplete (I/D)	Adds last part of data to a hash or HMAC sequence and returns the result in a digest list	SHS, SHA3, HMAC	objSens	CO	E	Approved
			seqAuth		Z	
TPM2_Certify (I/E/D)	Proves that an object with a specific Name is loaded in the TPM	SHS, SHA3, HMAC, DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	drbgState	CO	W, E	Approved
			objSens, shProof		E	
TPM2_CertifyCreation (I/E/D)	Proves the association between an object and its creation data	SHS, SHA3, HMAC, DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	drbgState	CO	W, E	Approved
			objSens, nullProof, phProof, ehProof, shProof		E	
TPM2_Quote (I/E/D)	Quotes PCR values	SHS, SHA3, HMAC,	drbgState	CO	W, E	Approved

		DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	objSens, shProof		E	
TPM2_GetSessionAuditDigest (I/E/D)	Returns a digital signature of the audit session digest	SHS, SHA3, HMAC, DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	drbgState	CO	W, E	Approved
			objSens, shProof		E	
TPM2_GetCommandAuditDigest (I/E/D)	Returns the current value of the command audit digest, a digest of the commands being audited, and the audit hash algorithm	SHS, SHA3, HMAC, DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	drbgState	CO	W, E	Approved
			objSens, shProof		E	
TPM2_GetTime (I/E/D)	Returns the current values of Time and Clock	SHS, SHA3, HMAC, DRBG, KBKDF, CKG, RSA (signature generation), ECC (signature generation)	drbgState	CO	W, E	Approved
			objSens, shProof		E	
TPM2_CertifyX509 (I/E/D)	X.509 certificate generation	SHS, SHA3, RSA (signature generation), ECC (signature generation	drbgState	CO	W, E	Approved
			objSens		E	
TPM2_VerifySignature (I/D)	Validates a signature on a message with the message digest passed to the TPM	HMAC, RSA (signature generation), ECC (signature generation)	objPub, nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_Sign (I/D)	Signs an externally provided hash with the specified symmetric or asymmetric signing key	SHS, SHA3, HMAC, DRBG, RSA (signature generation), ECC (signature generation)	objSens, nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_SetCommandCodeAuditStatus (I)	Changes the audit status of a command or to set the hash algorithm used for the audit digest	None	None	CO	N/A	Non-security relevant
TPM2_PCR_Extend (I)	Updates the indicated PCR	SHS, SHA3	None	CO	N/A	Approved
TPM2_PCR_Event (I/D)	Updates the indicated PCR and reports list of digests	SHS, SHA3	None	CO	N/A	Approved
TPM2_PCR_Read (I)	Returns the values of all PCR specified in pcrSelectionIn	None	None	CO	N/A	Non-security relevant

TPM2_PCR_Allocate (I)	Sets the desired PCR allocation of PCR and algorithms	None	None	CO	N/A	Non-security relevant
TPM2_PCR_Reset (I)	Sets the PCR in all banks to zero	None	None	CO	N/A	Non-security relevant
_TPM_Hash_Start	Indicates to the TPM interface the start of an H-CRTM measurement sequence	SHS, SHA3	None	CO	N/A	Approved
_TPM_Hash_Data	Indicates to the TPM interface data to be included in the H-CRTM measurement sequence	SHS, SHA3	None	CO	N/A	Approved
_TPM_Hash_End	Indicates to the TPM interface the end of the H-CRTM measurement sequence	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicySigned (I/E/D)	Includes a signed authorization in a policy	SHS, SHA3, HMAC, RSA (signature verification), ECC (signature verification)	objPub, nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_PolicySecret (I/E/D)	Includes a secret-based authorization to a policy	SHS, SHA3, HMAC	nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_PolicyTicket (I/D)	Includes a ticket in a policy	SHS, SHA3, HMAC	nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_PolicyOR (I)	Allows options in authorizations without requiring that the TPM evaluate all the options	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyPCR (I/D)	Causes conditional gating of a policy based on PCR	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyLocality (I)	Indicates that the policy will be limited to a specific locality	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyNV (I/D)	Causes conditional gating of a policy based on the contents of an NV Index	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyCounterTimer (I/D)	Causes conditional gating of a policy based on the contents of the TPMS_TIME_INFO structure	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyCommandCode (I)	Limits policy to a specific command code	SHS, SHA3	None	CO	N/A	Approved

TPM2_PolicyPhysicalPresence (I)	Physical presence will need to be asserted at the time the authorization is performed	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyCpHash (I/D)	Allows a policy to be bound to a specific command and command parameters	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyNameHash (I/D)	Allows a policy to be bound to a specific set of TPM entities without being bound to the parameters of the command	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyDuplicationSelect (I/D)	Allows qualification of duplication to allow duplication to a selected new parent	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyAuthorize (I/D)	Let a policy authority sign a new policy so that it may be used in an existing policy	SHS, SHA3, HMAC	nullProof, phProof, ehProof, shProof	CO	E	Approved
TPM2_PolicyAuthValue (I)	Allows a policy to be bound to the authorization value of the authorized entity	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyPassword (I)	Allows a policy to be bound to the authorization value of the authorized object	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyGetDigest (I/E)	Returns the current policyDigest of a policy session	None	None	CO	N/A	Non-security relevant
TPM2_PolicyNvWritten (I)	Allows a policy to be bound to the TPMA_NV_WRITTEN attributes	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyTemplate (I/D)	Allows a policy to be bound to a specific creation template	SHS, SHA3	None	CO	N/A	Approved
TPM2_PolicyAuthorizeNV (I)	Provides a capability that is the equivalent of a revocable policy	SHS, SHA3	None	CO	N/A	Approved
TPM2_CreatePrimary (I/E/D)	Creates a Primary Object under one of the Primary Seeds or a Temporary Object under TPM_RH_NULL	SHS, SHA3, HMAC, AES, DRBG, KBKDF, CKG, RSA (signature generation, verification, key generation), ECC (signature generation, verification, key generation)	objPub	CO	R, E	Approved
			nullSeed, ppSeed, epSeed, spSeed, nullProof, phProof, ehProof, shProof, ekRsa, ekEcc, shProofForReseed		E	
			objSeed, objSymKey, objHmacKey, tdrbgState		G, E	

			objSens		G, R, E	
			drbgState		W, E	
TPM2_HierarchyControl (I)	Enables and disables use of a hierarchy and its associated NV storage	None	None	CO	N/A	Non-security relevant
TPM2_SetPrimaryPolicy (I/D)	Sets the authorization policy for a hierarchy	None	None	CO	N/A	Non-security relevant
TPM2_ChangePPS (I)	Replaces the current platform primary seed (PPS) with a value from the DRBG and sets platformPolicy to the default initialization value	None	drbgState	CO	W, E	Approved
			ppSeed, phProof, objSeed, objSens, objPub		Z	
TPM2_ChangeEPS (I)	Replaces the current endorsement primary seed (EPS) with a value from the DRBG and sets endorsementPolicy to the default initialization value	None	drbgState	CO	W, E	Approved
			epSeed, ehProof, objSeed, objSens, objPub, ekRsa, ekEcc		Z	
TPM2_Clear (I)	Removes all TPM context associated with a specific Owner	None	drbgState	CO	W, E	Approved
			spSeed, ehProof, shProof, shProofForReseed, objSeed, objSens, objPub, objAuth		Z	
TPM2_ClearControl (I)	Disables and enables the execution of TPM2_Clear()	None	None	CO	N/A	Non-security relevant
TPM2_HierarchyChangeAuth (I/D)	Changes the authValue of hierarchies	None	None	CO	N/A	Non-security relevant
TPM2_DictionaryAttackLockReset (I)	Cancels the effect of a TPM lockout due to several successive authorization failures	None	None	CO	N/A	Non-security relevant
TPM2_DictionaryAttackParameters (I)	Changes the lockout parameters	None	None	CO	N/A	Non-security relevant
TPM2_VendorCmdFieldUpgradeStart (I)	Initiates a field upgrade session	SHS, SHA3, KBKDF, CKG, ECC (signature verification)	fuSigKey	CO	E	Approved
TPM2_VendorCmdFieldUpgradeData (I)	Conveys firmware in a field upgrade session	SHS	None	CO	N/A	Approved
TPM2_ContextSave		KBKDF, HMAC, AES, CKG	contextEncKey	CO	G, E, Z	Approved

