quadient

Virtual PSD Module

Non-Proprietary FIPS 140-2 Security Policy

Author: ATOMOS INC – Roman Kresina

Valid from: 4/28/22
Version No.: V2.18
Approved by: Sidhartha Sridharan

1 Table of Contents

2 List of Tables

3 List of Figures

4 Introduction

This document defines the Security Policy for the Virtual PSD Module, hereafter denoted the Module. The Module is comprised of application software running within the IBM 4767 Cryptographic Coprocessor hardware security module, hereafter denoted the IBM 4767. The module embodiment is a multi-chip embedded module. The application software utilizes FIPS approved cryptographic libraries as provided by IBM Corporation. The Quadient, Inc. (Quadient) application software does not utilize any custom cryptographic algorithms.

Table 1 – Cryptographic Module Configurations

	Module	FW Version
1	IBM 4767	Segment 1 Information Name: 5.3.19 P0130 M0130 P0130 F0D01 Partial Hash = E2157B6F
2	IBM 4767	Segment 2 Information Name: 5.3.23 Toolkit Development Partial Hash = F0B7 1A25 3731
3	Virtual PSD Module	Segment 3 Information Name: NeopostUDX Version: 2.0.3-21-06-02-Release-ee523cf0bd19dd6378c7f9fbe931cbd5b83316e7

The tested hardware configuration is:

Hardware model: IBM 4767-002

P/N: 00LV498-N37142

Hardware version: POST0 v0123 MB0 v0121

The non-volatile memory on a coprocessor is partitioned into four segments, each of which can contain program code and sensitive data:

- Segment 0 contains one portion of Miniboot, the most privileged software in the coprocessor.
- Miniboot implements, among other things, the protocols that ensure nothing is loaded into the coprocessor without the proper authorization. The code in segment 0 is in ROM.
- Segment 1 contains another portion of Miniboot. The code in segment 1 is saved in flash. The division of Miniboot into a ROM portion and a flash portion preserves flexibility while guaranteeing a basic level of security.
- Segment 2 contains the coprocessor operating system (Linux). This code is saved in flash.

- Segment 3 contains the coprocessor application. This code is saved in flash. A segment's sensitive data is either saved in high-speed-erase battery-backed RAM (HSE BBRAM) or is encrypted and saved in regular BBRAM or in flash. The coprocessor incorporates special hardware (independent of the CPU and whose operation cannot be affected by software) that prevents the operating system and any application (that is, code in segments 2 and 3) from modifying sensitive information in flash or reading secrets in BBRAM.

The Quadient application code resides strictly in Segment 3. Segment 3 does not implement basic (or foundational) cryptographic services. However, Segment 3 does implement higher level cryptographic services such as KAS. The application code in Segment 3 links to the IBM-provided "xc" library in order to obtain cryptographic services. The "xc" library bypasses Segment 2 and obtains those cryptographic services from Segment 1 and Segment 0 which have been validated by NIST CMVP (Cert. #3164).
Segment 2 does not provide any cryptographic services that would be accessible from outside of the module.
Only the Linux kernel and the Commgr execute in Segment 2. The Commgr provides communication between applications from outside the module and the Quadient application in Segment 3. The Commgr provides this communication by managing kernel buffers and virtual buffer pointers between to and from the Quadient application in Segment 3.
The IBM xc libraries are linked to the Quadient application in Segment 3 space but obtain actual basic cryptographic services in Segments 1 and 0 via the Commgr executing in Segment 2.

The FIPS 140-2 security levels for the Module are as follows:

Table 2 – Security Level of Security Requirements

Security Requirement	Security Level
Cryptographic Module Specification	3
Cryptographic Module Ports and Interfaces	3
Roles, Services, and Authentication	3
Finite State Model	3
Physical Security	4
Operational Environment	N/A
Cryptographic Key Management	3
EMI/EMC	3
Self-Tests	3
Design Assurance	3
Mitigation of Other Attacks	N/A
Overall Security Level	3

4.1 Module Description and Cryptographic Boundary

The physical form of the Module is depicted in Figure 1 for the IBM 4767; the red outlines depict the physical cryptographic boundary. Figure 1 displays the physical attributes of the IBM 4767. The IBM 4767 consists of two (2) electrical component cards with one mounted on top of the other and then both enclosed in a secure envelope. The Module relies on a host system that supplies a PCIe interface for input/output communication. The cryptographic boundary is the red line in Figure 1 enclosing all the hardware comprising the module.

Figure 1 – IBM 4767 Cryptographic Coprocessor

The Module's ports and associated FIPS defined logical interface categories are listed in Table 3.

