Century Longmai Technology Co. Ltd

mToken CryptoID

FIPS 140-3 Non-Proprietary Security Policy

Document Version: 1.1
Date: 2024-10-15

Table of Contents

List of Tables

List of Figures

1 – General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for the mToken CryptoID cryptographic module. It contains specific 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 3 module.

In this document, the terms “token”, “cryptographic module” or “the module” are used interchangeably to refer to the mToken CryptoID.

1.2 Security Levels

The FIPS 140-3 security levels for the Module are as follows from Table 1:

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

Table 1: Security Levels

2 – Cryptographic Module Specification

2.1 Description

The Longmai mToken CryptoID is a new generation smartcard chip based two-factor authentication device utilizing CCID drivers. The smartcard chip design utilizes the built-in mCOS to communicate with a General Purpose Computer (GPC) via the USB interface in a “plug-and-play” manner. The mToken CryptoID implements the USB Circuit Cards Interface Device (CCID) protocol to communicate with the host application running on a computer device. CCID drivers work to protect the device and are less susceptible to packet sniffing thus providing stronger authentication. The Application Protocol Data Unit (APDU) command-response protocol is transferred via the USB interface and is compatible with ISO/IEC 7816-4 standards. It provides the security services needed to interact with the Public Key Infrastructure (PKI) applications, including Digital Signature Generation/Verification for online authentication and Data Encryption/Decryption for online transactions. Here is the list of main functions provided by the Module:

- Transportation Management
- File System Management
- Secret PIN Management
- Access Control Management

- Session Key Management
- Key Pair Management
- Data Digest Management
- Algorithm Management
- Algorithm Hardware Engine

Purpose and Use:

The Module is intended to be used by Certificate Authorities, Digital Signature Service Providers or other markets that require FIPS 140-3 validated hardware cryptographic module. The Module is intended to be used in a General Purpose Computer (GPC) within the following temperature range: -30°C ~ 80°C, voltage range 2.7V~5.8V.

Module Type: Hardware

Module Embodiment: MultiChipStand

Module Characteristics:

- ✓ **Offers multiple application security Authentication support**
- ✓ mToken CryptoID supports standard smart card applications like Windows smart card logon, VPN, Bit Locker.etc.
- ✓ **Smart Card Based**
  Utilizes 32-bit smart card technology enabling smart card-based authentication and strong authentication.
- ✓ **Plug-and-Play**
  mToken CryptoID is certified from Microsoft HCK/HLK and automatically installs drivers from Microsoft Windows Update.
- ✓ **Unique Global Hardware ID**
  Employs both a unique global hardware ID and user-defined 32-bit software ID for device own identification.
- ✓ **Cryptography Standards Compliant**
  Microsoft SmartCard Mini Driver
  Microsoft CryptoAPI
  PKCS# 11 V2.20
  X509 v3 certificate storage
  SSL V3, IPSEC/KEC, PC/SC, CCID
- ✓ **Multiple Platform Compatibility**
  Support across various OS platform including Windows **Server 2003, Vista and 7,8,10**,11, **Mac OS X, and Linux**.

Cryptographic Boundary:

The physical form of the Module is depicted below in Figure 1 and the block diagram is depicted in Figure 2 below. The Module is a hardware module with a multi-chip standalone embodiment. The cryptographic boundary of the Module is defined by the hard, semi-transparent, polycarbonate (plastic) or metal casing of the USB token. The Module is comprised of a SCC-XE microcontroller sitting atop a Printed Circuit Board (PCB). The PCB carries the signals and instructions of the microcontroller to the other components contained within the Module. All cryptographic functions and firmware are stored within the microcontroller package and executed by a 32-bit CPU (Core Processing Unit).

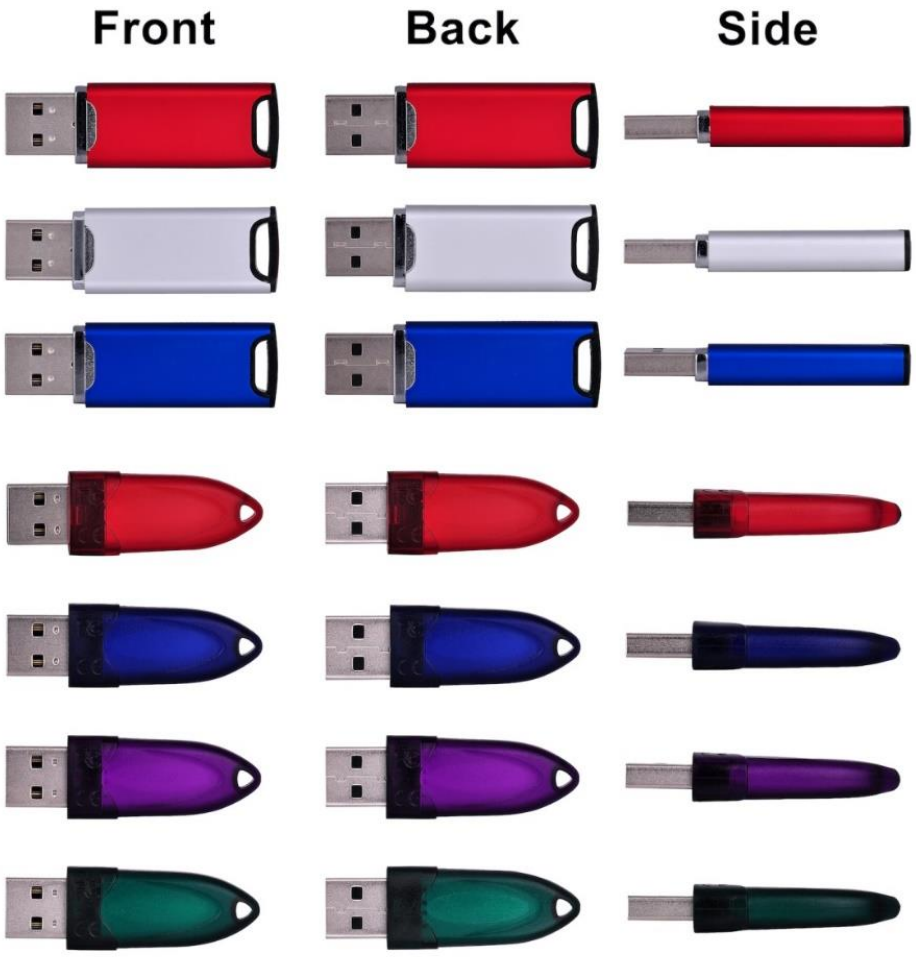

Figure 1 - mToken CryptoID-K9 and mToken CryptoID-A3

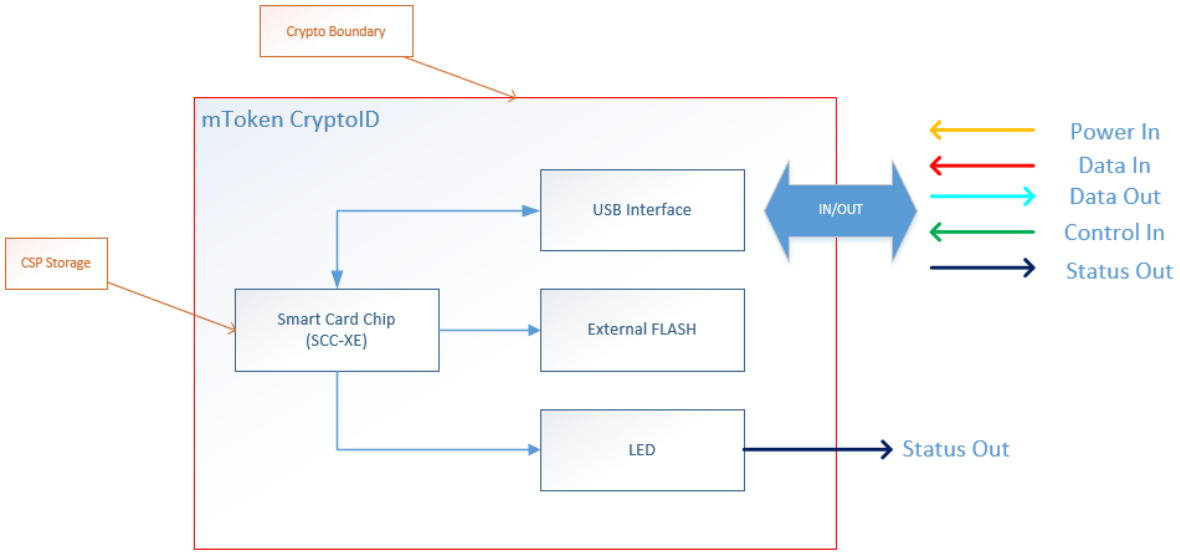

Figure 2 – Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Hardware:

The Module operates in a non-modifiable operational environment and does not implement a General Purpose Operating System. Once the firmware of the Module is loaded on the mToken CryptoID, it cannot be modified or erased, firmware cannot be upgraded/updated. The operational environment requirements do not apply to the Module.

Model and/or Part Number	Hardware Version	Firmware Version	Processors	Features
mToken CryptoID-K9	SCC-XE-K9	3.12	AisinoChip Electronics SCC-XE, ARM Cortex M0	Metal enclosure
mToken CryptoID-A3	SCC-XE-A3	3.12	AisinoChip Electronics SCC-XE, ARM Cortex M0	Plastic enclosure

Table 2: Tested Module Identification – Hardware

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

N/A for this module.