	Saves a session context, object context, or sequence object context outside the TPM		objSeed, objSens, objPub, objAuth		R	
			nullProof, phProof, ehProof, shProof, contextEncKey, contextKey		E	
TPM2_ContextLoad	Reloads a context that has been saved by TPM2_ContextSave()	KBKDF, HMAC, AES, CKG	contextEncKey	CO	G, E, Z	Approved
			objSeed, objSens, objPub, objAuth		R	
			nullProof, phProof, ehProof, shProof, contextEncKey, contextKey		E	
TPM2_FlushContext	Causes all context associated with a loaded object, sequence object, or session to be removed from TPM memory	None	objSeed, objSens, objPub, sesHmacKey, sesSymKey	CO	Z	Approved
TPM2_EvictControl (I)	Allows certain Transient Objects to be made persistent or a persistent object to be evicted	None	objSeed, objSens, objPub, objAuth	CO	R, W, Z	Approved
			sesHmacKey, sesSymKey		R, W	
TPM2_ReadClock (I)	Reads the current TPMS_TIME_INFO structure	None	None	CO	N/A	Non-security relevant
TPM2_ClockSet (I)	Advances the value of the TPM's clock	None	None	CO	N/A	Non-security relevant
TPM2_ClockRateAdjust (I)	Adjusts the rate of advance of *Clock* and *Time*	None	None	CO	N/A	Non-security relevant
TPM2_GetCapability (I)	Returns various information regarding the TPM and its current state	None	None	CO	N/A	Non-security relevant
TPM2_TestParms (I)	Checks if specific combinations of algorithm parameters are supported	None	None	CO	N/A	Non-security relevant
TPM2_NV_DefineSpace (I/D)	Defines the attributes of an NV Index and causes the TPM to reserve space to hold the data associated with the NV Index	None	nvAuth	CO	W	Approved
TPM2_NV_UndefineSpace (I)	Removes an Index from the TPM	None	nvAuth	CO	Z	Approved

TPM2_NV_UndefineSpaceSpecial (I)	Removal of a platform-created NV Index that has TPMA_NV_POLICY_DELETE SET	None	nvAuth	CO	Z	Approved
TPM2_NV_ReadPublic (I/E)	Reads the public area and Name of an NV Index	SHS, SHA3	None	CO	N/A	Approved
TPM2_NV_Write (I/D)	Writes a value to an area in NV memory that was previously defined by TPM2_NV_DefineSpace()	None	None	CO	N/A	Non-security relevant
TPM2_NV_Increment (I)	Increments the value in an NV Index that has the TPM_NT_COUNTER attribute	None	None	CO	N/A	Non-security relevant
TPM2_NV_Extend (I/D)	Extends a value to an area in NV memory that was previously defined by TPM2_NV_DefineSpace()	SHS, SHA3	None	CO	N/A	Approved
TPM2_NV_SetBits (I)	Sets bits in an NV Index that was created as a bit field	None	None	CO	N/A	Non-security relevant
TPM2_NV_WriteLock (I)	Inhibits further writes of the NV Index if the TPMA_NV_WRITEDEFINE or TPMA_NV_WRITE_STCLEAR attributes of an NV location are SET	None	None	CO	N/A	Non-security relevant
TPM2_NV_GlobalWriteLock (I)	Sets TPMA_NV_WRITELOCKED for all indexes that have their TPMA_NV_GLOBALLOCK attribute SET	None	None	CO	N/A	Non-security relevant
TPM2_NV_Read (I/E)	Reads a value from an area in NV memory previously defined by TPM2_NV_DefineSpace()	None	None	CO	N/A	Non-security relevant
TPM2_NV_ReadLock (I)	Prevents further reads of the NV Index until the next TPM2_Startup (TPM_SU_CLEAR) if TPMA_NV_READ_STCLEAR is SET	None	None	CO	N/A	Non-security relevant
TPM2_NV_ChangeAuth (I/D)	Allows the authValue of an NV Index to be changed	None	nvAuth	CO	W	Approved
TPM2_NV_Certify (I/E/D)	Certifies the contents of an NV Index or portion of an NV Index	SHS, SHA3, HMAC, ECC (signature generation), RSA (signature generation)	objSens	CO	E	Approved
TPM2_VendorCmdSetMode (I)	Sets the low power mode	None	None	CO	N/A	Non-security relevant

TPM2_VendorCmdSetCommandSet (I)	Activates and locks commands	None	None	CO	N/A	Non-security relevant
TPM2_VendorCmdSetCommandSetLock (I)	Prevents locking commands	None	None	CO	N/A	Non-security relevant
TPM2_VendorCmdGetRandom2 (I/E)	Get random value from DRBG	DRBG	drbgState	CO	W, E	Approved
TPM2_VendorCmdGPIOConfig (I)	Configures GPIO	None	None	CO	N/A	Non-security relevant
TPM2_VendorCmdGetRandom800_90B (I/E)	Get random value from ENT (P)	ENT	None	CO	N/A	Approved
TPM2_VendorCmdChangeObjectDeletionAuth (I)	Modifies deletion authorization for an object	None	None	CO	N/A	Non-security relevant
TPM2_VendorCmdRestoreEK (I)	Restore EK RSA or EK ECC in case of deletion by TPM2_ChangeEPS	None	ekRsa, ekEcc	CO	W	Approved
TPM2_VendorCmdZeroizeEK (I)	Zeroize EK RSA and EK ECC	None	ekRsa, ekEcc	CO	Z	Approved
TPM2_PP_Commands	Determines which commands require assertion of Physical Presence	None	None	CO	N/A	Non-security relevant
Integrity mechanism provided by sessions[1]	This service is not callable from TPM interface but is only used internally by any command and response with an authorization area. It consists in computing the integrity of the received command or transmitted response.	SHS, SHA3, DRBG, KBKDF, HMAC, CKG	sesHmacKey	CO	E, Z	Approved
Encryption mechanism provided by sessions[2]	This service is not callable from TPM interface but is only used internally by any command and response with an encryption or decryption session. It consists in decrypting the first parameter of a received command or encrypting the first parameter of a transmitted response.	SHS, SHA3, DRBG, KBKDF, CKG, AES, XOR	sesSymKey	CO	G, E, Z	Approved

Table 14 - Approved Services

[1] The internal security function is not directly callable from the security module external interfaces. Function is used (or might be used) by the services listed in this table. When a service is usable with a session, (I) is added next to the service name. When a service can additionally use the encryption mechanism of a session, (I/E) is added next to the service name.

