Nuvoton Cryptographic Library 2.3

Hardware Version 2.3.8

FIPS 140-3 Non-Proprietary Security Policy

Version 1.2

Last update: 2025-01-07

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

1 Table of Contents

1 General

This document is the non-proprietary FIPS 140-3 Security Policy for Hardware version 2.3.8 of the Nuvoton Cryptographic Library 2.3. It has a one-to-one mapping to the [SP 800-140B] starting with section B.2.1 named “General” that maps to section 1 in this document and ending with section B.2.12 named “Mitigation of other attacks” that maps to section 12 in this document. This document also contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for a Security Level 1 module. Table 1 describes the individual security areas of FIPS 140-3, as well as the Security Levels of those individual areas:

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	Not Applicable
6	Operational Environment	1
7	Physical Security	1
8	Non-invasive Security	Not Applicable
9	Sensitive Security Parameter Management	1
10	Self-tests	1
11	Life-cycle Assurance	1
12	Mitigation of Other Attacks	Not Applicable
Overall Level		1

Table 1 - Security Levels

2 Cryptographic Module Specification

The Nuvoton Cryptographic Library 2.3 cryptographic module (hereafter referred to as “the module”) is a Hardware single-chip cryptographic module. More specifically, the module is considered a sub-chip cryptographic subsystem as defined in IG 2.3.B.

The module has been tested by atsec CST lab on the following platforms:

Model/Part Number(s)	Hardware Version(s)	Firmware Version(s)	Processor(s)	Non-Security Relevant Distinguishing Features
Nuvoton NPCM8mnx Arbel Baseboard Management Controller (BMC)	2.3.8	N/A	ARM Cortex -M4 Core	N/A

Table 2 - Cryptographic Module Tested Configuration

2.1 Mode of Operation

The module only supports approved mode of operation. There are no non-approved but allowed algorithms used in approved mode. There are no non-approved algorithms used in the approved mode with no security claimed. There are no non-approved algorithms used in a non-approved mode.

2.2 Security Functions

The Table 3 below lists all security functions of the module, including specific key strengths employed for approved services, and implemented modes of operation.

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key / Curve / Modulus Size(s)	Use / Function
A4133	AES [SP 800-38 A] [SP 800-38 C]	CBC ECB CCM OFB CFB128	128, 192, 256 bits	AES Encryption and AES Decryption
	AES [SP 800-38 A]	CTR	128, 192, 256 bits	
	AES [SP 800-38 D]	GCM	128, 192, 256 bits	
	AES [SP 800-38 B]	CMAC	128, 192, 256 bits	CMAC Message Authentication Code Generation and CMAC Message Authentication Code Verification

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key / Curve / Modulus Size(s)	Use / Function
	AES [SP 800-38 D]	GMAC	128, 192, 256 bits	GMAC Message Authentication Code Generation and GMAC Message Authentication Code Verification
	HMAC [FIPS 198-1]	HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	256, 384, 512 bits	HMAC Message Authentication Code Generation
	RSA [FIPS 186-4]	RSA-PSS using SHA2-256, SHA2-384 or SHA2-512 RSA-PKCS#1 v1.5 using SHA2-256, SHA2-384 or SHA2-512	2048 or 3072 modulus	RSA Signature Generation, RSA Signature Verification
	KBKDF [SP800-108]	KDF Modes: Counter, Feedback, Double pipeline iteration MAC Modes: HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512	256, 384, 512 bits	Key Derivation Function
	KTS-IFC [SP800-56Brev2]	KTS-OAEP-basic	2048 or 3072 modulus	RSA Key Transport (key wrapping and un-wrapping)
	ECDSA [FIPS 186-4]	B.4.2 Testing Candidates	P-256, P-384, P-521 curves	ECDSA Key Generation
		NA	P-256, P-384, P-521 curves	ECDSA Key Verification
		SHA2-256, SHA2-384, SHA2-512	P-256, P-384, P-521 curves	ECDSA Signature Generation, ECDSA Signature Verification
		N/A	P-256, P-384, P-521 curves	ECDSA Signature Generation Component
	SHS [FIPS 180-4]	SHA2-256 SHA2-384 SHA2-512	N/A	Message Digest Generation
	KAS-ECC-SSC [SP800-56Arev3]	ephemeralUnified	P-256, P-384, P-521 curves	EC Diffie-Hellman Shared Secret Computation (complete)