Tested Module Identification – Hybrid Disjoint Hardware:

N/A for this module.

Tested Operational Environments - Software, Firmware, Hybrid:

N/A for this module.

Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid:

N/A for this module.

2.3 Excluded Components

NOTE: No Components were excluded from the cryptographic boundary

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	CO can set the module to Approved Mode	Approved	Steady green LED is on and the red LED is off. Returns "01" from the calling GetData APDU command
Non-Approved mode	By default, the module will be set to Non-Approved mode by manufacturer.	Non-Approved	Green LED is off and the steady red LED is on Returns "00" from the calling GetData APDU command

Table 3: Modes List and Description

Configuration of the Approved Mode of Operation

The mToken CryptoID module is originally non-compliant and must be configured to operate in an approved mode of operation. The Approved mode of operation is configured by the Issuer (CO) prior to being distributed to the end users.

Please see Section 11.1 Installation, Initialization, and Startup Procedures for initial Approved mode configuration.

When the Module operates in Approved mode, the steady green LED is on, and the red LED is off. The module returns “01” from the calling GetData APDU command which indicates that the module is in Approved mode, and the module returns “0000000000000000” to the calling GetHealthStatus APDU command which indicates that all self-tests have completed successfully.

In Approved mode, only approved algorithms are allowed and available to the services.

Please refer to the LED status table for the details of the LED status indicator of the module.

Configuration of the Non-Approved Mode of Operation
The Module is sent from manufacturing to the Issuer (CO) in Non-Approved mode.

When the Module operates in Non-Approved mode, the green LED is off and the steady red LED is on. The Module returns “00” from the calling GetData APDU command which indicates that the module is in Non-Approved mode, and the module returns “0000000000000000” to the calling GetHealthStatus APDU command which indicates that the self-test have completed successfully. The GetData APDU command returns an error when the Module is uninitialized.

Both Approved services and Non-Approved services are listed in Table 19: Approved Services and Table 20: Non-Approved Services are allowed in Non-Approved mode.

Mode Change Instructions and Status [O]:

Once the Module is initialized, the Issuer can switch the mode of operation by calling the ChangeWorkingMode APDU command with “P1” parameter. If “P1” is 1, the module is configured to operate in Approved mode; if “P1” is 0, the module is configured to operate in Non-Approved mode.

The Module is zeroized as part of the switching process preventing the sharing of CSPs between modes. After the successful completion of the pre-operational self-tests, the Module automatically enters Approved mode or Non-Approved mode based on the value of the “P1” parameter.

2.5 Algorithms

Approved Algorithms:

The Module implements the Approved cryptographic algorithms listed in the table below.

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A2660	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A2660	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-ECB	A2660	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-KW	A2660	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
ECDSA KeyGen (FIPS186-4)	A2663	Curve - P-256, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A2663	Curve - P-256, P-521	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A2663	Component - No Curve - P-256, P-521	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A2663	Component - No Curve - P-256, P-521	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
Hash DRBG	A2662	Prediction Resistance - No Mode - SHA2-256	SP 800-90A Rev. 1
HMAC-SHA2-256	A2661	Key Length - Key Length: 128-256 Increment 8	FIPS 198-1
HMAC-SHA2-384	A2661	Key Length - Key Length: 128-256 Increment 8	FIPS 198-1
HMAC-SHA2-512	A2661	Key Length - Key Length: 128-256 Increment 8	FIPS 198-1
KAS-ECC Sp800-56Ar3	A2659	Domain Parameter Generation Methods - P-256, P-521 Function - Key Pair Generation Scheme - ephemeralUnified - KAS Role - Initiator, Responder Key Length - 256	SP 800-56A Rev. 3
RSA KeyGen (FIPS186-4)	A2663	Key Generation Mode - B.3.3 Modulo - 2048 Primality Tests - Table C.2 Private Key Format - Standard	FIPS 186-4
RSA SigGen (FIPS186-4)	A2663	Signature Type - PKCS 1.5 Modulo - 2048	FIPS 186-4
RSA SigVer (FIPS186-4)	A2663	Signature Type - PKCS 1.5 Modulo - 2048	FIPS 186-4
SHA2-256	A2661	-	FIPS 180-4
SHA2-384	A2661	-	FIPS 180-4
SHA2-512	A2661	-	FIPS 180-4

Table 4: Approved Algorithms

Note: KAS [56Ar3] - 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. without key confirmation.

Vendor-Affirmed Algorithms:

The Module implements the Approved Vendor Affirmed cryptographic algorithms listed in the Table below.

Name	Properties	Implementation	Reference
CKG [IG D.H]	[133] Sections 4 and 5.1 Asymmetric signature key generation using unmodified DRBG output:AisinoChip Electronics SCC-XE [133] Sections 4 and 5.2 Asymmetric key establishment key generation using unmodified DRBG output:AisinoChip Electronics SCC-XE [133] Sections 4 and 6.1 Direct symmetric key generation using unmodified DRBG output:ARM Cortex M0 [133] Section 6.2.1 Derivation of symmetric keys	mToken CryptoID Core DRBG Component	Key Generation

Name	Properties	Implementation	Reference
	from a key agreement shared secret.:ARM Cortex M0		

Table 5: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

Note: The Module does not implement any Non-Approved, Algorithms Allowed in the Approved Mode of Operation.

N/A for this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

Note: The Module does not implement any Non-Approved, Algorithms Allowed with No Security Claimed in the Approved Mode of Operation.

N/A for this module.

Non-Approved, Not Allowed Algorithms:

The Module implements the Non-Approved, Not Allowed cryptographic algorithms listed in the table below.

Name	Use and Function
SHA-1	Hashing (not CAVP tested)
RSA	Digital Signature Generation and Verification using 1024 bit keys
KTS	RSA-based Key Wrapping
HMAC-SHA-1	Message Authentication Code (not CAVP tested)
Triple-DES	Encryption and Decryption (not CAVP tested)
CMAC with Triple-DES	Message Authentication Code

Table 6: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

The table below shows the Security Function Implementations that the module implements:

Name	Type	Description	Properties	Algorithms
KAS1	KAS-Full	Key Agreement	Caveat:Key establishment method provides	KAS-ECC Sp800-56Ar3: (A2659) Notes: Ephemeral

Name	Type	Description	Properties	Algorithms
			between 128 and 256 bits of encryption strength	Unified (Initiator, Responder), KPG, Partial with oneStepKdf Curves: P-256, P-521 SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
KTS1	KTS-Wrap	Key Transport	Caveat:Key establishment method provides between 128 and 256 bits of encryption strength	AES-KW: (A2660) Security Strength: 128, 192, 256
KTS2	KTS-Wrap	Key Transport	Publications:[IG D.G] Caveat:Key establishment method provides between 128 and 256 bits of encryption strength	AES-ECB: (A2660) Security Strength: 128, 192, 256 AES-CMAC: (A2660) Security Strength: 128, 192, 256
SymGen1	CKG	Symmetric Key Generation	Publications:[IG D.H], SP800-133r2 Ref: Section 6.1	AES-ECB: (A2660) Security Strength: 128, 192, 256 AES-CBC: (A2660) Security Strength: 128, 192, 256
AsymGen1	AsymKeyPair-KeyGen	Asymmetric Key-Pair Generation		ECDSA KeyGen (FIPS186-4): (A2663) Curves: P-256, P-521 Hash DRBG: (A2662)
AsymVer1	AsymKeyPair-KeyVer	Asymmetric Key-Pair Verification		ECDSA KeyVer (FIPS186-4): (A2663) Curves: P-256, P-521 Hash DRBG: (A2662)
SigGen1	DigSig-SigGen	Digital Signature Generation		ECDSA SigGen (FIPS186-4): (A2663) Curves: P-256, P-

Name	Type	Description	Properties	Algorithms
				521 Hash DRBG: (A2662) SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
SigVer1	DigSig-SigVer	Digital Signature Verification		ECDSA SigVer (FIPS186-4): (A2663) Curves: P-256, P-521 Hash DRBG: (A2662) SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
AsymGen2	AsymKeyPair-KeyGen	Asymmetric Key-Pair Generation		RSA KeyGen (FIPS186-4): (A2663) Size: n=2048 Hash DRBG: (A2662)
SigGen2	DigSig-SigGen	Digital Signature Generation		RSA SigGen (FIPS186-4): (A2663) Size: n=2048 Hash DRBG: (A2662) SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
SigVer2	DigSig-SigVer	Digital Signature Verification		RSA SigVer (FIPS186-4): (A2663) Size: n=2048 Hash DRBG: (A2662) SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
SigVer3	DigSig-SigVer	Digital Signature Verification		ECDSA SigVer (FIPS186-4):

Name	Type	Description	Properties	Algorithms
				(A2663) Curve: P-256 Hash DRBG: (A2662) SHA2-256: (A2661)
RBG	DRBG	Random Number Generation		Hash DRBG: (A2662) Security Strength: 256 SHA2-256: (A2661)
Entropy	ENT-ESV	Entropy Source		Hash DRBG: (A2662) Security Strength: 256
Hash	SHA	Secure Hash		SHA2-256: (A2661) SHA2-384: (A2661) SHA2-512: (A2661)
Message Authentication	MAC	Message Authentication		HMAC-SHA2-256: (A2661) HMAC-SHA2-384: (A2661) HMAC-SHA2-512: (A2661) AES-CMAC: (A2660) Security Strength: 128, 192, 256
AES Encryption	BC-Auth	Block Cipher		AES-CBC: (A2660) Security Strength: 128, 192, 256 AES-ECB: (A2660) Security Strength: 128, 192, 256
AES Decryption	BC-UnAuth	Block Cipher		AES-CBC: (A2660) Security Strength: 128, 192, 256 AES-ECB: (A2660) Security Strength: 128, 192, 256
SymGen2	CKG	Symmetric Key Generation	Publications:[IG D.H], SP800-133r2 Ref: Section 6.2.1	AES-CBC: (A2660) Security Strength: 128, 192, 256 AES-ECB: (A2660)

Name	Type	Description	Properties	Algorithms
				Security Strength: 128, 192, 256

Table 7: Security Function Implementations

2.7 Algorithm Specific Information

Note: There is no Algorithm Specific Information.