Name	Description	Algorithms Accessed	Role	Indicator
TPM2_Create TPM2_CreateLoaded TPM2_Load TPM2_LoadExternal	Creation or loading of an ECC key with a non-approved elliptic curve: • ECC key with curve BN P-256	ECC BN P-256	CO	Not approved
	Creation or loading of an ECC signing key with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	-		
	Creation or loading of an RSA decryption key with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	-		
	Creation or loading of a 1024-bit RSA key	RSA		
TPM2_CreateLoaded	Derivation of an ECC key from a derivation parent key	KBKDF ECC derived keys		
TPM2_Load TPM2_LoadExternal	Loading of an ECC or RSA key (sensitive and public parts) in the NULL hierarchy	-		
TPM2_Duplicate TPM2_Rewrap TPM2_Import	Key transport with a 1024-bit RSA key Key agreement scheme with a non-approved ECC curve: • BN P-256	RSA ECC BN P-256	CO	Not approved
TPM2_RSA_Encrypt TPM2_RSA_Decrypt	Key transport with a non-approved scheme: • RSAES-PKCS1-v1_5 • RSA with no padding mode (null scheme) Key transport with an RSA decryption key: • Generated with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL) • Loaded in the NULL hierarchy	RSAES-PKCS1-v1_5 RSA with no padding scheme KTS-RSA	CO	Not approved
TPM2_ECDH_KeyGen	Use of a non-approved elliptic curve: • ECC key with curve BN P-256	ECC BN P-256		
TPM2_ECDH_ZGen	Use of an ECC key: • Generated on curve BN P-256 • Derived from a derivation parent key • Loaded in the NULL hierarchy	ECC BN P-256 KBKDF		
TPM2_ZGen_2Phase	This command is only usable jointly with TPM2_EC_Ephemeral service that is non approved as using key derivation to generate ECC keys	-		
TPM2_HMAC	HMAC generation with a key length < 112 bits	HMAC	CO	Not approved

TPM2_HMAC_Start TPM2_SequenceUpdate TPM2_SequenceComplete	HMAC generation with a key length < 112 bits	HMAC	CO	Not approved
TPM2_Certify TPM2_CertifyCreation TPM2_Quote TPM2_GetSessionAuditDigest TPM2_GetCommandAuditDigest TPM2_GetTime TPM2_CertifyX509	Digital signature with a non-approved signature scheme: • ECC signature with ECDAA signature scheme • ECC signature with ECSchnorr signature scheme • RSA signature with key length of 1024 bits • ECC or RSA signature key using SHA-1 as digest method • ECC signature with curve BN P-256	ECDAA, ECSchnorr, RSA, SHA-1, ECC BN P-256	CO	Not approved
	Digital signature with an ECC signing key generated with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	ECDSA		
	Digital signature with an ECC signing derived from a derivation parent key	ECDSA		
	Digital signature with an ECC or RSA key loaded in the NULL hierarchy	RSA, ECDSA		
TPM2_Commit	Generation of an ECC key through key derivation method	KBKDF	CO	Not approved
TPM2_EC_Ephemeral	Generation of an ECC key through key derivation method	KBKDF		
TPM2_VerifySignature	Digital signature verification with a non-approved signature scheme or a non-approved curve: • ECDAA signature scheme • ECSchnorr signature scheme • ECC signature with curve BN P-256	ECDAA, ECSchnorr, ECC BN P-256	CO	Not approved
TPM2_Sign	Digital signature generation with a non-approved signature scheme: • ECC signature with ECDAA signature scheme • ECC signature with ECSchnorr signature scheme • RSA signature with key length of 1024 bits • ECC or RSA signature key using SHA-1 as digest method • ECC signature with curve BN P-256	ECDAA, ECSchnorr, RSA, SHA-1, ECC BN P-256		
	Digital signature with an ECC signing key generated with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	ECDSA		
	Digital signature with an ECC signing derived from a derivation parent key	ECDSA		

	Digital signature with an ECC or RSA key loaded in the NULL hierarchy	RSA, ECDSA		
TPM2_PolicySigned	Digital signature verification with a non-approved signature scheme or a non-approved curve: • ECDAA signature scheme • ECSchnorr signature scheme • ECC signature with curve BN P-256	ECDAA, ECSchnorr, ECC BN P-256	CO	Not approved
TPM2_CreatePrimary	Creation and loading of an ECC key with a non-approved elliptic curve: • ECC key with curve BN P-256	ECC BN P-256	CO	Not approved
	Creation and loading of an ECC signing key with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	-		
	Creation and loading of an RSA decryption key with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	-		
TPM2_NV_Certify	Digital signature with a non-approved signature scheme: • ECC signature with ECDAA signature scheme • ECC signature with ECSchnorr signature scheme • RSA signature with key length of 1024 bits • ECC or RSA signature key using SHA-1 as digest method • ECC signature with curve BN P-256	ECDAA, ECSchnorr, ECC BN P-256 RSA, SHA-1	CO	Not approved
	Digital signature with an ECC signing key generated with an undetermined scheme (field inPublic.buffer.parameters.scheme.scheme = TPM_ALG_NULL)	ECDSA		
	Digital signature with an ECC signing derived from a derivation parent key	ECDSA		
	Digital signature with an ECC or RSA key loaded in the NULL hierarchy	RSA, ECDSA		

Table 15 - Non-Approved Services

5 SOFTWARE/FIRMWARE SECURITY

A block diagram of the FW is provided in Figure 4.

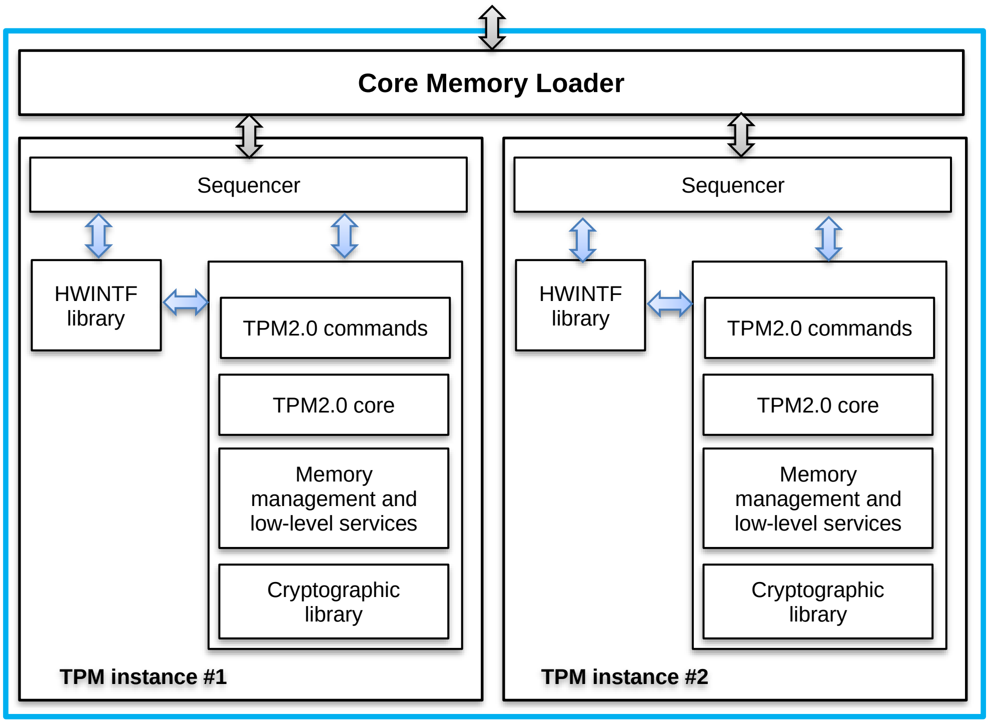

Figure 4 - FW block diagram

FW integrity is verified by computing an EDC (CRC-16 ISO 13239) over the active FW and comparing it to a reference value. FW integrity is verified during boot sequence before execution of one of the code blocks (CML and TPM) and can be triggered on demand by the operator with the execution of the service TPM2_SelfTest (full parameter must be set to YES) or TPM2_IncrementalSelfTest. If failure is detected during boot sequence, TPM enters an infinite reset loop that can be exit only by a power-off/power-on sequence. If failure is detected during self-tests, the security module enters failure mode.

6 OPERATIONAL ENVIRONMENT

Module operational environment is “limited” because it allows loading authenticated firmware that meets all applicable requirements of **[FIPS 140-3]** standard.