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key / Curve / Modulus Size(s)	Use / Function
	Hash_DRBG [SP800-90A]	SHA2-512	512	Random Number Generation
Vendor Affirmed	CKG (Cryptographic Key Generation) [SP800-133rev2] [FIPS 186-4]	SP800-133rev2 Section 5.1 and FIPS 186-4: direct output U from approved DRBG; no XOR, no post-processing	N/A	ECDSA Key Generation
E94	ESV [SP800-90B]	N/A	Used to seed the SP800-90Arev1 DRBG	Random Number Generation

Table 3 - Approved Algorithms

2.3 Module Overview

Figure 1 depicts the module's block diagram with a red outline indicating the Tested Operational Environment's Physical Perimeter (TOEPP) of the NPCM8mnx and the blue dotted outline depicting the cryptographic boundary of the sub-chip embedded within the physical perimeter.

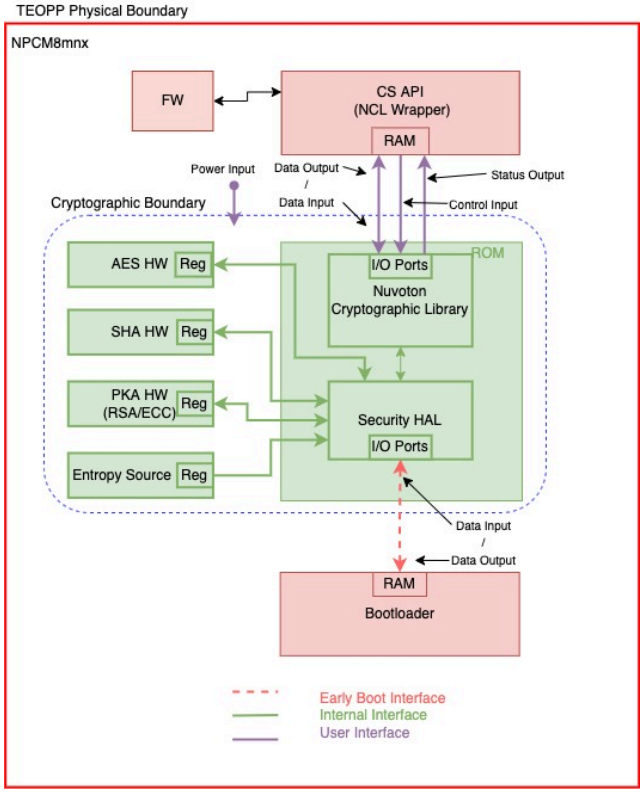

Figure 1 - [Block Diagram]

Figure 2 shows a picture of the NPCM8mnx (BMC) in which the sub-chip module is embedded.

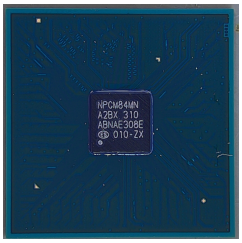

Figure 2: Nuvoton NPCM8mnx

3 Cryptographic Module Ports and Interfaces

The underlying logical interfaces of the module are the module's C language Application Programming Interfaces (APIs). All data input and data output, status ports and control ports are directed through the interface of the module's logical component, which are the APIs while the physical interface is considered the I/O ports of the sub-chip module through which the data input and data output, status output and control input traverse.

Physical Interface	Logical Interface[1]	Data that passes over port/interface
I/O Ports	Data Input	Data inputs are provided in the variables passed in the API and callable service invocations, generally through caller-supplied buffers
I/O Ports	Data Output	Data outputs are provided in the variables passed in the API and callable service invocations, generally through caller-supplied buffers
I/O Ports	Control Input	Control inputs which control the operation of the module are provided through dedicated parameters.
I/O Ports	Status Output	Status output is provided in return codes and through messages. Documentation for each API lists possible return codes. A complete list of all return codes returned by the C language APIs within the module is provided in the header files and the API documentation. Messages are documented also in the API documentation.
Power Port	Power Interface	Power interface is provided internally by TEOPP in which the cryptographic module is embedded.

Table 4 - Ports and Interfaces

[1] The module does not implement a Control Output interface.

4 Roles, services, and authentication

The module supports two authorized roles: A Crypto Officer Role and a User Role. No support is provided for a Maintenance operator. The module does not implement a bypass mode nor concurrent operators.