2.8 RBG and Entropy

Cert Number	Vendor Name
E44	Century Longmai Technology Co. Ltd

Table 8: Entropy Certificates

The Module uses the following entropy source**:**

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
mToken CryptoID Core	Physical	AisinoChip Electronics SCC-XE	1bit	0.222745 bits	N/A

Table 9: Entropy Sources

2.9 Key Generation

For Key Generation, see Section 2.5 and Section 2.6 above.

2.10 Key Establishment

For Key Establishment, see Section 2.5 and Section 2.6 above.

2.11 Industry Protocols

Note: The Module does not provide any industry protocols.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

The Module's ports and associated FIPS defined logical interface categories are listed in the table below.

Physical Port	Logical Interface(s)	Data That Passes
LED	Status Output	N/A
USB Data pins	Data Input Data Output Control Input Status Output	All logical data
USB Power pin	None	N/A

Table 10: Ports and Interfaces

The mToken CryptoID uses an LED to encode various status conditions as shown in the table below:

Table 11 – LED status

Condition	Green LED	Red LED
Power Off	OFF	OFF
Self-test in progress	ON	ON
Approved mode	ON	OFF
Approved mode error	BLINK (200ms)	OFF
Switching to Non-Approved mode	SLOW BLINK (1000ms)	OFF
Non-Approved mode	OFF	ON
Non-Approved mode error	OFF	BLINK (200ms)
Switching to Approved mode	OFF	SLOW BLINK (1000ms)

4 Roles, Services, and Authentication

4.1 Authentication Methods

The mToken CryptoID supports Identity-Based Authentication to authenticate the operators.

The Module supports External Authentication with KEY_EA which authenticates the host application running on a computer to the module. The Module also supports Internal Authentication with KEY_IA which the module authenticates itself to the host application.

The Issuer is authenticated by providing the Main Control Key (KEY_MC) with ExternalAuth APDU command. Only the manufacturer or the distributor can be authenticated as an Issuer. When the Module is delivered to the end user, they can authenticate themselves by providing their KEY_PIN (i.e., Admin PIN or User PIN) with Verify PIN APDU command. The operators can logout by calling the ClearSecureState APDU command or by unplugging the module from the GPC.

The Module uses the "challenge-response" method for all types of authentications.

For key-based authentication, the host application calls the GetChallenge command to get a random value from the Module that has the same length as the authenticated keys. The host application then encrypts the random value with the AES-256 symmetric algorithm and transmits the ciphertext to the module for verification. The Module decrypts the ciphertext with the AES-256 symmetric algorithm and compares the result with the original random value from the GetChallenge command. If the results are the same, the authentication is completed successfully; otherwise, the authentication fails, and the Module will decrement the value pre-defined in "LeftRetryTimes". Once the value stored in "LeftRetryTimes" reaches 0, the key is locked. The authentication data uses default keys and are required to be changed upon first login.

For PIN-based authentication, the host application calls the GetChallenge command to get a random value from the Module that has the same length as the authenticated keys. The host application then generates a SHA2-256 hash value from the PIN provided by the operator. The random value is encrypted with the SHA2-256 hash value of the PIN using an AES-128 symmetric algorithm. The PIN ID and ciphertext are transmitted to the Module for verification. The Module generates a SHA2-256 hash value from the PIN based on the PIN ID, decrypts the ciphertext with the SHA2-256 hash value using the pre-defined symmetric algorithm, and compares the result with the original random value from the GetChallenge command. Once the default PIN is used, changing it is required upon first login. If the results are the same, the authentication has been completed successfully; otherwise, the authentication fails, and the module will decrement the "LeftRetryTimes" variable. Once the value stored in "LeftRetryTimes" reaches 0, the PIN is locked.

No visible display of the authentication data is allowed from the Module. The host application running on the GPC interacting with the module obscures the authentication data during data entry. The authentication data is stored in the key files which can never be exported outside the mToken CryptoID.

Method Name	Description	Security Mechanism	Strength Each Attempt	Strength per Minute
AM#1	Single-Factor Cryptographic Software as the cryptographic key (KEY_MC) is only given to a single user.	Challenge-response mechanism: The host application get a random value from the module. The host application then encrypts the random value with the pre-defined	The KEY_MC is a 256-bit AES key. The authentication mechanism has a 256-bit security strength which is 1/2^256.	The device locks after 15 consecutive failed authentication attempts. The probability of a successful random attempt during a one-minute period is approximately 15/2^256

Method Name	Description	Security Mechanism	Strength Each Attempt	Strength per Minute
		symmetric algorithm and transmits the ciphertext to the module for verification.		
AM#2	Memorized Secret as a specific Secret (KEY_PIN) is associated with a specific user.the host application calls the gets a random value from the module. The host application then generates a SHA2-256 hash value from the PIN provided by the operator. The random value is encrypted with the SHA2-256 hash value of the PIN using a pre-defined symmetric algorithm.	The host application calls the gets a random value from the module. The host application then generates a SHA2-256 hash value from the PIN provided by the operator. The random value is encrypted with the SHA2-256 hash value of the PIN using a pre-defined symmetric algorithm.	The KEY_PIN length must be between 8 and 32 characters with a combination of letters, numeric characters, and special characters (i.e., A-Z a-z 0-9 ~ ` ! @ # $ % ^ & * ( ) _ + | - = \ { } [ ] : ; " ' < > , . ? / Tab and Space) which is 1/96^8.	The module will lock an account after 15 consecutive failed authentication attempts. An Admin may unlock a User or unlocking may be performed by reinitializing (remove and reinsert). The probability of a successful random attempt during a one-minute period is approximately 15/96^8.

Table 12: Authentication Methods

4.2 Roles

The Module supports three (3) distinct operator roles, Issuer (CO), Admin and User. The cryptographic module enforces the separation of roles by associating the current connection with the most recent authenticated user identity.

The Module does not support a maintenance role or bypass capability. The Module does not support concurrent operators. The current operator status is stored in the security context in memory. When the operator logs out or if another operator attempts to login, the security context will be cleared, and the operator will be logged out automatically. The security context will also be cleared when the Module is powered off.

The Roles table and the Approved Services table lists all operator roles (Issuer (CO) – "I", Admin – "A" and User – "U") supported by the Module and their related services. In addition, the Module supports services which do not require to be authenticated, listed as Role "N" in The Roles table and the Approved Services table. Note, all services return an indication of success or specific error in addition to the output indicated. When in Non-Approved mode the services annotated with an * allow the Non-Approved algorithms and key sizes indicated in Table 7. Unless annotated with an ~, services are available using secure messaging using S_ENC and S_MAC.

The Roles Table below lists all operator roles supported by the Module.

The Module does not support concurrent operators.

Name	Type	Operator Type	Authentication Methods
Issuer (CO)	Identity	CO	AM#1
Admin	Identity	Other	AM#2
User	Identity	Other	AM#2

Table 13: Roles

4.3 Approved Services

All approved services implemented by the Module are listed in the table below:
Unless annotated with an ~, the service is available using secure messaging which uses S_ENC with CBC-AES256 and S_MAC with CMAC-AES256 with access rights of E as defined below.