Loading of FW on the security module can be achieved by using two services:

- TPM2_VendorCmdFieldUpgradeStart that performs the software/firmware load test detailed in the self-test section of this document to determine if the authorizations to start a loading session are granted
- TPM2_VendorCmdFieldUpgradeData that transports the protected (confidentiality and integrity) parts of the FW

Data outputs are inhibited until the loading session has completed successfully. Execution of the successfully loaded FW is only effective after the next reset of the security module.

New firmware versions must be validated through the FIPS 140-3 evaluation process. Any other firmware loaded into this module is out of the scope of this validation and require a separate FIPS 140-3 validation.

The core memory loader (CML) represented in Figure 4 is non-modifiable, only the TPM instances are modifiable by using an authenticated firmware upgrade mechanism.

The security module contains two instances of the FW but only one FW instance is executed after a boot sequence.

7 PHYSICAL SECURITY

The security module meets the Physical Security protection requirements for single-chip module at FIPS 140-3 Level 1. The module is production grade.

7.1 Zeroization

Zeroization, performed for physical security purposes by some services (refer to detailed services in Table 15), occurs in a sufficiently small time-period to prevent the recovery of the sensitive data between the time of detection and the actual zeroization.

8 NON-INVASIVE SECURITY

The security module does not claim support of non-invasive security attack mitigation techniques referenced in **[NIST SP800-140F]**.

9 SENSITIVE SECURITY PARAMETERS MANAGEMENT

9.1 Storage Areas

Next table lists the SSP storage methods.

Name	Description	Persistence Type
Dynamic RAM	Volatile memory used to store SSPs between two consecutive resets or power-on/power-off sequence of the security module. SSPs doesn't persist after command execution.	Dynamic
Static RAM	Volatile memory used to store SSPs between two consecutive resets or power-on/power-off sequence of the security module. SSPs persist after command execution.	Static
NVRAM	Non-volatile memory (flash-based) used to store SSPs and make them persistent to a reset or a power-off/power-on sequence of the security module	Static

Table 16 - Storage Areas

9.2 SSP Input-Output Methods

Next table lists the SSP input and output methods.

Name	From	To	Format type	Distribution type	Entry type	SFI or Algorithm [O]
Input plaintext to NVRAM	Outside of cryptographic boundary	NVRAM	Plaintext	Manual or Automated	Electronic	None
Input protected to NVRAM	Outside of cryptographic boundary	NVRAM	Encrypted	Manual or Automated	Electronic	KTS (AES cert + HMAC cert) **(A2553 + A2551)**
Input plaintext to RAM	Outside of cryptographic boundary	Static RAM	Plaintext	Manual or Automated	Electronic	None
Input protected to RAM	Outside of cryptographic boundary	Static RAM	Encrypted	Manual or Automated	Electronic	KTS (AES cert + HMAC cert) **(A2553 + A2551)**
Output plaintext from NVRAM	NVRAM	Outside of cryptographic boundary	Plaintext	Manual or Automated	Electronic	None
Output protected from NVRAM	NVRAM	Outside of cryptographic boundary	Encrypted	Manual or Automated	Electronic	KTS (AES cert + HMAC cert) **(A2553 + A2551)**
Output plaintext from RAM	Static RAM	Outside of cryptographic boundary	Plaintext	Manual or Automated	Electronic	None
Output protected from RAM	Static RAM	Outside of cryptographic boundary	Encrypted	Manual or Automated	Electronic	KTS (AES cert + HMAC cert) **(A2553 + A2551)**
Input asym. encrypted to RAM	Outside of cryptographic boundary	Static RAM	Encrypted	Manual or Automated	Electronic	KTS-RSA **(A2554)** KAS **(A2555)**
Output asym. encrypted to RAM	Static RAM	Outside of cryptographic boundary	Encrypted	Manual or Automated	Electronic	KTS-RSA **(A2554)** KAS **(A2555)**
Input during manufacturing	Outside of cryptographic boundary	NVRAM	Obfuscated	Automated	Electronic	None

Table 17 - SSP Input-Output Methods

9.3 SSP Zeroization Methods

Next table lists the SSP zeroization methods.

Method	Description	Rationale	Operator Initiation Capability
Reset	Zeroization of all volatile SSPs	-	Activation of reset signal
TPM2_Clear	Zeroization of all contexts associated with an Owner	SSPs linked to an Owner must not persist if the Owner changes	Send TPM2_Clear command
TPM2_Startup	Zeroization of platformAuth	Zeroize platformAuth before its first use after a reset	Send TPM2_Startup command
TPM2_ChangePPS	Zeroize the platform primary seed and flush all transient and persistent objects in the Platform hierarchy	Platform hierarchy renewal	Send TPM2_ChangePPS command

TPM2_ChangeEPS	Zeroize the endorsement primary seed and flush all transient and persistent objects in the Endorsement hierarchy	Endorsement hierarchy renewal	Send TPM2_ChangeEPS command
TPM2_EvictControl	Zeroize an object from NVRAM	Method required to zeroize a dedicated object in NVRAM	Send TPM2_EvictControl command
TPM2_FlushContext	Zeroize an object from RAM	Method required to zeroize a dedicated object in RAM	Send TPM2_FlushContext command
Automatic	Zeroize SSPs at the end of a command processing	Method for limited life-cycle SSPs	No, zeroization is automatic.
TPM2_NV_UndefineSpace TPM2_NV_UndefineSpaceSpecial	Zeroize a NV index	Method required to flush NV indices from NVRAM	Send TPM2_NV_UndefineSpace command. Send TPM2_NV_UndefineSpaceSpecial command
TPM2_VendorCmdZeroizeEK	Zeroize the endorsement key provisioned	Mandatory zeroization method for EK SSPs	Send TPM2_ZeroizeEK command
TPM2_SequenceComplete TPM2_EventSequenceComplete	Zeroize a hash or HMAC sequence	Method required to flush sequences from RAM	Send TPM2_SequenceComplete command. Send TPM2_EventSequenceComplete command

Table 18 - SSP Zeroization Methods

9.4 SSPs

Next tables list all the SSPs in the security module.

Name[1]	Description	Size (bits)	Strength	Type	Generated by[2]	Established by	Inputs / Outputs	Storage	Zeroization	Used by[3]	Category	Related SSPs
nullProof	Proof (secret value) of the null hierarchy	512	256	Symmetric key	DRBG	Internal	-	Obfuscated in Static RAM	Reset	• KBKDF CTR to generate context encryption key and IV (cf. [TPM2.0 Part1] §30.3.1) • HMAC SHA2-384 to compute context blob integrity (cf. [TPM2.0 Part1] §30.3.2) • HMAC SHA2-384 to compute/verify tickets	CSP	contextEncKey is derived from nullProof / phProof / ehProof nullProof / phProof / ehProof are derived from drbgState
phProof	Proof (secret value) of the platform hierarchy	512	256	Symmetric key	DRBG	Internal	-	Obfuscated in NVRAM	TPM2_ChangePPS		CSP	
ehProof	Proof (secret value) of the endorsement hierarchy	512	256	Symmetric key	DRBG	Internal	-	Obfuscated in NVRAM	TPM2_ChangeEPS		CSP	
shProof	Proof (secret value) of the storage hierarchy	512	256	Symmetric key	DRBG	Internal	-	Obfuscated in NVRAM	TPM2_Clear	• KBKDF CTR to generate context encryption key and IV (cf. [TPM2.0 Part1] §30.3.1) • HMAC SHA2-384 to compute context blob integrity (cf. [TPM2.0 Part1] §30.3.2) • HMAC SHA2-384 to compute/verify tickets • KBKDF CTR to generate obfuscation value used in attestation commands (cf. [TPM2.0 Part1] §36.7)	CSP	contextEncKey is derived from shProof shProof is derived from drbgState
shProofForReseed	Random value	512	256	Entropy source	ENT (P)	Internal	-	Obfuscated in NVRAM	TPM2_Clear	DRBG for reseed before generating *objSeed* PSP in the endorsement hierarchy (cf. [TPM2.0 Part1])	CSP	drbgState is reseeded with shProofForReseed
platformAuth	Authentication value for the platform hierarchy	512	128 to 256 (depending on the underlying hash algorithm used)	Authentication value / Symmetric key	Set to 0 by default at each reset / -	Internal / External	Input protected to RAM or Input plaintext to RAM (as parameter of TPM2_HierarchyChangeAuth)	Obfuscated in Static RAM	TPM2_Startup	• HMAC SHS/SHA3 authorization in case of unsalted and unbound session • KBKDF CTR to generate session key used in HMAC authorization in case of bound session	CSP	sesHmacKey can be derived from platformAuth / endorsementAuth / ownerAuth / lockoutAuth

[1] Temporary storage duration column was removed for readability purpose because when temporary storage is indicated, duration corresponds to the duration of a command execution.