The Crypto Officer is implicitly assumed. Crypto Officer may be used to facilitate the module's audit functions by invoking the "Get Module Description" or "Show-Status" services. The User can perform any of the other services mentioned in Table 5. The Users of the module are software applications that implicitly assume the User Role when requesting any cryptographic services provided by the module.

FIPS 140-3 does not require authentication mechanism for level 1 modules. Therefore, the module does not implement an authentication mechanism.

The module only implements Approved security functions in an Approved mode. Table 5 below lists services available. The module provides an approved service indicator by receiving a return code of “NCL_STATUS_OK to indicate that the service executed an approved security function.

NOTE: The module does not implement any non-Approved Algorithms (neither with nor without security claim).

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

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

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

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

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

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

Service	Description	Inputs	Outputs	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
AES Encryption	Data Encryption	AES key and plain text	cipher text	AES-CBC AES-ECB AES-CCM AES-OFB AES-CFB128 AES-CTR AES-GCM	AES key	User	W, E	NCL_STATUS_OK
AES Decryption	Data Decryption	AES key and cipher text	plain text	AES-CBC AES-ECB AES-CCM AES-OFB AES-CFB128 AES-CTR AES-GCM	AES key	User	W, E	NCL_STATUS_OK
CMAC Message Authentication Code Generation	Message Authentication Code Generation	AES key and message M	MAC	AES-CMAC	AES key	User	W, E	NCL_STATUS_OK

Service	Description	Inputs	Outputs	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
CMAC Message Authentication Code Verification	Message Authentication Code Verification	MAC and Message	“VALID” or “INVALID”	AES-CMAC	AES key	User	W, E	NCL_STATUS_OK
GMAC Message Authentication Code Generation	Message Authentication Code Generation	AES key, AAD	authentication tag	AES-GMAC	AES key	User	W, E	NCL_STATUS_OK
GMAC Message Authentication Code Verification	Message Authentication Code Verification	AES key, AAD, IV, authentication tag	“PASS” or “FAIL”	AES-GMAC	AES key	User	W, E	NCL_STATUS_OK
HMAC Message Authentication Code Generation	Message Authentication Code Generation	HMAC key and message	MAC	HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	HMAC key	User	W, E	NCL_STATUS_OK
Message Digest Generation	SHS Message Digest Generation	message	digest (hash value)	SHA2-256 SHA2-384 SHA2-512	none	User	N/A	NCL_STATUS_OK
RSA Key Transport (encapsulation)	Key encapsulation using KTS-OAEP-basic	RSA public key and key to be encapsulated	encapsulated key	KTS-IFC	RSA public key	User	W, E	NCL_STATUS_OK
RSA Key Transport (un-encapsulation)	Key Un-encapsulation using KTS-OAEP-basic	RSA private key and key to be un-encapsulated	plaintext key	KTS-IFC	RSA private key	User	W, E	NCL_STATUS_OK
RSA Digital Signature Generation	Digital Signature Generation	RSA private key, message and hash algorithm	signature	RSA-PSS, RSA-PKCS#1 v1.5 Signature Generation, Hash_DRBG	RSA private key	User	W, E	NCL_STATUS_OK
RSA Digital Signature Verification	Digital Signature Verification	RSA public key, signature and hash algorithm	True or False	RSA-PSS, RSA-PKCS#1 v1.5 Signature Verification	RSA public key	User	W, E	NCL_STATUS_OK