The SSPs modes of access shown in the table below are defined as:

- 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 (SSP is input).
- E = Execute: The Module uses the SSP in performing a cryptographic operation.
- Z = Zeroize: The Module zeroizes the SSP.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
FormatDevice	Initialize and format the file system. Can also use to configure certain mode. Set authentication keys. Performing the zeroization service.	Approved Mode	Format configuration, or working mode configuration, authentication keys	Set new configurations to module	None	Issuer (CO) - DRBG_EI: Z - DRBG_Seed: Z - DRBG_State Keys: Z - ECDSA Private Key: Z - KEY_EA: Z - KEY_IA: Z - KEY_MC: Z - KEY_PIN:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Z - PWD Hash: Z - RSA Private Key: Z - S_ENC: Z - S_MAC: Z - S_Private: Z - Session Key (up to 4): Z - ECDSA Public: Z - RSA Public Key: Z - S_Host: Z - S_Public: Z - HMAC Key: Z
ChangeWorkingMode	Change working mode: Approved or Non-Approved	Approved Mode	Mode (Approved/Non-Approved)	Switch to certain mode	None	Issuer (CO) - DRBG_EI: Z - DRBG_Seed: Z - DRBG_State Keys: Z - ECDSA Private Key: Z - KEY_EA: Z - KEY_IA: Z - KEY_MC: Z - KEY_PIN: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- PWD Hash: Z - RSA Private Key: Z - S_ENC: Z - S_MAC: Z - S_Private: Z - Session Key (up to 4): Z - ECDSA Public: Z - RSA Public Key: Z - S_Host: Z - S_Public: Z - HMAC Key: Z
InstallPIN(secure messaging only)	Install a new KEY_PIN	Approved Mode	KEY_PIN for current ADF, key metadata	N/A	None	Issuer (CO) - KEY_PIN: W
DRBGReseed	Reseed DRBG with internal entropy.	Approved Mode	Internal Entropy	DRBG reseed	RBG Entropy	Issuer (CO) - DRBG_EI: G,E - DRBG_Seed: G,E - DRBG_State Keys: G,E Admin - DRBG_EI: G,E - DRBG_Seed: G,E - DRBG_St

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						ate Keys: G,E User - DRBG_EI: G,E - DRBG_Seed: G,E - DRBG_State Keys: E
GetFSMaxSpace~	Get max available space for file system	Approved Mode	None	Available space	None	Issuer (CO) Admin User
GetChipID~	Get the smart card chip serial number	Approved Mode	None	Chip ID	None	Issuer (CO) Admin User
GetHealthStatus~	Get self-test result	Approved Mode	None	Self-test result	None	Issuer (CO) Admin User
GetErrorLog~	Get self-test and last cmd error log	Approved Mode	None	Get self-test and last cmd error log	None	Issuer (CO) Admin User
HealthCheck~	On demand self-test	Approved Mode	None	Perform on demand self-test again	None	Issuer (CO) Admin User
SessionKeyKAS	Generate a key pair or Session Key using EC Diffie Hellman (P-256/P-521) KAS and KDF. KDF use alg: SHA256/384/512 Generate key Type: AES 128/192/256	Approved Mode	Algorithm IDs, KAS input keys, Key ID	AsymAlgID, PublicKey	KTS1 AsymGen1	Admin - S_Private: G,W,E,Z - S_Public: G,R,W,E,Z - Session Key (up to 4): G,W,Z - DRBG_State Keys: G,E - HMAC Key: G,E User - S_Private:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						G,W,E,Z - S_Public: G,R,W,E, Z - Session Key (up to 4): G,W,Z - DRBG_State Keys: G,E - HMAC Key: G,E
GenSessionKey	Generate a Session Key	Approved Mode	Key ID, Alg ID	Generate session key with DRBG engine	SymGen1 SymGen2 RBG	Admin - DRBG_EI: G,E - Session Key (up to 4): G,W,Z - DRBG_State Keys: G,E - HMAC Key: G,E User - DRBG_EI: G,E - Session Key (up to 4): G,W,Z - DRBG_State Keys: G,E - HMAC Key: G,E
DestroySessionKey	Destroy Session Key	Approved Mode	Key ID	Destroy the certain session key, or destroy all	None	Admin - Session Key (up to 4): Z User - Session Key (up to 4): Z
GetSessionInfo	Get the Session Key's algorithm	Approved Mode	Key ID	Alg ID	None	Admin User

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Create File	Create a DF or EF	Approved Mode	File metadata (access control, secure message req., etc.)	Create a DF or EF	None	Issuer (CO) - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private Key: W,Z Admin - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private Key: W,Z User - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private Key: W,Z
Delete File	Delete a DF or EF	Approved Mode	File identifier	Delete a DF or EF	None	Issuer (CO) - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Key: W,Z Admin - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private Key: W,Z User - RSA Public Key: W,Z - RSA Private Key: W,Z - ECDSA Public: W,Z - ECDSA Private Key: W,Z
ReadBinary	Read binary file data	Approved Mode	File identifier	Binary file data	None	Admin User
UpdateBinary	Update binary file data	Approved Mode	File identifier, binary data	Update data into binary file	None	Admin User
AppendRecord	Append new record to record file	Approved Mode	File identifier, record data	Record ID	None	Admin User
ReadRecord	Read record data of record file	Approved Mode	File identifier, record data	read data	None	Admin User
UpdateRecord	Update data into a record of the record file	Approved Mode	File identifier, record data	Update Record	None	Admin User
Get Data	Get device information (Module name, hardware, firmware versions, approved status) or get the specified	Approved Mode	Info tag/identifier	Device info	None	Admin User

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	tag data. Authentication is required for tag ID 0x4000-0XFFFF according to the "Read" access control of EF 3F01					
Put Data	Set device information or save the specified tag data	Approved Mode	Info tag, new device info	Set device info or save the specified tag data	None	Admin User
Select File	Select a DF or EF and get the information	Approved Mode	File identification	Select a DF or EF, and get detail info	None	Issuer (CO) Admin User
ChangeSecretKey(secure messaging only)	Change the specified KEY_MC, KEY_IA, KEY_PIN, KEY_EA	Approved Mode	Key identifier, new key value	Change the specified key value	None	Issuer (CO) - KEY_MC: W,Z - KEY_PIN: W,Z - KEY_EA: W,Z - KEY_IA: W,Z Admin - KEY_MC: W,Z - KEY_PIN: W,Z - KEY_EA: W,Z - KEY_IA: W,Z User - KEY_MC: W,Z - KEY_PIN: W,Z - KEY_EA: W,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- KEY_IA: W,Z
UnblockSecretKey(secure messaging only)	Unblock the specified Keys. Admin can unblock KEY_PIN, Issuer can unblock KEY_EA under MF	Approved Mode	Key ID, PIN or data	Unblock the specified Key_PIN or Key_EA	Hash	Issuer (CO) - KEY_PIN: W,Z - KEY_EA: W,Z Admin - KEY_PIN: W,Z - KEY_EA: W,Z
GetChallenge~	Get random challenge data from SP 800-90A DRBG	Approved Mode	length	Random challenge data	RBG	Issuer (CO) - DRBG_State Keys: R,G,E Admin - DRBG_State Keys: R,G,E User - DRBG_State Keys: R,G,E
ExternalAuth~	External authentication with KEY_EA	Approved Mode	acknowledge response	Authenticate with KEY_MC or KEY_EA	KTS1	Issuer (CO) - KEY_MC: E - KEY_EA: E Admin - KEY_MC: E - KEY_EA: E User - KEY_MC: E - KEY_EA: E
InternalAuth	Internal authenticatio	Approved Mode	authentication data	Calculated response	KTS1	Issuer (CO) - KEY_IA:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	n with KEY_IA					E Admin - KEY_IA: E User - KEY_IA: E
Verify Pin	Verify Admin or User PIN	Approved Mode	PIN ID, challenge response	Verified PIN	KTS1 Hash	Issuer (CO) - KEY_PIN: E Admin - KEY_PIN: E User - KEY_PIN: E
GetSecretKeyInfo	Get the specified information (i.e., Algorithm ID, PIN type, Left retry times, access control and Unblock PIN ID) of KEY_MC, KEY_IA, KEY_PIN, KEY_EA	Approved Mode	Key ID	Alg, type, retry, AC,PIN id Secret role, Default secret flag	None	Issuer (CO) Admin User
ClearSecureState	Logoff current DF or MF	Approved Mode	MF, ADF, both, Role/FIPS	Logoff current ADF or MF	None	Issuer (CO) Admin User
SymImportSessionKey	Import a Session Key that is wrapped by a Session Key	Approved Mode	Alg ID, unwrap key ID, key ID to import, wrapped key value	Import wrapped session key	KTS1	Admin - Session Key (up to 4): W,E User - Session Key (up to 4): W,E
SymExportSessionKey	Export a Session Key that is wrapped by a different Session Key	Approved Mode	wrap key ID, key ID to export	Alg ID, wrapped key	KTS1	Admin - Session Key (up to 4): R,E User - Session

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Key (up to 4): R,E
SymCryptInit~	Symmetric algorithm initialization	Approved Mode	Key ID, padding scheme, IV	First step of encryption or decryption	AES Encryption AES Decryption	Admin - Session Key (up to 4): E User - Session Key (up to 4): E
SymEncryptUpdate~	Continuously encrypt data part	Approved Mode	plaintext	ciphertext	AES Encryption	Admin - Session Key (up to 4): E User - Session Key (up to 4): E
SymEncryptFinal~	Encrypt the final data part and finish the operation	Approved Mode	plaintext	ciphertext	AES Encryption	Admin - Session Key (up to 4): E User - Session Key (up to 4): E
SymDecryptUpdate~	Continuously decrypt data part	Approved Mode	ciphertext	plaintext	AES Decryption	Admin - Session Key (up to 4): E User - Session Key (up to 4): E
SymDecryptFinal~	Decrypt the final data part and finish the operation	Approved Mode	ciphertext	plaintext	AES Decryption	Admin - Session Key (up to 4): E User - Session Key (up to 4): E
MacInit~	Initialize MAC Calculation	Approved Mode	Key ID	Initialize a new Mac operation	Message Authentication	Admin - Session Key (up to 4): E - HMAC Key: E User - Session Key (up to 4): E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- HMAC Key: E
MacUpdate~	MAC calculation update	Approved Mode	Data	Continue processing another data part	Message Authentication	Admin - Session Key (up to 4): E - HMAC Key: G,E User - Session Key (up to 4): E - HMAC Key: G,E
MacFinal~	MAC calculation finish	Approved Mode	Data	MAC	Message Authentication	Admin - Session Key (up to 4): E - HMAC Key: G,E User - Session Key (up to 4): E - HMAC Key: G,E
AsymGenKeypair	Generate asymmetric key pairs	Approved Mode	Pub key ID and Pri Key, Alg ID, use	RSA or ECDSA keypair	AsymGen1 AsymGen2 AsymVer1	Admin - RSA Public Key: G,W,Z - RSA Private Key: G,W,Z - ECDSA Public: G,W,Z - ECDSA Private Key: G,W,Z - DRBG_State Keys: G,E User - RSA Public Key: G,W,Z - RSA Private