[2] The algorithms indicated in this column correspond to the certified algorithms listed in Table 5.

[3] The algorithms indicated in this column correspond to the certified algorithms listed in Table 5.

endorsementAuth	Authentication value for the endorsement hierarchy	512		Authentication value / Symmetric key	Set to 0 by default / -	Internal / External		Obfuscated in NVRAM	TPM2_Clear TPM2_ChangeEPS	- HMAC SHA-2/SHA3 authorization in case of salted or bound session (key is concatenation of sessionKey and authValue) - KBKDF CTR to generate session key used in HMAC authorization in case of salted and bound session (key is concatenation of authValue and salt)	CSP	New input platformAuth / endorsementAuth / ownerAuth / lockoutAuth values can be wrapped by sesSymKey and integrity protected by sesHmacKey
ownerAuth	Authentication value for the storage hierarchy	512		Authentication value / Symmetric key	Set to 0 by default / -	Internal / External		Obfuscated in NVRAM	TPM2_Clear		CSP	
lockoutAuth	Authentication value for the lockout hierarchy	512		Authentication value / Symmetric key	Set to 0 by default / -	Internal / External		Obfuscated in NVRAM	TPM2_Clear		CSP	
objSeed	Seed value for object generation	384	128 to 256	Data, Symmetric key	DRBG or KBKDF	Internal	-	Obfuscated in Static RAM or NVRAM	TPM2_Clear TPM2_ChangePPS TPM2_ChangeEPS	- Data in SHS/SHA3 (all modes) computation to generate object's unique value (HMAC and symmetric key creation) - Key in KBKDF CTR to generate a symmetric encryption key used in TPM2B_PRIVATE structure encryption/decryption. - Key in KBKDF CTR to generate HMAC key used in TPM2B_PRIVATE integrity protection generation or verification	CSP	objSymKey and objHmacKey are derived from objSeed objSeed can be derived from tdrbgState for primary objects, from drbgState for ordinary objects, from parents seed for derived objects
objAuth	Object's authorization value	1 to 384	1 to 256	Authentication value / Symmetric key	User	External	Input protected to RAM or Input plaintext to RAM on keys creation commands. Changed with command TPM2_ObjectChangeAuth.	Obfuscated in Static RAM or NVRAM	TPM2_Clear TPM2_ChangePPS TPM2_ChangeEPS	HMAC SHS/SHA3 and/or KBKDF CTR keys or part of keys in session based on HMAC or password (usage is the same than for endorsementAuth, ownerAuth, platformAuth and lockoutAuth)	CSP	sesHmacKey and sesSymKey can be derived from objAuth objAuth can be protected by sesHmacKey and sesSymKey
objSymKey	Encryption key of object private part	256	256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM or NVRAM	Automatic	Symmetric encryption / decryption key with AES CFB128 of TPM2B_PRIVATE structure	CSP	objSens is wrapped by objSymKey objSymKey can wrap platformAuth / endorsementAuth / ownerAuth / lockoutAuth / objAuth
objHmacKey	Integrity key of object private part	160, 256, 384	128 to 256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM or NVRAM	Automatic	Integrity protection generation or verification with HMAC SHS/SHA3 of TPM2B_PRIVATE structure	CSP	objSens is integrity protected by objHmacKey objHmacKey can protect platformAuth / endorsementAuth / ownerAuth / lockoutAuth / objAuth
objSens	Object private part	2048, 3072, 4096 (RSA) 128, 192, 256 (AES) 256, 384 (ECC) 1 to 1024 (HMAC)	1 to 256	Symmetric or asymmetric private key	DRBG or KBKDF / -	Internal / External	Output protected from RAM Input protected to RAM Input plaintext to RAM	Obfuscated in Static RAM or NVRAM	TPM2_Clear TPM2_ChangePPS TPM2_ChangeEPS	Depending on object's type, sensitive is used as private key for: - Symmetric encryption/decryption (AES all modes) - Obfuscation/De-obfuscation (XOR) - Asymmetric encryption/decryption (RSA all modes) - Signature generation (RSA, ECDSA, HMAC all modes) - Secret value exchange (KAS all modes) - Key for derivation of derived objects (KBKDF CTR) Key type and length are selected by user thanks to the keys creation commands.	CSP	objSymKey wraps objSens objHmacKey can integrity protect objSens objSens can be generated from tdrbgState for primary objects, from drbgState for ordinary objects and derived from parents seed for derived objects
objPub	Object public part	2048, 3072, 4096 (RSA) 512,768 (ECC)	112 to 192	Asymmetric public key	ECDSA key generation, RSA key generation / -	Internal / External	Output plaintext from RAM Input plaintext to RAM	Obfuscated in Static RAM or NVRAM	TPM2_Clear TPM2_ChangePPS TPM2_ChangeEPS	- Encrypt data or verify signature (RSA SHA-1, SHA2-256, SHA2-384, RSASSA-PKCS-v1.5, RSASSA-PSS) - Secret key exchange (KAS ECC One pass DH) or signature verification (ECDSA SHA-1,	PSP	objPub is computed from objSens