Table 3 – Ports and Interfaces

Physical Port	Description	Logical Interface Type
PCI Express signals:	**4-lane (x4) external**	
PCIe data/addresses	Bidirectional	Data input Data output
PCIe control	Bidirectional; PCIe v2.0 compliant "single function" device	Control input Status output
Auxiliary signals:	**Tunneled over shared flexcables**	
Serial ports	only used as status output by current IBM firmware	Status out
USB port	bidirectional; may tunnel other signals (such as Ethernet-over USB)not used by current IBM firmware	N/A (with current firmware)
PCIe power	3.3 V	Power
Battery power	variable, nominal 3.0 V	Power
External warning	host connectivity test, latching removal from host bus monitored within Module	Control input (from sensor) Status output (to host)

4.2 Mode of Operation

This Module is always used in FIPS mode.

5 Cryptographic Functionality

The Module implements the FIPS Approved and Non-Approved but Allowed cryptographic functions listed in the tables below.

Table 4 – Approved Algorithms

Cert. #	Algorithm	Mode	Description	Functions/Caveats
4814 and 4815	AES [FIPS 197]	CBC [38A]	Key Size: 256	Encrypt, Decrypt IBM bound module
Vendor Affirmed	CKG [IG D.12]	[133] Section 6.1 Asymmetric signature key generation using unmodified DRBG output		Key Generation IBM bound module
		[133] Section 7.1 Direct symmetric key generation using unmodified DRBG output		
A1559	CVL: ECC CDH Primitive [SP800-56Ar3]		P-521	ECC CDH Primitive
1486 and 1487	CVL: ECDSA [FIPS 186-4]		P-521 SHA-512	Signature Generation IBM bound module
1674 and 1675	DRBG [SP800-90Ar1]	CTR	AES-256	Deterministic Random Bit Generation Security Strength = 256 Symmetric Key Generation IBM bound module
1214 and 1215	ECDSA [FIPS 186-4]		P-521	Asymmetric Key Generation IBM bound module
			P-521 SHA-512	Signature Verification
A1558	KAS ECC [SP800-56Ar3]	Static Unified	P-521, SHA-256, Concat	Key Agreement Scheme provides 256 bits of encryption strength
3956 and 3957	SHS [FIPS 180-4]		SHA-256 and SHA-512	Digital Signature Generation, Digital Signature Verification, non-Digital Signature Applications IBM bound module

The following Approved algorithms implemented in the IBM 4767 were tested but not used:

- AES- ECB, CBC 128 and 192
- SHA-1, SHA-224, SHA-384, SHA-512/224
- CMAC w/AES with key sizes: 128, 192, 256 bits
- CMAC w/Triple-DES
- HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512
- Triple-DES

Table 5 – Non-Approved but Allowed Cryptographic Functions

Algorithm	Description
NDRNG	[Annex C] The Non-Deterministic RNG (NDRNG) output is used to seed the FIPS Approved AES-CTR-DRBG. The minimum number of bits of entropy generated by the module for use in key generation is 1024 bits.

5.1 Critical Security Parameters

All CSPs used by the Module are described in this section. All usage of these CSPs by the Module (including all CSP lifecycle states) is described in the services detailed in Section 3.

Table 6 – Critical Security Parameters (CSPs)

CSP	Description / Usage
NDRBG Seed	The entropy source to the DRBG. Provided by the bound IBM 4767 Cryptographic Coprocessor Security Module (Cert. #3164).
DRBG State	The internal state of the DRBG. Used, Generated and Zeroized as provided by the bound IBM 4767 Cryptographic Coprocessor Security Module (Cert. #3164). The values of *V* and *Key* are the secret values of the internal state.
HIK Private Key	Each Module is identified by a HSM Identity Key (HIK). This key is an ECDSA P-521 key pair. The HIK certificate is signed by the Neopost Root Key (located on the Host server). The HIK private key is generated within the Module and never leaves the module. (Does not cross HSM boundary)
Candidate HIK Private Key	When a new HIK key pair is generated, its certificate has to be signed by the Neopost Root Key (NRK) [The NRK resides on the host system and not on the Module]. Until the HIK public key is signed, it is only considered as a candidate. Once the NRK has signed the public key certificate and returned it to the Module, the Candidate HIK is “promoted” to the active HIK status – meaning, that it is to be used thereafter as the “active” HIK. (Does not cross HSM boundary)
Active Master Key	The Master Key (MK) is an AES-256 key. It is used to encrypt and decrypt a data structure related to a Virtual PSD. This data structure is stored outside of the Module. It is passed to/from the Module as needed to perform operations related to the Virtual PSD.