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Key: G,W,Z - ECDSA Public: G,W,Z - ECDSA Private Key: G,W,Z
AsymWrapImportPub	Import a public key wrapped by a Session Key	Approved Mode	Unwrap key ID, Pub key ID, Alg ID, use, wrapped key	Import a wrapped public key	KTS2 SigVer1 SigVer2	Admin - RSA Public Key: W,Z - ECDSA Public: W,Z - Session Key (up to 4): E User - RSA Public Key: W,Z - ECDSA Public: W,Z - Session Key (up to 4): E
AsymImportPub	Import a Public key in plaintext	Approved Mode	Pub key ID, Alg ID, use, Pub key	Import a plaintext public key	None	Admin - RSA Public Key: W,Z - ECDSA Public: W,Z User - RSA Public Key: W,Z - ECDSA Public: W,Z - Session Key (up to 4): E
AsymWrapImportPri	Import a private key wrapped by a Session Key	Approved Mode	Unwrap key ID, Pri key ID, Pub key ID, Alg ID, use, wrapped key	Import a wrapped private key	KTS2 SigGen1 SigGen2	Admin - RSA Private Key: W,Z - ECDSA Private Key: W,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- Session Key (up to 4): E User - RSA Public Key: W,Z - ECDSA Public: W,Z - Session Key (up to 4): E
AsymWrapExportPub	Export a public key wrapped with a Session Key	Approved Mode	Wrap SessionID, Pub key ID	Alg ID, use, wrapped key	KTS2 SigVer1 SigVer2	Admin - RSA Public Key: R - ECDSA Public: R - Session Key (up to 4): E User - RSA Public Key: R - ECDSA Public: R - Session Key (up to 4): E
AsymExportPub	Export a Public key in plaintext	Approved Mode	Pub key ID	Alg ID, use, key	None	Admin - RSA Public Key: R - ECDSA Public: R User - RSA Public Key: R - ECDSA Public: R
AsymWrapExportPri	Export a Private key wrapped by a Session Key	Approved Mode	Wrap SessionID, Pri key ID	Alg ID, use, wrapped key	KTS2 SigGen1 SigGen2	Admin - RSA Private Key: R - ECDSA Private Key: R - Session Key (up to 4): E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						User - RSA Private Key: R - ECDSA Private Key: R - Session Key (up to 4): E
AsymSign	Calculate digital signature	Approved Mode	Pri key ID, Hash Alg ID, hash/data	Alg ID, signature	SigGen1 SigGen2	Admin - RSA Private Key: E - ECDSA Private Key: E User - RSA Private Key: E - ECDSA Private Key: E
AsymVerifySign	Verify digital signature	Approved Mode	Pub key ID, Hash Alg ID, signature	Alg ID, signature	SigVer1 SigVer2 SigVer3	Admin - RSA Private Key: E - ECDSA Private Key: E User - RSA Private Key: E - ECDSA Private Key: E
HashInit~	Initialize a hash operation	Approved Mode	Alg ID	Initialize a hash operation	Hash	Issuer (CO) Admin User
HashUpdate~	Continuously hash data part	Approved Mode	Data	Continue processing another data part	Hash	Issuer (CO) Admin User
HashFinale~	Hash the final data part and get the hash value	Approved Mode	Data	Digest result	Hash	Issuer (CO) Admin User
SecureMessageKAS~	EC Diffie-Hellman (P-521) KAS	Approved Mode	Alg ID, kdf hash ID,	Alg ID, pub key	KAS1	Issuer (CO) -

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	(including KDF) to generate the Secure Message Keys (S_ENC and S_MAC) to establish a secure channel between the module and the host application		otherInfo, pub key			S_Private: G,W,E,Z - S_Public: G,R,W,E, Z - S_Host: E - S_ENC: G,W,Z - S_MAC: G,W,Z Admin - S_Private: G,W,E,Z - S_Public: G,R,W,E, Z - S_Host: E - S_ENC: G,W,Z - S_MAC: G,W,Z User - S_Private: G,W,E,Z - S_Public: G,R,W,E, Z - S_Host: E - S_ENC: G,W,Z - S_MAC: G,W,Z

Table 14: Approved Services

Note: “use” is a parameter to specify the usage of the private-public key pairs. Option 1 means signature generation and verification only, Option 2 means key wrapping only, and Option 3 means the key pair can be used for both signature generation and verification, and key wrapping which is not allowed in Approved mode.

Note: For Getdata, authentication is required for tag ID 0x4000- 0XFFFF according to the “Read” access control of EF 3F01.

The key attribute can identify the type of key, ID, number of retries, whether it is unmodified, etc. These attributes are not CSP or PSP.

You cannot obtain the specific content of the key through the properties of the key, nor can you use (signature, verify, encrypt and decrypt) or clear the key through the properties of the key.

For example：
A user PIN ID is 0x01
Retry time is 10
PIN value is 12345678
Through GetSecretKeyInfo you can get the ID and Retry, but can't get the PIN value.

For details regarding service inputs, corresponding service outputs, status return codes and description of each service listed, please refer to section 4 of [APDU].

4.4 Non-Approved Services

All Non-Approved services implemented by the Module are listed in the table below:

Name	Description	Algorithms	Role
SessionKeyKAS	Generate a key pair or Session Key using EC Diffie Hellman (P-256/P-521) KAS and KDF	SHA-1	Admin, User
GenSessionKey	Generate a Session Key	HMAC-SHA-1 Triple-DES CMAC with Triple-DES	Admin, User
Verify PIN	Verify Admin or User PIN.	Triple-DES	Issuer, Admin, User, Unauthenticated
SymImportSessionKey	Import a Session Key that is wrapped by a Session Key	Triple-DES	Admin, User
SymExportSessionKey	Import a Session Key that is wrapped by a Session Key	Triple-DES	Admin, User
SymCryptInit	Symmetric algorithm initialization	Triple-DES	Admin, User
SymEncryptUpdate	Continuously encrypt data part	Triple-DES	Admin, User
SymEncryptFinal	Encrypt the final data part and finish the operation	Triple-DES	Admin, User
SymDecryptUpdate	Continuously decrypt data part	Triple-DES	Admin, User
SymDecryptFinal	Decrypt the final data part and finish the operation	Triple-DES	Admin, User
MacInit	Initialize MAC Calculation	CMAC with Triple-DES	Admin, User
MacUpdate	MAC calculation update	CMAC with Triple-DES	Admin, User
MacFinal	MAC calculation finish	CMAC with Triple-DES	Admin, User
AsymWrapImportPub	Import a public key wrapped by a Session Key	Triple-DES	Admin, User

Name	Description	Algorithms	Role
AsymWrapImportPri	Import a private key wrapped by a Session Key	Triple-DES	Admin, User
AsymWrapExportPub	Export a public key wrapped with a Session Key	Triple-DES	Admin, User
AsymWrapExportPri	Export a Private key wrapped by a Session Key	Triple-DES	Admin, User
AsymSign	Calculate digital signature	SHA-1	Admin, User
AsymVerifySign	Verify digital signature	SHA-1	Admin, User
HashInit	Initialize a hash operation	SHA-1	Issuer, Admin, User, Unauthenticated
HashUpdate	Continuously hash data part	SHA-1	Issuer, Admin, User, Unauthenticated
HashFinale	Hash the final data part and get the hash value	SHA-1	Issuer, Admin, User, Unauthenticated
CreateSessionKey	Create a Session Key with plaintext key value	SHA-1	Admin, User
AsymEnDecrypt	RSA PKCS V1.5 encrypt/decrypt	RSA	Admin, User
AsymKeyOperation	RSA encrypt/decrypt (no padding)	RSA	Admin, User
AsymImportSessionKey	Import a Session Key that is wrapped by a Public Key	KTS Triple-DES	Admin, User
AsymExportSessionKey	Export an AES Session Key that is wrapped by a Public Key	KTS Triple-DES	Admin, User

Table 15: Non-Approved Services

4.5 External Software/Firmware Loaded

NOTE: There is no External Software/Firmware Loaded.

5 Software/Firmware Security

5.1 Integrity Techniques

The Module is composed of a single firmware component. The mToken CryptoID uses ECDSA P-256(SHA2-256) for the integrity test of its firmware. ECDSA digital SigVer is an Approved algorithm that is provided by the module. The known digital signature value of the firmware is stored in the firmware binary. When the Module is powered-up, the Module will generate the SHA2-256 hash value of the firmware of the Module. Then use ECDSA P-256 to digitally verify the generated SHA2-256 hash value along with the stored digital signature. If this fails, the integrity test fails, and the Module enters the Error state.

5.2 Initiate on Demand

The operator can initiate the integrity test on demand via the HealthCheck service.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment:

Non-Modifiable

How Requirements are Satisfied [O]:

6.2 Configuration Settings and Restrictions

The Module operates in a non-modifiable operational environment and does not implement a General Purpose Operating System. Once the firmware of the Module is loaded on the mToken CryptoID, it cannot be modified or erased, firmware cannot be upgraded/updated. The operational environment requirements do not apply to the Module.