Service	Description	Inputs	Outputs	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ECDSA Digital Signature Generation	Digital Signature Generation	ECDSA private key, message and hash algorithm	signature	ECDSA Digital Signature Generation, Hash_DRBG	ECDSA private key	User	W, E	NCL_STATUS_OK
ECDSA Digital Signature Generation Component	Digital Signature Generation Component	ECDSA private key and message digest	signature	ECDSA Digital Signature Generation Component, Hash_DRBG	ECDSA private key	User	W, E	NCL_STATUS_OK
ECDSA Digital Signature Verification	Digital Signature Verification	ECDSA public key, signature and hash algorithm	True or False	ECDSA Digital Signature Verification	ECDSA public key	User	W, E	NCL_STATUS_OK
ECDSA Key Generation	Asymmetric Key Pair Generation	Curve size	generated private and public keys	ECDSA Key Generation, Hash_DRBG, CKG	ECDSA Key pair	User	G, R	NCL_STATUS_OK
EC Diffie-Hellman Shared Secret Computation	Shared Secret Computation using Elliptic Curve Cryptography	received public key and possessed private key	shared secret	KAS-ECC-SSC	ECDH public key	User	W, E	NCL_STATUS_OK
					ECDH private key		E	
					shared secret		G, R	
Key derivation	Perform key derivation	Key material	Derived key	KBKDF	Derived key	User	G, R, E	NCL_STATUS_OK
Random Number Generation	Deterministic Random Number Generation	number of bits	random numbers	Hash_DRBG	Entropy input string, nonce	User	W	NCL_STATUS_OK
					seed, V, and C		G	
Get Module Description	Outputs Module Name + Version Number	None	Module Name + Module Version Number	N/A	None	CO	N/A	N/A

Service	Description	Inputs	Outputs	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
SSP Zeroisation	zeroizes crypto function context and releases memory space	handle of crypto function context	zeroized and released memory space	N/A	All Keys / SSPs	User	Z	N/A
Show-Status	Outputs Operational/ Error status of the module	None	Operational /Error status	N/A	None	CO	N/A	N/A
Self-test[2]	Executes on-demand self-test and outputs Pass/Fail status	None	Pass/Fail status	HMAC-SHA2-512	HMAC Key	User	E	NCL_STATUS_OK
				SHA2-256	N/A			
				AES-CCM	AES Key			
				AES-CBC	AES Key			
				RSA PKCS#1 v1.5 Signature Generation	RSA Private Key			
				RSA PKCS#1 v1.5 Signature Verification	RSA Public Key			
				KBKDF	Key Derivation Key, Derived Key			
				KTS-IFC (encapsulation)	RSA Key Pair, Encapsulated key			
				ECDSA Signature Generation	ECDSA Private Key			
				ECDSA Signature Verification	ECDSA Public Key			

[2]Keys and SSPs used in this service are hard-coded in the module and used exclusively for self-tests.

Service	Description	Inputs	Outputs	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
				KAS-ECC-SSC	ECDH Key Pair, Shared Secret			
				Hash_DRBG	Seed			

Table 5 - Approved Services

5 Software/Firmware Security

5.1 Software/Firmware Integrity Technique

The module’s executable code is programmed in a masked ROM which is a type of Read-Only Memory (ROM) where content is programmed by the integrated circuit manufacturer during the silicon manufacturing (rather than by the Operator of the module). The memory technology is non reconfigurable memory as defined in IG 5.A, which will not have any change or degradation of data for a minimum of 10 years after manufactured date. As such, it is considered a hardware only module with a non-modifiable operational environment. The requirements of this area are not applicable to the module.

6 Operational Environment

The Nuvoton Cryptographic Library 2.3 operates in a non-modifiable operational environment. The module is programmed by the manufacturer during the silicon manufacturing (rather than by the user). It maintains its own memory region which can only be accessed by the module. There is no additional application present within the operating environment. The module does not spawn any cryptographic processes. The operational environments in which the module was tested are listed in Table 2.

7 Physical Security

The Nuvoton Cryptographic Library 2.3 cryptographic module is a Hardware cryptographic module in a single-chip embodiment. More specifically, the module is considered a sub-chip cryptographic subsystem.

The module consists of production-grade components that include standard passivation techniques (e.g., a conformal coating applied over the module’s circuitry to protect against environmental or other physical damage). The module does not implement a maintenance role and has no maintenance access interface.

8 Non-invasive Security

Currently, the non-invasive security is not required by FIPS 140-3 (see NIST SP 800-140F). The requirements of this area are not applicable to the module.