										SHA2-256, SHA2-384, SHA3-256, SHA3-384)		
nvAuth	Authorization of NV index	1 to 384	1 to 256	Authentication value / Symmetric key	User	External	Input protected to RAM Input plaintext to RAM Changed with command TPM2_NV_ChangeAuth.	Obfuscated in NVRAM	TPM2_NV_UndefineSpace TPM2_NV_UndefineSpaceSpecial	HMAC SHS/SHA3 and/or KBKDF CTR keys or part of keys in session based on HMAC or password (usage is the same than for endorsementAuth, ownerAuth, platformAuth and lockoutAuth)	CSP	sesHmacKey can be derived from nvAuth New input nvAuth value can be wrapped by sesSymKey and integrity protected by sesHmacKey
sesSalt	Salt for keys diversification	160, 256, 384	128 to 256	Symmetric key	User	External	Input protected to RAM	Obfuscated in Dynamic RAM	Automatic	Part of KBKDF CTR key to generate the sesHmacKey CSP (cf. [TPM2.0 Part1	CSP	sesHmacKey is derived from sesSalt
sesHmacKey	HMAC session key	160, 256, 384	128 to 256	Symmetric key	KBKDF	Internal / External	Input protected to RAM	Obfuscated in Dynamic RAM	Automatic	• HMAC SHS/SHA3 key used to generate and verify command authorization • Part of KBKDF CTR key used to generate encryption key and IV of encryption-based session	CSP	sesHmacKey can protect all inputs CSPs contextKey and contextEncKey keys can wrap sesHmacKey
sesSymKey	Encrypted session key	128, 192, 256	128 to 256	Symmetric key	KBKDF	Internal / External	Input protected to RAM	Obfuscated in Dynamic RAM	Automatic	• Key and IV for symmetric encryption / decryption of first parameter of command / response if parameter structure is of type TPM2B_	CSP	sesSymKey is derived from sesHmacKey and platformAuth / endorsementAuth / ownerAuth / lockoutAuth / objAuth / seqAuth
contextKey	Derivation key for context protection	128	128	Symmetric key	DRBG	Internal	-	Obfuscated in RAM	Reset	First part of key used in KBKDF CTR to generate a symmetric encryption key and IV used in context blob encryption / decryption	CSP	contextKey is generated from drbgState contextEncKey is derived from contextKey
contextEncKey	Wrapping key for context protection	256	256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM	Automatic	AES CFB128 encryption / decryption of context blob	CSP	contextEncKey is derived from contextKey and nullProof / phProof / ehProof / shProof
dupInSymKey	Wrapping key for duplicated object	128, 192, 256	128 to 256	Symmetric key	DRBG	Internal / External	Input plaintext to RAM Input protected to RAM Output plaintext from RAM Output protected from RAM	Obfuscated in Dynamic RAM	Automatic	AES CFB128 symmetric encryption / decryption key to protect TPM2B_PRIVATE output structure	CSP	dupInSymKey can be wrapped by sesSymKey and protected by sesHmacKey
dupSeed	Seed for protection keys derivation	160 to 384	128 to 256	Symmetric key	DRBG, KAS	Internal / External	Input asym. encrypted to RAM Output asym. encrypted from RAM	Obfuscated in Dynamic RAM	Automatic	• KBKDF CTR to generate a symmetric encryption / decryption key for outer protection • KBKDF CTR to generate a HMAC key for outer integrity protection	CSP	dupSeed is encrypted by objPub key (RSA or KAS)
dupOutSymKey	HMAC key for duplicated objects	128, 192, 256	128 to 256	Symmetric key	KBKDF	Internal	-	Obfuscated in RAM	Automatic	AES CFB128 symmetric encryption / decryption key to protect TPM2B_PRIVATE output structure	CSP	dupOutSymKey is derived from dupSeed dupOutSymKey wraps objSens
dupOutHmacKey	Encryption key for duplicated objects	160, 256, 384	128 to 256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM	Automatic	HMAC SHS/SHA3 key for outer protection of TPM2B_PRIVATE output structure	CSP	dupOutHmacKey is derived from dupSeed dupOutHmacKey protects objSens
creSeed	Seed for credential keys derivation	160 to 384	128 to 256	Symmetric key	User	External	Input asym. encrypted to RAM	Obfuscated in Dynamic RAM	Automatic	• KBKDF CTR to generate a symmetric encryption / decryption key for outer protection • KBKDF CTR to generate a HMAC key for outer integrity protection	CSP	
creSymKey	HMAC key for credentials	128, 192, 256	128 to 256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM	Automatic	AES CFB128 symmetric encryption / decryption key for outer protection of credentialBlob	CSP	creSymKey is derived from creSeed
creHmacKey	Encryption key for credentials	160, 256, 384	128 to 256	Symmetric key	KBKDF	Internal	-	Obfuscated in Dynamic RAM	Automatic	HMAC SHS/SHA3 integrity key for outer protection of credentialBlob	CSP	creHmacKey is derived from creSeed
ephSensEccKey	ECC ephemeral private key	256, 384	128 to 192	ECC private key	DRBG	Internal	-	Obfuscated in Dynamic RAM	Automatic	Part of KAS ECC one pass DH service	CSP	ephSensEccKey is derived from drbgState
ephPubEccKey	ECC ephemeral public key	512, 768	128 to 192	ECC public key	ECDSA key generation	Internal	-	Obfuscated in Dynamic RAM	Automatic	Part of KAS ECC one pass DH service	PSP	ephSensEccKey is generated from ephSensEccKey

ekRsa	Provisioned RSA endorsement key	2048	112	RSA private key	RSA key generation	External	Input during manufacturing	Obfuscated in NVRAM	TPM2_ZeroizeEK	KTS-RSA KTS-OAEP basic	CSP	ekRsa is copied in objSens
ekEcc	Provisioned ECC endorsement key	256, 384	128 to 192	ECC private key	ECDSA key generation	External	Input during manufacturing	Obfuscated in NVRAM	TPM2_ZeroizeEK	KAS ECC one pass DH service	CSP	ekEcc is copied in objSens
fuSigKey	Field upgrade signature verification key	384	192	ECC public key	ECDSA key generation	External	Input during manufacturing	Obfuscated in NVRAM	-	ECDSA SHA2-384 signature verification on a FW upgrade start command	PSP	-
seqAuth	Authorization value for hash or HMAC sequence	1 to 384	1 to 256	Authentication value / Symmetric key	User	External	Input plaintext to RAM Input protected to RAM on TPM2_HashSequenceStart or TPM2_HMAC_Start commands	Obfuscated in NVRAM	TPM2_SequenceComplete TPM2_EventSequenceComplete	HMAC SHS/SHA3 and/or KBKDF CTR keys or part of keys in session based on HMAC or password for TPM2_SequenceUpdate, TPM2_SequenceComplete or TPM2_EventSequenceComplete commands authorizations	CSP	sesSymKey and sesHmacKey are derived from seqAuth

Table 19 - SSPs (list of keys)

Name[1]	Description	Size (bits)	Strength	Type	Generated by[2]	Established by	Inputs / Outputs	Storage	Zeroization	Used by[3]	Category	Related SSPs
nullSeed	Seed of the null hierarchy	512	256	Seed	ENT(P)	Internal	-	Obfuscated in Static RAM	Reset	DRBG HASH_based SHA2-256 to generate random used for sensitive part creation of primary keys (prime numbers for RSA and private key for ECC / KEYEDHASH / SYMCIPHER objects) and *objSeed* CSP creation for all types of primary keys.	CSP	tdrbgState is instantiated by nullSeed / phSeed / ehSeed / shSeed
phSeed	Seed of the platform hierarchy	512	256	Seed	ENT(P)	Internal	-	Obfuscated in NVRAM	TPM2_ChangePPS		CSP	
ehSeed	Seed of the endorsement hierarchy	512	256	Seed	ENT(P)	Internal	-	Obfuscated in NVRAM	TPM2_ChangeEPS		CSP	
shSeed	Seed of the storage hierarchy	512	256	Seed	ENT(P)	Internal	-	Obfuscated in NVRAM	TPM2_Clear		CSP	
drbgState	Internal state (V and C secret values) of the DRBG (based on SHA256)	256	256	State	DRBG	Internal	-	Obfuscated in Static RAM	TPM2_Clear	Random numbers and seeds	CSP	drbgState is seeded by drbgSeed
drbgSeed	Seed value for the DRBG	512	256	Seed	ENT(P)	Internal	-	Obfuscated in Dynamic RAM	Automatic	drbgState	CSP	drbgSeed seeds drbgState
tdrbgState	Internal state (V and C secret values) of the transient DRBG (based on SHA256) used to generate prime numbers for primary RSA keys.	256	256	State	DRBG	Internal	-	Obfuscated in Dynamic RAM	Automatic	Prime numbers generation for primary RSA keys	CSP	tdrbgState is instantiated by nullSeed / phSeed / ehSeed / shSeed

Table 20 - SSPs (not used as keys)

Next table gives the security strength of a key depending on the underlying algorithm used and its size.