7 Physical Security

7.1 Mechanisms and Actions Required

The Module's enclosure, which surrounds the SCC-XE microcontroller, is made of fully hardened production grade polycarbonate (plastic) or metal. A colored polycarbonate or metal enclosure blocks the clear view of internal hardware components. There is a hard, non-malleable metal casing around the USB connector.

The SCC-XE microcontroller is covered in a black, opaque, tamper resistant epoxy resin coating thus completely covering all critical components from visual inspection. Any attempt to remove or penetrate the enclosure is highly likely to cause serious damage to the Module and hardware components inside the enclosure, which will expose clear evidence of tampering. Removing the metal around the USB connector will cause physical damage to the USB connector and its related pins, making the entire cryptographic module inoperable. If evidence of tampering occurs, the Module should be returned to the issuer immediately to be destroyed.

Once the cryptographic Module is powered off, all plaintext keys and unprotected CSPs will be zeroized.

Mechanism	Inspection Frequency	Inspection Guidance
Hard enclosure	During Initialization by CO and Before first use by end user.	Operator should look for damage

Table 16: Mechanisms and Actions Required

7.5 EFP/EFT Information

Temp/Voltage Type	Temperature or Voltage	EFP or EFT	Result
LowTemperature	-30C	EFP	Zeroization/Shutdown
HighTemperature	80C	EFP	Zeroization/Shutdown
LowVoltage	2.7v	EFP	Shutdown
HighVoltage	5.8v	EFP	Shutdown

Table 17: EFP/EFT Information

7.6 Hardness Testing Temperature Ranges

Temperature Type	Temperature
LowTemperature	-30C
HighTemperature	80C

Table 18: Hardness Testing Temperatures

The Module was only tested at nominal room temperature.

8 Non-Invasive Security

The Module does not implement any mitigation method against non-invasive attacks.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
System Memory	Only stored in volatile memory (RAM) in plaintext.	Dynamic
Disk Drive	Stored in flash in plaintext, associated by memory location (pointer).	Static

Table 19: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
Input with Key wrapping using SP800-38f	the host	Disk Drive	Encrypted	Manual	Electronic	KTS1
Input with Key wrapping using IG D.G	the host	Disk Drive	Encrypted	Manual	Electronic	KTS2
Input in plain text	the host	System Memory	Plaintext	Manual	Electronic	SigVer3
Output with Key wrapping using SP800-38f	Disk Drive	the host	Encrypted	Manual	Electronic	KTS1
Output with Key wrapping using IG D.G	Disk Drive	the host	Encrypted	Manual	Electronic	KTS2
Output in plain text	System Memory	the host	Plaintext	Manual	Electronic	SigVer3

Table 20: SSP Input-Output Methods

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Z1	By rebooting the module.	Reboot module, all data stored in volatile memory (RAM) will be lost	Implicit
Z2	Derive New	Zeroize the old SSP, and then derive a new SSP	Explicit
Z3	"FormatDevice" APDU.	Format the file system of the device with the specified configuration. Overwrite with 1's followed by reset to default value if appropriate.	Explicit

Zeroization Method	Description	Rationale	Operator Initiation
Z4	Erased by "DeleteFile" APDU	Deletes the specified DF or EF with 0xff.	Explicit
Z5	Erased by "DestroySessionKey" APDU	Destroy the specified session key or all session keys with 0xff	Explicit
Z6	Erased by "ClearSecureState" APDU	Clear Auth status	Implicit
Z7	SecureMessageKAS	Zeroization of S_ENC, S_MAC, S_Private and S_Pulic keys and generates new of these.	Implicit
Z8	Automatically destroyed after authentication.	Set 0xff to the temporary data	Implicit
Z9	Erased by "ChangeWorkingMode" APDU	Format the file system of the device with the specified configuration. Overwrite with 1's followed by reset to default value if appropriate.	Explicit

Table 21: SSP Zeroization Methods

9.4 SSPs

All usage of these SSPs by the Module are described in the services detailed in Section 4.3

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
DRBG_EI	EntropHash DRBG entropy input. ESV Cert. #E44y input, Nonce from the entropy source DRBG (Cert. #A2662)	1150 - N/A	N/A - CSP	Entropy		RBG
DRBG_Seed	Entropy input, Nonce from the entropy source DRBG (Cert. #A2662)	1724 - N/A	N/A - CSP	Entropy		RBG
DRBG_State Keys	Derived from inputs into the Hash_DRBG (V and C) DRBG (Cert. #A2662)	440 - 256	Symmetric Key - CSP	RBG		SymGen1 AsymGen1 SigGen1 AsymGen2 SymGen2

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
ECDSA Private Key	ECDSA Signature Generation key	P256, P512 - 128, 256	Asymmetric Private Key - CSP	AsymGen1		ECDSA KeyGen (FIPS186-4) (A2663) ECDSA KeyVer (FIPS186-4) (A2663)
KEY_EA	Used to authenticate the host to the module	8 and 32 character password - N/A	Password - CSP			AES-KW (A2660)
KEY_IA	Used to authenticate the module to the host	8 and 32 character password - N/A	Password - CSP			AES-KW (A2660)
KEY_MC	Used to authenticate the Issuer	256 - 256	Symmetric Key - CSP			AES-KW (A2660)
KEY_PIN	Used to authenticate the Admin or User	8 and 32 character password - N/A	Password - CSP			
PWD Hash	256-bit password hash for KEY_PIN	256 - 256	Hash Key - CSP	Hash		Hash
RSA Private Key	RSA PKCS1 Signature Generation Key	2048 - 112	Asymmetric Private Key - CSP	AsymGen2		RSA KeyGen (FIPS186-4) (A2663)
S_ENC	Secure messaging encryption key	256 - 256	Symmetric Key - CSP		KAS1	AES-CBC (A2660) AES-CMAC (A2660) AES-ECB (A2660)
S_MAC	Secure messaging MAC	256 - 256	MAC Key - CSP		KAS1	AES-CBC (A2660) AES-CMAC (A2660) AES-ECB (A2660)
S_Private	EC DH Private key agreement key	P-521 - 256	Asymmetric Private Key - CSP	AsymGen1		KAS1 ECDSA KeyGen (FIPS186-4) (A2663) ECDSA KeyVer (FIPS186-4) (A2663)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Session Key (up to 4)	1. Data encryption /decryption 2. Wrapping other keys 3. HMAC/CMAC calculation	128, 192, 256 - 128, 192, 256	Symmetric Key - CSP	SymGen1 SymGen2		KTS2
HMAC Key	Keys used for HMAC-SHA2-256, HMAC-SHA2-384, and HMAC-SHA2-512	HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512 - 256, 384, 512	MAC key - CSP	Message Authentication		Message Authentication
ECDSA Public	ECDSA Verification and KAS	P256, P521 - 128, 256	Asymmetric Public Key - PSP	AsymGen1		KAS-ECC Sp800-56Ar3 (A2659) ECDSA SigVer (FIPS186-4) (A2663)
RSA Public Key	RSA PKCS1 Verification Key	2048 - 112	Asymmetric Public Key - PSP	AsymGen2		RSA SigVer (FIPS186-4) (A2663)
S_Host	Secure messaging ephemeral host public key	P-256, P-512 - 128, 256	Asymmetric Public Key - PSP			KAS-ECC Sp800-56Ar3 (A2659) ECDSA KeyVer (FIPS186-4) (A2663)
S_Public	Secure messaging ephemeral module public key	P-256, P-512 - 128, 256	Asymmetric Public Key - PSP	AsymGen1		KAS-ECC Sp800-56Ar3 (A2659) ECDSA SigVer (FIPS186-4) (A2663)

Table 22: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
DRBG_EI		System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z9	DRBG_Seed:Used with Hash_DRBG