9 Sensitive Security Parameter Management

The following table summarizes the keys and Sensitive Security Parameters (SSPs) that are used by the cryptographic services implemented in the module. Modification of PSPs by unauthorized operators is prohibited.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import /Export	Establishment	Storage	Zeroization	Use & related keys
AES key	128, 192, 256 - bits of security strength	AES CAVP Cert. #A4133	Not Applicable. The key is entered via API parameter	Entry: The key is entered into the module within the TOEPP[3] via API input parameters in plaintext. Output: N/A	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: AES Data Encryption and Decryption Related Keys: N/A
RSA private and public key	112 to 128 bits of security strength	KTS-IFC CAVP Cert. #A4133	Not Applicable. The key is entered via API parameter	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: The key is output from the module within the TOEPP via API output parameters in plaintext	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: Key Encapsulation and Un-encapsulation Related Keys: Used to establish Encapsulated key
Encapsulated key	112 to 128 bits of security strength	KTS-IFC CAVP Cert. #A4133	N/A	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: The key is output from the module within the	Established by KTS-IFC	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: Established by KTS-IFC Related Keys: Established using RSA private and public keys

[3] TOEPP - Tested Operational Environment's Physical Perimeter

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import /Export	Establishment	Storage	Zeroization	Use & related keys
				TOEPP via API output parameters in plaintext				
RSA private and public key pair	112 to 128 bits of security strength	RSA CAVP Cert. #A4133	Not Applicable. The key is entered via API parameter	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: The key is output from the module within the TOEPP via API output parameters in plaintext	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: Signature Generation and Verification Related Keys: N/A
ECDSA private and public key pair	128 to 256 bits of security strength	ECDSA CAVP Cert. #A4133	The private keys can be generated using FIPS186-4 Key Generation method, and the random value used in the key generation is generated using SP800-90Arev1 DRBG	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: The key is output from the module within the TOEPP via API output parameters in plaintext	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: Key Generation and Verification, Signature Generation and Verification **Related Keys**: Generated using DRBG internal state
HMAC key	112 or greater bits of security strength	HMAC CAVP Cert. #A4133	Not Applicable. The key is entered via API parameter	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: N/A	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: Hashed Message Authentication Code Generation Related Keys: N/A

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import /Export	Establishment	Storage	Zeroization	Use & related keys
ECDH key pair (including intermediate key generation values)	128 to 256-bits of security strength	EC keygen CAVP Cert. #A4133	The private keys are generated using FIPS186-4 Key Generation method, and the random value used in the key generation is generated using SP800-90Arev1 DRBG	Entry: The public key is entered into the module within the TOEPP via API input parameters in plaintext. Output: The key is output from the module within the TOEPP via API output parameters in plaintext	N/A	Volatile memory	automatic zeroization when structure is deallocated or when the system is powered down.	**Use**: ECDH Shared Secret Computation **Related Keys**: Generated using DRBG internal state, Used to establish EC Diffie-Hellman Shared Secret
ECC Shared Secret		KAS-ECC-SSC CAVP Cert. #A4133	N/A	Entry: N/A Output: The key is output from the module within the TOEPP via API output parameters in plaintext	Established by KAS-ECC-SSC			**Use**: ECDH Shared Secret Computation **Related Keys**: Established from ECDH key pair
Derived key	256, 384, 512 bits	KBKDF CAVP Cert. #A4133	Derived by SP 800-108 KBKDF	Entry: N/A Output: The key is output from the module within the TOEPP via API output parameters in plaintext	N/A			**Use**: Key derivation **Related Keys**: Derived from Key Derivation Key
Key Derivation Key	256, 384, 512 bits	KBKDF CAVP Cert. #A4133	The key can be entered via API parameters, or generated using SP800-90Arev1 DRBG	Entry: The key is entered into the module within the TOEPP via API input parameters in plaintext. Output: N/A	N/A			**Use**: Key derivation **Related Keys**: Used to derive Derived key

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import /Export	Establishment	Storage	Zeroization	Use & related keys
Entropy Input String + Nonce	256-bits of security strength	Entropy Source ESV Cert. E94	N/A	Entry: N/A Output: N/A	N/A			**Use**: Random Number Generation **Related Keys**: DRBG internal state, Seed
DRBG internal state (i.e., Hash_DRBG V and C values), Seed	256-bits of security strength	Hash DRBG CAVP Cert. #A4133	Derived from entropy input string as defined by SP800-90Arev1	Entry: N/A Output: N/A	N/A			**Use**: Random Number Generation **Related Keys**: Entropy Input String + Nonce

Table 6 - SSPs

9.1 Random Number Generation

The module employs a Hash_DRBG using a SHA-512 PRF. Per section 10.1.1.1 of [SP800-90A], the internal state of the Hash_DRBG is the V, C, and reseed counter. The Hash_DRBG is seeded by an SP800-90B Entropy Source for which the estimated amount of entropy is ~0.6/bit. The DRBG is seeded with 1024-bits of entropy input thereby providing 256-bits of entropy during initialization and reseeding. The DRBG internal state is not accessible by non-DRBG functions. All random values used by approved security functions, SSP generation, or SSP establishment method are provided by the Hash_DRBG.