Algorithm	Underlying algorithm	Key size (bits)	Security strength (bits)
KBKDF	SHA-1	size ≥ 128	128
		size < 128	Key size
	SHA2-256	size ≥ 192	192
		size < 192	Key size
	SHA2-384	size ≥ 256	256
		size < 256	Key size
HMAC	SHA-1	size ≥ 128	128
		size < 128	Key size
	SHA2-256	size ≥ 192	192

[1] Temporary storage duration column was removed for readability purpose because when temporary storage is indicated, duration corresponds to the duration of a command execution.
[2] The algorithms indicated in this column correspond to the certified algorithms listed in Table 5.
[3] The algorithms indicated in this column correspond to the certified algorithms listed in Table 5.

		size < 192	Key size
	SHA2-384	size ≥ 256	256
		size < 256	Key size
DRBG	SHA2-256	-	256
AES	-	128	128
	-	192	192
	-	256	256
RSA	-	2048	112
	-	3072	128
	-	4096	142
ECC	-	256	128
	-	384	192

Table 21 - Security strength of a key depending on the underlying algorithm used and its size

9.5 List of RBGs

The security module implements:

- A Hash-DRBG based on SHA256 and compliant with the **[SP800-90A]** standard (state is indicated as drbgState in Table 20). It is seeded at each module start-up with 512 bits issued from the ENT (P). Hash-DRBG is used for any generation of random values used as SSP in a cryptographic operation. It can be reseeded by using the service TPM2_StirRandom.
- A transient Hash-DRBG based on SHA256 and compliant with the **[SP800-90A]** standard (state is indicated as tdrbgState in Table 20.) involved only in primary keys generation and seeded as defined in **[TPM2.0 Part1]** and **[TPM2.0 Part3].**
- An ENT (P) as detailed in table below.

Entropy Source	Minimum number of bits of entropy	Details
ENT (P)	Min-entropy of 0.814324 per 1-bit sample	This ENT (P) has been evaluated according to the non-IID evaluation path of the **[SP800-90B]** standard. It is used to generate random numbers not dedicated to being used as cryptographic material or to seed or reseed the Hash-DRBG (indicated as drbgSeed in Table 20**.**) listed above with a minimum of 414 bits of entropy.

Table 22 - Non-Deterministic Random Number Generation Specification

10 SELF-TESTS

Self-tests run by the cryptographic module are split into two categories:

- Pre-operational self-tests
- Conditional self-tests

The self-tests do not require operator intervention to run.

10.1 Self-tests error states

In case of self-test failure, the security module outputs the return code TPM_RC_FAILURE as defined in **[TPM2.0 Part2]** via the status interface and the module enters the failure state. In failure state, the module does not perform any cryptographic functions and all data output via the data output interface are inhibited. The only usable services in failure state are TPM2_GetTestResult and TPM2_GetCapability to get a status on the functionality whose self-test failed. Failure can be exit by resetting the security module.

If pre-operational self-tests passed successfully, no success status is indicated but commands that require self-tests to be completed can be successfully executed.

10.2 Pre-operational tests

The module performs the following pre-operational self-tests:

Algorithm	Implementation	Test properties	Test Method	Type	Indicator	Details
Firmware integrity	NA	CRC-16	EDC	Integrity Test	Processing of TPM2_Startup command indicates tests have been run	FW integrity is verified by computing an EDC (CRC-16 ISO 13239) and comparing it to reference values.
HW integrity	NA	HW registers verification		Critical Function		HW integrity is guaranteed via check of HW sensors. If failure is detected during boot sequence, status is set to FAIL, and error is returned.
ENT(P)	NA	RCT and APT	SP 800-90B Health-Tests	Critical Function		TPM performs AIS31 and SP800-90B (RCT and APT) start-up health tests on ENT(P) output sequence. If test fails, test status is set to FAIL, and an error is returned.

Table 23 - Pre-Operational Self-Tests

10.3 Conditional self-tests

The Module performs the following conditional self-tests:

Algorithm	Implementation	Test properties	Test Method	Type	Indicator[1]	Details	Condition
Firmware integrity	NA	CRC-16	EDC	Integrity Test	Bit #1 clear	FW integrity is verified by computing an EDC (CRC-16 ISO 13239) and comparing it to reference values.	TPM2_SelfTest (full = YES)
HW integrity	NA	NA	Flags verification	Critical Function		HW integrity is guaranteed via check of HW sensors. If failure is detected during boot sequence, status is set to FAIL, and error is returned.	

[1] Bit index indicated corresponds to the index in the algo_status field in the TPM2_GetTestResult response

ENT(P)	NA	RCT and APT	SP 800-90B Health-Tests	Critical Function		AIS31 and SP800-90B (RCT and APT) start-up health tests on ENT(P) output sequence. If test fails, test status is set to FAIL, and error is returned.	
Hash-DRBG	NA	Seed (64 bytes)	KAT	CAST		Instantiate then Reseed are seeded with a known seed value. Random is then generated with Generate API to output a 32-bytes value compared to a reference value (single test sequence done in accordance with §11.3 of [SP800-90A]).	
SHA1	Certs #A2548 and #A2549 implementations	Known data (16 bytes)			Bit #1 clear	Hash of known data and comparison of output to an expected digest (20 bytes).	
SHA256					Bit #2 clear	Hash of known data and comparison of output to an expected digest (32 bytes).	
SHA384					Bit #3 clear	Hash of known data and comparison of output to an expected digest (48 bytes).	
SHA3_256	NA				Bit #4 clear	Hash of known data and comparison of output to an expected digest (32 bytes).	
HMAC SHA1	Certs #A2551 and #A2552 implementations	known data (16 bytes) known key (16 bytes			Bit #5 clear	HMAC on known data and known key. Comparison of output to an expected MAC value (20 bytes).	TPM2_SelfTest (full = YES) or TPM2_SelfTest (full = NO) or TPM2_IncrementalSelfTest or Execution of command requiring algorithm or Automatic execution
KDF SP800-108	NA	known data (16 bytes) known label (“TEST”)			Bit #6 clear	KDF on known data and known label. Comparison of output to an expected derivation value (32 bytes).	
AES	NA	known data (32 bytes) known key (16 bytes) known IV (16 bytes).			Bit #7 clear	AES CBC 128 encryption of known data compared to a reference value. AES CBC 128 decryption of encrypted data and comparison to the initial plaintext data.	
KAS	NA	known private key d (32 bytes) known point P (2*32 bytes) NIST P-256 curve			Bit #8 clear	Primitive “Z” Computation and key derivation are implemented: a known private key d is used with a known point P of NIST P-256 curve to compute Q = dP. Key derivation of Q performed with SHA-1 underlying algorithm to output a key of 20 bytes that is compared to a refence value.	
ECDSA	NA	Known key (256 bits) known data (20 bytes) fixed k (20 bytes) NIST P-256 curve			Bit #9 clear	ECDSA signature generation on known data with known key and k. Output of signature is compared to a reference signature. Signature verification performed on the generated signature.	

RSA	NA	Known key (2048 bits) known data (20 bytes) RSASSA-PKCS1-v1_5			Bit #10 clear	RSA signature generation on known data with a known key. Output of signature is compared to a reference signature. Signature verification performed on the generated signature (covers also KTS-RSA functionality).	
FW load	NA	ECDSA NIST P-384) SHA384		Firmware load	Bit #1 clear	Verification of chained digest and signature (ECDSA NIST P-384) to ensure authentication of the FW	
RSA key generation	NA	known data (16 bytes)	PCT	PCT	Key creation failure	Depending on the key purpose (signing or encrypting) indicated in sign attribute of the key, en/decryption or signing/verification is done on known data.	RSA key generation
ECC key generation	NA	fixed k (20 bytes) NIST P-256 or NIST P-384	PCT	PCT	Key creation failure	Depending on the key purpose (signing or key establishment) an ECDSA signature is generated (k fixed and the message varies) and verified with pairwise consistency test as defined by SP800-56Ar3.	ECC key generation

Table 24 - Conditional Self-Tests

10.4 Verification

Successful completion of self-tests can be verified through use of TPM2_GetTestResult command. The first 4 bytes of response indicate self-tests status. If they are equal to 0, self-tests completed successfully. If not, the subsequent 4 bytes indicate the list of algorithms not fully self-tested.

11 LIFE-CYCLE ASSURANCE

11.1 Module installation

During installation of the module:

- Connection of the module with its environment must be done accordingly to the pinout description given at §3.1.

11.2 Module initialization

No initialization procedures are required.