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
DRBG_Seed		System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z9	DRBG_EI:Derived from entropy source output and nonce DRBG_State Keys:Used by the Hash_DRBG
DRBG_State Keys		System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z9	DRBG_Seed:Derived from entropy source output and nonce ECDSA Private:Derived from the DRBG into the HASH_DRBG RSA Private Key:Derived from the DRBG into the HASH_DRBG ECDSA Public:Derived from the DRBG into the HASH_DRBG RSA Public Key:Derived from the DRBG into the HASH_DRBG S_Private:Derived from the DRBG into the HASH_DRBG S_Public:Derived from the DRBG into the HASH_DRBG
ECDSA Private Key	Input with Key wrapping using SP800-38f Output with Key wrapping using SP800-38f	Disk Drive :Plaintext	N/A	Z1 Z3 Z4 Z9	DRBG_State Keys:Derived From ECDSA Public:Paired With
KEY_EA	Input with Key wrapping using IG D.G	Disk Drive :Plaintext	N/A	Z1 Z3 Z9	
KEY_IA	Input with Key wrapping using IG D.G	Disk Drive :Plaintext	N/A	Z3 Z9	
KEY_MC	Input with Key wrapping using IG D.G	Disk Drive :Plaintext	N/A	Z3 Z9	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
KEY_PIN	Input with Key wrapping using IG D.G	Disk Drive :Encrypted	N/A	Z3 Z4 Z9	PWD Hash:As hash value of KEY_PIN
PWD Hash		System Memory :Plaintext	Until it is used	Z3 Z8 Z9	KEY_PIN:As the original value
RSA Private Key	Input with Key wrapping using SP800-38f Output with Key wrapping using SP800-38f	Disk Drive :Plaintext	N/A	Z3 Z4 Z9	DRBG_State Keys:Used With RSA Public Key:Paired With
S_ENC		System Memory :Plaintext	Until the module is rebooted	Z1 Z2 Z3 Z6 Z7 Z9	S_Private:Join the derive process with S_MAC S_MAC:Used With
S_MAC		System Memory :Plaintext	Until the module is rebooted	Z1 Z2 Z3 Z6 Z7 Z9	S_Private:Join the derive process with S_ENC S_ENC:Used With
S_Private		System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z7 Z9	S_ENC:Join the derive process with S_MAC S_MAC:Join the derive process with S_ENC S_Public:Paired With
Session Key (up to 4)	Input with Key wrapping using SP800-38f Output with Key wrapping using SP800-38f	System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z5 Z6 Z7 Z9	DRBG_State Keys:Use random value as session key
HMAC Key		System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z5 Z6 Z7 Z9	Session Key (up to 4):Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
ECDSA Public	Output with Key wrapping using IG D.G	Disk Drive :Plaintext	N/A	Z3 Z4 Z9	ECDSA Private Key:Paired With
RSA Public Key	Output with Key wrapping using IG D.G	Disk Drive :Plaintext	N/A	Z3 Z4 Z9	RSA Private Key:Paired With
S_Host	Input in plain text	System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z6 Z9	S_Private:: Join KAS with S_Private S_ENC:Derive from KAS, as a shared secret S_MAC:Derive from KAS, as a shared secret
S_Public	Output in plain text	System Memory :Plaintext	Until the module is rebooted	Z1 Z3 Z6 Z9	S_Private:Paired With

Table 23: SSP Table 2

10 Self-Tests

10.1 Pre-Operational Self-Tests

The Module performs self-tests to ensure the proper operation of the Module. Per FIPS 140-3 requirements, these are categorized as either pre-operational self-tests or conditional self-tests.

When the mToken CryptoID is powered on, the Pre-Operational self-tests are executed automatically without any operator intervention. If the Module completes the Pre-Operational self-tests successfully, the Module will enter either Approved Mode or Non-Approved Mode depending on the initial configuration of the Module by the Manufacturer or Distributor. If the Module enters approved mode, a steady green LED will be observed; if the module enters Non-Approved mode, a steady red LED will be observed. The Module returns "0000000000000000" from the calling GetHealthStatus APDU command which indicates that the Pre-Operational self-tests have completed successfully.

If any Pre-Operational self-test fails, the Module enters into the Error state. All data output is prohibited, and no further cryptographic operation is allowed. The green LED will blink quickly (flashing every 200ms), and the module will return an error code to indicate that the Module is in the Error state.

The on-demand self-tests can be invoked by the HealthCheck APDU command or by unplugging the token and plugging it back in to reinitiate the Pre-Operational self-tests.

The Module logs the last self-test error which can be retrieved via the APDU command GetHealthStatus.

1. The Error log is stored on RAM, which records two types of errors: self-test error and error returned from module function interface. For detail error definition, please refer to the table description of STATUS CODE+ algorithm self-test status of "mToken CryptoID APDU Specification V1.4".
2. Obtain the error log via 80 05 command. No interface exists to modify/delete error log. While executing the read command, requires check the module's authority, if is not login status of authorized user (CO/USR), then return error 0x6982. Authorized user may obtain certain log records after verifying PIN. By default, this will return "0", if there are no errors; otherwise, the latest error will be returned.

The Module performs the following pre-operational self-tests in table below:

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
ECDSA SigVer (FIPS186-4) (A2663)	ECDSA P-256	Signature Verification	SW/FW Integrity	steady green LED will be observed and the GetHealthStatus APDU command returns code: 0000000000000000	Executed on the whole firmware before the Module transition to the idle state.
ESV	RCT and APT	SP 800-90B Health-Test	Critical Function	Success or Failure Code	As specified in [90B] section 4.4. for start-up requirements

Table 24: Pre-Operational Self-Tests

10.2 Conditional Self-Tests

The Module performs the following conditional self-tests in the table below:

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A2660)	AES-128 ECB	KAT	CAST	steady green LED will be observed and the GetHealthStatus APDU command returns code: 0000000000000000	Encryption	bootup
AES-CMAC (A2660)	256	KAT	CAST	Success or Failure Code	Authentication Encryption	bootup
AES-ECB (A2660)	AES-128 ECB	KAT	CAST	Success or Failure Code	Decryption	bootup
AES-CMAC (A2660)	256	KAT	CAST	Success or Failure Code	Authenticated Decryption	bootup
Hash DRBG (A2662)	SHA2-256	KAT	CAST	Success or Failure Code	Hash_DRBG health tests per SP 800-90A Section 11.3 (Generate, Instantiate, and reseed)	bootup
ECDSA SigGen (FIPS186-4) (A2663)	P-521	KAT	CAST	Success or Failure Code	Signature Generation	bootup
ECDSA SigVer (FIPS186-4) (A2663)	P-521	KAT	CAST	Success or Failure Code	Signature Verification	bootup
ECDSA KeyGen (FIPS186-4) (A2663)	P-521	PCT inclusive to KeyGen and KeyVer	PCT	Success or Failure Code	Key Generation Pairwise Consistency Test for Signature Generation and ECDH key generation. Inclusive to KeyGen and KeyVer	bootup
ESV	RCT and APT	SP 800-90B Health-Test	CAST	Success or Failure Code	As specified in [90B] section 4.4 for continuous tests	Continuous

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC-SHA2-256 (A2661)	HMAC-SHA2-256	KAT	CAST	Success or Failure Code	HMAC-SHA2-256 KAT	bootup
HMAC-SHA2-384 (A2661)	HMAC-SHA2-384	KAT	CAST	Success or Failure Code	HMAC-SHA2-384 KAT	bootup
HMAC-SHA2-512 (A2661)	HMAC-SHA2-512	KAT	CAST	Success or Failure Code	HMAC-SHA2-512 KAT	bootup
RSA SigGen (FIPS186-4) (A2663)	2048-bit RSA PKCSv1.5 with SHA2-224	KAT	CAST	Success or Failure Code	Signature Generation	bootup
RSA SigVer (FIPS186-4) (A2663)	2048-bit RSA PKCSv1.5 with SHA2-224	KAT	CAST	Success or Failure Code	Signature Verification	bootup
RSA KeyGen (FIPS186-4) (A2663)	2048-bit	PCT	PCT	Success or Failure Code	RSA Pairwise Consistency Test	When a new RSA key is generated
SHA2-256 (A2661)	SHA2-256	KAT	CAST	Success or Failure Code	SHA2-256 KAT	bootup
SHA2-384 (A2661)	SHA2-384	KAT	CAST	Success or Failure Code	SHA2-384 KAT	bootup
SHA2-512 (A2661)	SHA2-512	KAT	CAST	Success or Failure Code	SHA2-512 KAT	bootup
KAS-ECC Sp800-56Ar3 (A2659)	Primitive Z and KDF	KAT	CAST	Success or Failure Code	Per IG D.F, separately tested Primitive Z and KDF	bootup

Table 25: Conditional Self-Tests

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA SigVer (FIPS186-4) (A2663)	Signature Verification	SW/FW Integrity	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				until the APDU command processing is completed.
ESV	SP 800-90B Health-Test	Critical Function	N/A	N/A

Table 26: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A2660)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed.
AES-CMAC (A2660)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed.
AES-ECB (A2660)	KAT	CAST	Automatically performed by the Module every 2 minutes	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-CMAC (A2660)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
Hash DRBG (A2662)	KAT	CAST	Automatically performed before Reseed. When the bit number less than 2^48	Automatically performed: programmatically
ECDSA SigGen (FIPS186-4) (A2663)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
ECDSA SigVer (FIPS186-4) (A2663)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
ECDSA KeyGen (FIPS186-4) (A2663)	PCT inclusive to KeyGen and KeyVer	PCT	N/A	N/A
ESV	SP 800-90B Health-Test	CAST	Continuous	As specified in [90B] section 4.4 for continuous tests
HMAC-SHA2-256 (A2661)	KAT	CAST	Automatically performed by the	Automatically performed:

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
			Module every 2 minutes.	programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
HMAC-SHA2-384 (A2661)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
HMAC-SHA2-512 (A2661)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
RSA SigGen (FIPS186-4) (A2663)	KAT	CAST	Automatically performed by the Module every 2 minutes.	utomatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
RSA SigVer (FIPS186-4) (A2663)	KAT	CAST	Automatically performed by the	Automatically performed: programmatically;

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
			Module every 2 minutes.	If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
RSA KeyGen (FIPS186-4) (A2663)	PCT	PCT	N/A	N/A
SHA2-256 (A2661)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
SHA2-384 (A2661)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed
SHA2-512 (A2661)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KAS-ECC Sp800-56Ar3 (A2659)	KAT	CAST	Automatically performed by the Module every 2 minutes.	Automatically performed: programmatically; If the module is in the process of executing an APDU command it will delay the periodic self-test until the APDU command processing is completed

Table 27: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
C_HEALTH_ERROR	The Module fails a self-test.	Noise Generator Self-Test Error Symmetric Algorithm Self-Test Error Pairwise consistency test Error	Reboot module	The Module enters the error state and outputs status indication: The green LED blinks every 200ms