Entropy Source	Minimum number of bits of entropy	Details
E94	256-bits strength	The module includes SP800-90B compliant entropy source based on Ring Oscillators implemented in hardware TRNG. When output is requested from the entropy source, the entropy source fills a 1024-bit buffer with random bits obtained with a single request for entropy data. All 1024-bits are then provided as output from the entropy source.

Table 7 - Non-Deterministic Random Number Generation Specification

9.2 Key/SSP Generation

The module generates Keys and SSPs in accordance with FIPS 140-3 IG D.H. The cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys as per [SP800-

133rev2] (vendor affirmed), compliant with [FIPS186-4] and using DRBG compliant with [SP800-90Arev1]. A seed (i.e., the random value) used in asymmetric key generation is obtained from [SP800-90Arev1] DRBG as described in Section 4 of [SP800-133rev2]. The key generation service for ECDSA, as well as the [SP 800-90Arev1] DRBG have been ACVT tested with algorithm certificates found in Table 3.

9.3 Key/SSP Establishment

The module provides the following key/SSP establishment services:

1. The module implements a key-based key derivation method compliant with SP800-108.
2. The module implements KAS-ECC-SSC EC Diffie-Hellman Shared Secret Computation compliant to [SP800-56Arev3] and IG D.F Scenario (2) path (1).
    - The shared secret computation provides between 128 and 256 bits of encryption strength.
3. Within the TOEPP, the module offers RSA key wrapping and unwrapping using KTS-OAEP-basic scheme. The implementation supports 2048 and 3072 modulus size, with both key encapsulation and un-encapsulation supported. The module does not implement key confirmation. See section 11.2 for operator guidance details.
    - The SSP establishment methodology provides 112 or 128 bits of encryption strength.

9.4 Key/SSP Entry and Output

Keys/SSPs entered or output the module are electronically entered in plaintext form from the invoking User firmware running on the same device. No Keys/SSPs are entered or output from the module to outside the TOEPP. According to IG 2.3.B, *Transferring SSPs including the entropy input between a sub-chip cryptographic subsystem and an intervening functional subsystem for Security Levels 1 and 2 on the same single chip is considered as not having Sensitive Security Parameter Establishment crossing the HMI of the sub-chip module per IG 9.5.A*.

9.5 Key/SSP Storage

The module does not provide persistent storage for keys/SSPs. Keys/SSPs are stored in volatile memory only and are received for use by the module only at the request of the User firmware.

9.6 Key/SSP Zeroization

The module includes different methods for zeroization:

- Implicit Zeroization which is done automatically by the module when a SSP is no longer needed. Specifically, when completing any service call before the module returns the control to the caller, the module clears out any SSPs that are no longer needed by writing zeros using the memset() function.

- Explicit Zeroization which can be invoked by the caller using NCL_*_Clear (where '*' is the one of the approved algorithms implemented in the module) which deallocates the structure associated with the cipher and overwriting the SSP values with all zeros OR the caller can also power down the device to zeroize all SSPs.

While zeroization is being performed, the interface with the module is inhibited and the successful completion of a requested zeroization service suffices as the implicit indicator that zeroization has completed.

10 Self-tests

Self-tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected. While the module is executing the self-test, no services are available, and input and output are inhibited. The module will boot only after successfully passing the SHA2-256, HMAC-SHA2-512 and KBKDF-HMAC-SHA2-256 CASTs. If an error is detected in any self-test, the module will enter the Error State.

10.1 Pre-Operational Self-Tests

The module is solely implemented in hardware (i.e., only contains executable code that is stored in non- reconfigurable masked ROM[4]). As such, the module does not perform any pre-operational software/firmware integrity test, but instead performs a Cryptographic Algorithm Self-Test on the SHA2-256, HMAC-SHA2-512 and KBKDF-HMAC-SHA2-256 algorithms when the module is powered on.

The module does not implement a pre-operational bypass test nor pre-operational critical functions test.