11.3 Module operation

11.3.1 Approved modes of operation

TPM is operated in an approved mode of operation as long as no non-approved service using a non-approved algorithm (listed resp. in Table 15 and Table 14**),** is used. No specific rules of operation are required to operate this module at FIPS 140-3 Level 1.

11.3.2 Normal operation

TPM is in normal operation mode when all pre-operational and conditional self-tests (apart from FW load and PCT tests) are complete. All approved and non-approved services are listed resp. in Table 14 and Table 15 with the corresponding indicator reporting if the service uses an approved cryptographic algorithm or security function.

11.3.3 Error modes

TPM may reach specific states depending on the sequence of operations that occurred.

11.3.3.1 Shutdown mode

The shutdown mode is an infinite HW reset loop that may be exit only by a power-off/power-on sequence. This state is entered when TPM detects a failure of the FW integrity verification during the TPM boot sequence. No output control or data is available in this mode.

11.3.3.2 Failure state

Failure state is a state of the TPM that restricts the executable commands to TPM2_GetCapability and TPM2_GetTestResult (status services). TPM answers to all other commands with the error code TPM_RC_FAILURE (0x101) and doesn't process the requested service. This state is entered when a self-test fails (except FW integrity test during the boot sequence). This state can be exit with a reset of the TPM.

11.3.3.3 Non-approved mode of operation

The module enters a non-approved mode if one of the non-approved services listed in Table 15 is used by the operator. To check if the TPM is in a non-approved mode of operation, TPM2_GetCapability (capability = TPM_CAP_VENDOR_PROPERTIES) with the sub-capability TPM_SUBCAP_VENDOR_TPMA_MODES = 0x7 shall be used. It outputs a 2-bit indicator equals to 0x2 or 0x3 if the module is in a non-approved mode of operation.

11.4 Module termination

End-of-life of the product requires the following zeroization commands to be executed:

- TPM2_Clear
- TPM2_ChangeEPS
- TPM2_ChangePPS

12 MITIGATIONS OF OTHER ATTACKS

The security module does not claim mitigation of other attacks.

13 REFERENCES

Reference	Document
TPM2.0 standard	
[TPM2.0 Part1]	TPM2.0 Main, Part 1, Architecture, rev 1.59, TCG
[TPM2.0 Part2]	TPM2.0 Main, Part 2, Structures, rev 1.59, TCG
[TPM2.0 Part3]	TPM2.0 Main, Part 3, Commands, rev 1.59, TCG
[TPM2.0 Part4]	TPM2.0 Main, Part 4, Supporting routines, rev 1.59, TCG
[TPM2.0 PTP]	TCG PC Client Platform TPM Profile (PTP) Specification, rev. 1.05
[TPM2.0 FIPS 140-3]	TCG FIPS 140-3 Guidance for TPM2.0, v1.0, TCG
FIPS 140-3 standard	
[ISO/IEC 19790]	Information technology — Security techniques — Security requirements for cryptographic modules, ISO/IEC 19790:2012I
[ISO/IEC 24759]	Information technology — Security techniques — Test requirements for cryptographic modules, ISO/IEC 24759:2017I
[FIPS 140-3]	FIPS PUB 140-3, Security Requirements for Cryptographic Modules, National Institute of Standards and Technology (NIST), March 22, 2019
[NIST SP800-140]	NIST Special Publication 800-140, FIPS 140-3 Derived Test Requirements (DTR), CMVP Validation Authority Updates to ISO/IEC 24759, March 2020
[NIST SP800-140A]	NIST Special Publication 800-140A, CMVP Documentation Requirements, CMVP Validation Authority Updates to ISO/IEC 24759, March 2020
[NIST SP800-140B]	NIST Special Publication 800-140B, CMVP Security Policy Requirements, CMVP Validation Authority Updates to ISO/IEC 24759 and ISO/IEC 19790 Annex B, March 2020
[NIST SP800-140C]	NIST Special Publication 800-140Cr1, CMVP Approved Security Functions, CMVP Validation Authority Updates to ISO/IEC 24759, May 2022
[NIST SP800-140D]	NIST Special Publication 800-140Dr1, CMVP Approved Sensitive Security Parameter Generation and Establishment Methods, CMVP Validation Authority Updates to ISO/IEC 24759, May 2022
[NIST SP800-140E]	NIST Special Publication 800-140E, CMVP Approved Authentication Mechanisms, CMVP Validation Authority Requirements for ISO/IEC 19790:2012 Annex E and ISO/IEC 24759 Section 6.17, March 2020
[NIST SP800-140F]	NIST Special Publication 800-140F, CMVP Approved Non-Invasive Attack Mitigation Test Metrics, CMVP Validation Authority Updates to ISO/IEC 24759, March 2020

Reference	Document
[FIPS 140-3 IG]	National Institute of Standards and Technology and Canadian Centre for Cyber Security, Implementation Guidance for FIPS 140-3 and the Cryptographic Module Validation Program
NIST approved security functions	
[SP800-131Ar2]	National Institute of Standards and Technology, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, March 2019.
[FIPS 197]	National Institute of Standards and Technology, *Advanced Encryption Standard (AES),* Federal Information Processing Standards Publication 197, November 2001
[SP800-38A]	National Institute of Standards and Technology, *Recommendation for Block Cipher Modes of Operation: Methods and Techniques,* December 2001.
[SP800-38F]	National Institute of Standards and Technology, *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping,* December 2012.
[FIPS 186-4]	National Institute of Standards and Technology, *Digital Signature Standard (DSS),* Federal Information Processing Standards Publication 186-4, July 2013
[FIPS 180-4]	National Institute of Standards and Technology, *Secure Hash Standard*, Federal Information Processing Standards Publication 180-4, August 2015
[FIPS 202]	National Institute of Standards and Technology, SHA3 Standard: Permutation-Based Hash and Extendable-Output Functions, August 2015
[FIPS 198-1]	National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code,* NIST Computer Security Division Page 3 07/26/2011, *(HMAC)*, Federal Information Processing Standards Publication 198-1, July, 2008
[SP800-135]	National Institute of Standards and Technology, Recommendation for *Existing Application-Specific Key Derivation Functions*, December 2011.
[SP800-108]	National Institute of Standards and Technology, *Recommendation for Key Derivation Using Pseudorandom Functions*, October 2009.
[SP800-90A]	National Institute of Standards and Technology, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, June 2015.
[SP800-56A] Rev 3	National Institute of Standards and Technology, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography,* April 2018.
[SP800-56B] Rev 2	National Institute of Standards and Technology, *Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography,* March 2019
[SP800-56C] Rev 1	National Institute of Standards and Technology, *Recommendation for Key-Derivation Methods in Key-Establishment Schemes,* April 2018
[SP800-133] Rev 2	National Institute of Standards and Technology, *Recommendation for Cryptographic Key Generation*, June 2020

14 ACRONYMS

Term	Definition
AES	Advanced Encryption Standard
CO	Crypto Officer
DES	Data Encryption Standard
DSAP	Delegate Specific Authorization Protocol
EK	Endorsement Key
FIPS	Federal Information Processing Standard
FUM	Field Upgrade Mode
GPIO	General Purpose I/O
HMAC	Keyed-Hashing for Message Authentication
HW	Hardware
KDF	Key derivation function
NIST	National Institute of Standards and Technology
NV	Non-volatile (memory)
OIAP	Object-Independent Authorization Protocol
OSAP	Object Specific Authorization Protocol
PCR	Platform Configuration Register
RSA	Rivest Shamir Adelman
RTM	Root of Trust for Measurement
RTR	Root of Trust for Reporting
SHA	Secure Hash Algorithm
SPI	Serial Peripheral Interface
SRK	Storage Root Key
TCG	Trusted Computed Group
TPM	Trusted Platform Module
TSS	TPM Software Stack

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

This document may be reproduced only in its original entirety without revision.

www.st.com