Table 28: Error States

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

Installation and Initialization:

Physical access to the Module shall be limited to the Crypto-Officer, and the CO shall be responsible for putting the module into the Approved mode upon initialization.
First time access to the mToken CryptoID requires the CO to connect to the module and perform the initialization steps via the Factory Tool.
The following steps must be performed by the CO to securely install, initialize and start up the cryptographic module in the FIPS 140-3 Approved mode of operation:

1. The Issuer should open the Factory Tool, click **Find** button then **Connect** if the connected module is listed on the drop-down list next to the Find button. Select **FIPS Mode** checkbox and then click **FormatDevice** to initialize the Module. The process of FormatDevice is to clear all CSPs and set the module to Approved Mode. At this point, the Module's LED will flash Green which indicates the operation has been done successfully.
2. The Issuer needs to re-plug the Module to continue the setting. Repeat the **Find** then **Connect** step. To ensure module's security, the default keys must be changed upon the first use, thus no further steps are allowed until the default keys are changed. The Issuer should be responsible for keeping the new keys secure.
3. The Issuer needs to input correct KEY_MC, then click **ExternalAuth** to perform the Key Auth of the CO role., Later click **SecureMessageKAS** to establish an encrypted connection to initialize the device.
4. The Issuer needs click **InstallPin** to install KEY_PIN of Admin and User. Admin PIN is "admin123" by default; User PIN is "12345678" by default.
5. The Issuer may click **ChangeWorkingMode** to switch the module to Approved Mode or Non-Approved Mode accordingly, but the prerequisite is that the CO role is authenticated.

Startup Procedures

The security rules for Module usage are listed below for the Crypto Officer and User roles:

- SecureMessageKAS service is required to be used to establish a secure channel between the Module and the host application running on a GPC before requesting any service.
- The operator should check the LED status indicator to determine the state of the Module.
- The operators should logout when they finish using the Module to maintain secure usage of the module. The operators can logout by calling the ClearSecureState APDU command or by unplugging the module from the GPC.

The security rules for Module usage are listed below for the Crypto Officer role:

- When installing the KEY_MC and KEY_PIN, the "LeftRetryTimes" field is required to be set to fifteen (15) so that the Module will block the KEY_MC and KEY_PIN after fifteen (15) failed login attempts.
- The default value of KEY_MC is defined and exchanged between the Manufacturer and the Distributor as part of the contract. It is the Issuer's responsibility to change the default key immediately once he/she receives the module. The Issuer can use the ChangeSecretKey APDU command to change the KEY_MC.
- The Issuer should call the GetData APDU command to ensure that the module is configured to operate in approved mode. If the module is not configured to operate in approved mode, the Issuer should refer to Section 2.5 "Modes of Operation" for instructions on how to configure the module to operate in approved mode.
- The Manufacturer provides a secure delivery and distribution process to maintain the integrity of the module. The Distributor should inspect the exterior of the delivered package and the tamper tab on each box that contains the modules and confirm the basic information matching with the received modules. Please refer to document "mToken CryptoID Configuration Management Plan" for more information.

The security rules for module usage are listed below for the User role:

- The default values of KEY_PIN (i.e., Admin PIN and User PIN) are required to be securely delivered to the end user by the Distributor to maintain the integrity of the module. It is the end

user's responsibility to change the default KEY_PIN immediately once they receive the module. The end user can use the ChangeSecretKey APDU command to change the KEY_PIN.

Delivery:

The Module is shipped from the manufacturer without initialization to the Issuer (CO).

The following steps must be performed to securely deliver the *mToken CryptoID* cryptographic Module to the authorized operator:

1. The CO shall receive the Module from Longmai via trusted couriers (e.g., United Parcel Service, Federal Express, etc.).

2. On receipt, the CO must inspect the delivered package for any tampering and if any signs of tampering are found, the CO should contact Longmai.

11.2 Administrator Guidance

This Module needs to negotiate a pair of AES Keys (S_ENC and S_ENC) to allow authentication and secure initialization of the Module. All communications to initialize the Module will require a secure session using this key pair which will encrypt and authenticate all data input. During module initialization, the operator of the Module will only operate it in approved mode and provide only approved and allowed security functions.

To confirm operation in the Approved Mode of Operation, the operator can use the "Get Data" APDU command.

All communication of the module will require a secure session using the S_ENC and S_ENC for encrypting and MACing all data input and output.

Operators shall maintain physical possession of the device until keys are zeroized successfully. In this way, the zeroization technique is performed in a time that will not allow the CSPs to be compromised.

11.3 Non-Administrator Guidance

The Admin and User role can't initialize the device, install KEY_MC, KEY_IA, KEY_EA or KEY_PIN, and can't change the work mode either.

After the Admin and User role performs KEY_PIN authentication, it can create and delete CSP files such as public and private keys.

11.4 Design and Rules [O]

Rules of Operation

1. The Module provides three (3) distinct operator roles: Issuer (CO), Admin, and User.
2. The Module provides identity-based authentication.
3. The Module clears previous authentications on power cycle.

4. An operator does not have access to any cryptographic services prior to assuming an authorized role.
5. The Module allows the operator to initiate pre-operational and conditional self-tests via command as well as restarting the Module.
6. Pre-Operational self-tests do not require any operator action.
7. Data output is inhibited during key generation, self-tests, zeroization, and error states.
8. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the Module.
9. There are no restrictions on which keys or SSPs are zeroized by the zeroization service.
10. The Module does not support concurrent operators.
11. The Module does not support a maintenance interface or role.
12. The Module does not support a manual SSP establishment method.
13. The Module does not have any proprietary external input/output devices used for entry/output of data.
14. The Module does not enter or output plaintext CSPs.
15. The Module does not output intermediate key values.
16. The Module does not provide bypass services for ports/interfaces.

11.5 Maintenance Requirements

N/A

11.6 End of Life

After the end-of-life, the operator should send the device back to the Issuer (CO). The CO should zeroize all SSPs using the “FormatDevice“ service followed by shredding the Module.

12 Mitigation of Other Attacks

The Module implement the following mitigation methods against other attacks.

References and Definitions

The following standards may be referred to in this Security Policy.

Table 29 - References

Abbreviation	Full Specification Name
[FIPS140-3]	*Security Requirements for Cryptographic Modules*, March 22, 2019
[APDU]	*mToken CryptoID APDU Specification, Century Longmai Technology Co. Ltd, December 29, 2021*
[ISO19790]	*International Standard, ISO/IEC 19790, Information technology — Security techniques — Test requirements for cryptographic modules, Third edition, March 2017*
[ISO24759]	*International Standard, ISO/IEC 24759, Information technology — Security techniques — Test requirements for cryptographic modules, Second and Corrected version, 15 December 2015*
[IG]	*Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program, November 22, 2023*
[108]	*NIST Special Publication 800-108r1, Recommendation for Key Derivation Using Pseudorandom Functions (Revised), August 2022*
[131A]	*Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, Revision 2, March 2019*
[132]	*NIST Special Publication 800-132, Recommendation for Password-Based Key Derivation, Part 1: Storage Applications, December 2010*
[133]	*NIST Special Publication 800-133, Recommendation for Cryptographic Key Generation, Revision 2, June 2020*
[135]	*National Institute of Standards and Technology, Recommendation for Existing Application-Specific Key Derivation Functions, Special Publication 800-135rev1, December 2011.*
[186]	*National Institute of Standards and Technology, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-4, July 2013.*
[197]	*National Institute of Standards and Technology, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, November 26, 2001*
[198]	*National Institute of Standards and Technology, The Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards Publication 198-1, July, 2008*
[180]	*National Institute of Standards and Technology, Secure Hash Standard, Federal Information Processing Standards Publication 180-4, August, 2015*
[202]	*FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, FIPS PUB 202, August 2015*
[38A]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation, Methods and Techniques, Special Publication 800-38A, December 2001*

Abbreviation	Full Specification Name
[38B]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, Special Publication 800-38B, May 2005*
[38C]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality, Special Publication 800-38C, May 2004*
[38D]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, Special Publication 800-38D, November 2007*
[38E]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices, Special Publication 800-38E, January 2010*
[38F]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, Special Publication 800-38F, December 2012*
[56Ar3]	*NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, April 2018*
[56Br2]	*NIST Special Publication 800-56B Revision 2, Recommendation for Pair-Wise Key Establishment Schemes Using Finite Field Cryptography, March 2019*
[56Cr2]	*NIST Special Publication 800-56C Revision 2, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, August 2020*
[67]	*National Institute of Standards and Technology, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Special Publication 800-67, May 2004*
[90A]	*National Institute of Standards and Technology, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, Special Publication 800-90A, Revision 1, June 2015.*
[90B]	*National Institute of Standards and Technology, Recommendation for the Entropy Sources Used for Random Bit Generation, Special Publication 800-90B, January 2018.*

Table 30 – Acronyms and Definitions

Acronym	Definition
APDU	Application Protocol Data Unit
CCID	Circuit Cards Interface Device
MF	Master File. The root directory, contains all other directories and file
DF	Dedicated File. A directory contains other files and sub DF
EF	Elementary File. A file contains data, can be Binary File, Linear Variable Record File, Public Key File or Private Key File
IC	Integrated Circuit

Acronym	Definition
LED	Light Emitting Diode
PIN	Personal Identification Number
PKI	Public Key Infrastructure