10.2 Conditional Self-Tests

The module performs a conditional self-test when the conditions specified for the following tests occur:

Conditional Cryptographic Algorithm Self-Test
Conditional Pair-Wise Consistency Test

The module does not implement a Software/Firmware Load Test, Manual Entry Test, Conditional Bypass Test nor Conditional Critical Functions Test.

10.2.1 Conditional Cryptographic Algorithm Self-Tests

The module conducts conditional cryptographic algorithm self-test prior to the first operational use of each cryptographic algorithm (with the exception of KBKDF, HMAC and SHA CASTs, which are performed at power on). The table below describe the conditional tests supported by the module.

Algorithm	Test
HMAC	HMAC-SHA2-512 MAC Generation KAT using 160-bit key
SHA	SHA2-256 Message Digest KAT
KBKDF	Counter mode using HMAC-SHA2-256 using 160-bit key
AES	AES-CCM Encryption KAT using 128-bit key AES-CBC Decryption KAT using 128-bit key
KTS-IFC	KTS-OAEP-basic Encryption KAT with 2048 -bit key and SHA2-256 KTS-OAEP-basic Decryption KAT with 2048 -bit key and SHA2-256

[4] A masked ROM is a type of Read-Only Memory (ROM) where content is programmed by the integrated circuit manufacturer during the silicon manufacturing.

Algorithm	Test
RSA	PKCS#1 v1.5 Signature Generation KAT with 2048 -bit key and SHA2-256 PKCS#1 v1.5 Signature Verification KAT with 2048 -bit key and SHA2-256
ECDSA	ECDSA Signature Generation KAT with P-256 curve and SHA2-256 ECDSA Signature Verification KAT with P-256 curve and SHA2-256
KAS-ECC-SSC	• ECDH shared secret computation KAT with P-256 curve
Hash_DRBG	• Hash_DRBG random number generation KAT using predefined seed.
ENT	• RCT (Repetition Count Test) • APT (Adaptive Proportion Test) • Startup self-tests with 1024-bit samples (Same process as the two continuous tests shown above)

Table 8 - Conditional Cryptographic Algorithm Self-Tests

10.2.2 Conditional Pair-Wise Consistency Test

The module performs a pair-wise consistency test on when a new ECDSA key pair is generated. The pair-wise consistency test is performed by calculating a digital signature and then verifying it. If the signature cannot be verified, the pair-wise consistency test will fail.

10.2.3 Periodic Self-Test

During runtime, operators can initiate the conditional self-tests on demand by calling *NCL_MISC_SelfTest* and passing the algorithm as an argument.

The module's entropy source is powered on only momentarily to seed the module's SP800-90Arev1 DRBG. The module performs ENT health tests defined in Section 4 of SP800-90B on the generated output prior to seeding the SP800-90Arev1 DRBG. After completing its execution, the entropy source powers down.

10.3 Self-Test Error Handling

For any of the conditional self-tests, the module enters an error state upon failing the self-test. A failure in the conditional CAST or conditional PCT results in "*NCL_STATUS_FAIL*". Likewise, a failure of the ENT health tests will result in an "*ENTROPY_SRC_ERROR"* status returned to the user. When in the error state, no cryptographic services are provided. The control and data output interfaces are prohibited while in the error state. The only method to clear this error state is to power cycle the device and then successfully pass the conditional self-tests.

Cause of Error	Status Indicator
failure in conditional self-test (conditional CAST or conditional PCT)	NCL_STATUS_FAIL

Cause of Error	Status Indicator
failure of the ENT health test	ENTROPY_SRC_ERROR

Table 9 - Error States

11 Life-cycle assurance

11.1 Delivery and Operation

As explained in Section 10.1.1, the module is placed in a masked ROM by manufacturer during the silicon manufacturing. The module is delivered as part of the Nuvoton NPCM8mnx platform (listed in Table 2). During manufacturing – each chip is tested to make sure the masked ROM was manufactured correctly; this is done using CRC32 algorithm on the entire masked ROM code on each device before it is shipped out.

The hardware version can be found printed on the chip package. The last three letters of the printed version may be represented with the following:

- m: # of cores
- n: GFX (0=No, 5=yes)
- x: Customer

During execution – As part of the device boot process, the code is verified by a dedicated hardware inside the chip that checks every byte of code compared to a known parity bit. If any byte fails, the parity test then an internal error is generated; the error is handled by the application (User) firmware.

11.2 Crypto Officer Guidance

11.2.1 Configuration

The module is configured to be operational by default. If the device starts up successfully and has successfully passed the SHA2-256, HMAC-SHA2-512 and KBKDF-HMAC-SHA2-256 CASTs, it is operating correctly and can begin servicing User requests.

11.2.2 End of Life

Once the module reaches its end-of-life stage (End of Life (EOL) date for the Nuvoton device is 10 years from manufacturing date) or sanitation is initiated by the module's Operator, it is the Operator's responsibility to clear all existing SSPs from the module. This can be achieved by either performing a full device reset, or by explicitly invoking the following sequence of APIs to clear the data from all modules:

- NCL_SHA_Clear - For each of existing SHA, HMAC and KBKDF contexts
- NCL_DRBG_Clear - For each of existing DRBG contexts
- NCL_AES_Clear - For each of existing AES contexts
- NCL_RSA_Clear - For each of existing RSA contexts
- NCL_ECC_Clear - For each of existing ECDSA and ECDH contexts

11.2.3 AES-GCM

The module's AES-GCM implementation conforms to IG C.H scenario 2. The module uses the approved Hash_DRBG to generate the IV with a length of 96-bits. The entropy source producing the DRBG seed is located inside the module's cryptographic boundary.

11.2.4 RSA Key Wrapping

To comply with SP800-56Brev2 assurances found in its Section 6 (specifically SP800-56Brev2 *Section 6.4 Required Assurances*) The entity using the IUT must obtain required assurances listed in section 6.4 of SP 800-56BRev2 by performing the following steps:

1. The entity requesting the RSA key unwrapping (un-encapsulation) service from the module, shall only use an RSA private key that was generated by an active FIPS validated module that implements FIPS 186-4 compliant RSA key generation service and performs the key pair validity and the pairwise consistency as stated in section 6.4.1.1 of the SP 800-56BRev2. Additionally, the entity shall renew these assurances over time by using any method described in section 6.4.1.5 of the SP 800-56BRev2.
2. For use of an RSA key wrapping (encapsulation) service in the context of key transport per IG D.G, the entity using the module, shall verify the validity of the peer's public key using any method specified in section 6.4.2.1 of the SP 800-56BRev2.
3. The entity using the module, shall confirm the peer's possession of private key by using any method specified in section 6.4.2.3 of the SP 800-56BRev2.

12 Mitigation of other attacks

The module does not implement security mechanisms to mitigate other attacks.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
ACVP	Algorithm Certification Validation Program
CBC	Cipher Block Chaining
CAST	Cryptographic Algorithm Self-Test
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cipher Feedback
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CTR	Counter Mode
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ESV	Entropy Source Validation
EOL	End Of Life
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
KAS	Key Agreement Scheme
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
OFB	Output Feedback
PSS	Probabilistic Signature Scheme
RSA	Rivest, Shamir, Addleman
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SSC	Shared Secret Computation
TOEPP	Tested Operational Environment’s Physical Perimeter

Appendix B. References

FIPS140-3	FIPS PUB 140-3 - Security Requirements For Cryptographic Modules March 2019 https://doi.org/10.6028/NIST.FIPS.140-3
FIPS140-3_IG	Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program November 2023 https://csrc.nist.gov/CSRC/media/Projects/cryptographic-module-validation-program/documents/fips140-3/FIPS 140-3 IG.pdf
FIPS180-4	**Secure Hash Standard (SHS)** March 2012 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.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
RFC3394	**Advanced Encryption Standard (AES) Key Wrap Algorithm** September 2002 http://www.ietf.org/rfc/rfc3394.txt
RFC5649	**Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm** September 2009 http://www.ietf.org/rfc/rfc5649.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
SP800-38B	**NIST Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication** May 2005 http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf
SP800-38C	NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality May 2004 http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf

SP800-38D	NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC November 2007 http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf
SP800-38F	**NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping** December 2012 http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf
SP800-56Arev3	NIST Special Publication 800-56A Revision 3 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography April 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf
SP800-56Brev2	**Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography** March 2019 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf
SP800-90Arev1	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 Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation** January 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf
SP800-133rev2	NIST Special Publication 800-133 - Recommendation for Cryptographic **Key Generation** December 2012 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf
SP800-140B	NIST Special Publication 800-140B - CMVP Security Policy Requirements March 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-140B.pdf