CSP	Description / Usage
Candidate Master Key	A new MK is generated on the Module. Until the MK key is backed up in a secure manner, it is only considered a “candidate”.
New Master Key	When a new MK is imported into a Module from another Module that created it, it is considered a New MK. At a later time, the system promotes a New MK to the Active MK.
Previous Master Key	After a New MK has been imported and promoted to the Active MK, the previous Active MK is moved to a Previous MK.
PSDSVK Private Key	Each virtual PSD has a PSD State Verification Key pair (PSDSVK) which is used to check if the PSD State data of the Virtual PSD has been tampered or not. This CSP is unique to each Virtual PSD and is not stored on the Module but is passed from the Host server in a data structure which has been encrypted with the Master Key. This data structure is called a “Security Package”.
Indicia Private Key	Each virtual PSD has an Indicia key. This does not reside on the Module but is passed in from the Host server within the Security Package described above.

5.2 Public Keys

The following table lists those public keys which are either stored on the Module or are processed by the Module in a transitory fashion.

Table 7 – Public Keys

Key	Description / Usage
NRK Public Key	Since the Neopost Root Key (NRK) private key (Residing on the Host) signs both CO messages as well as HIK certificates, the Module stores the NRK certificate in order to use the public key for verifying NRK signatures.
HIK Public Key	The HSM Identity Key (HIK) Public Key is used by other Modules to verify the authenticity of Master Key import messages as well as to derive an encryption/decryption key using ECDH key agreement.
Candidate HIK Public Key	When a new HIK key pair is generated, it is not used until it is signed by the NRK. After it is signed, it transitions from a Candidate to the Active HIK.
HAK Public Key	The Host Authentication Key (HAK) Public key is used by the Module to validate User Mode messages coming from the Host. The Module stores the HAK Public Key in a certificate and subsequently uses it to verify the HAK signatures.
PSDSVK Public Key	Each virtual PSD has a PSD State Verification Key pair (PSDSVK) Public key is used to check if the PSD State data of the Virtual PSD has been tampered or not. This Public key is unique to each Virtual PSD and is not stored on the Module but is passed from the Host server in a data structure which has been encrypted with the Master Key. This data structure is called a “Security Package”. The Private key is used for signing the PSD State data.
Indicia Public Key	Each virtual PSD has an Indicia key. This does not reside on the Module but is passed in from the Host server within the Security Package described above. The Indicia Public Key is used by the Postal Service to authenticate the signature of a Indicia.

6 Roles, Authentication and Services

6.1 Assumption of Roles

The Module supports three distinct operator roles, **Cryptographic Officer (CO)**, **Application (User)** and **Unauthenticated User**.

CO related messages are signed by the Neopost Root Key (NRK) and the signature is verified by the resident NRK certificate.

Application (User) role messages are signed by the HAK Private key residing on the Host.

Unauthenticated User role has no form of authentication as no CSPs are affected other than being zeroized.

Table 8 lists all operator roles supported by the Module. The Module does not support a maintenance role or bypass capability. The Module supports concurrent operators since concurrent CO, Application (User), and Unauthenticated User requested services can be handled internally in a completely isolated fashion. Internally to the Module, the requested services are handled without any cross-coupling of data related to the individual services.

Table 8 – Roles Description

Role ID	Role Description	Authentication Type	Authentication Data
CO	Cryptographic Officer – Message Signing	Identity	Signature
Application (User)	Secure operations	Identity	Signature
Unauthenticated User	Non-security Operations	None	None

6.2 Authentication Methods

Message Signing

The requests that are related to the CO are signed by the Neopost Root Key (NRK). The NRK is an ECC P-521 key.

User Role requests are signed by the Host Authentication Key (HAK). The HAK is a ECC _P-521 key.

Table 9 – Authentication Description

Authentication Method	Strength of Mechanism
Digital Signature ECC P-521	ECC P-521 using SHA-512 is used for the signing and verification of digital signatures. The probability that a random attempt will succeed, or a false acceptance will occur is 1/2^256, which is less than 1/1,000,000. Each NRK signature could take 5ms. That is 200 requests per second. The probability of successfully authenticating to the module within one minute through random attempts is (200*60)/2^256, which is less than 1/100,000.

The Application (User) role authentication is based on the payload data of a command. The payload data must be successfully verified for the Application (User) role to be authenticated. For example, the Import Root Key certificate service works for the Application (User) role when it is authenticated via NRK signature verification. The new NRK certificate is signed by the previous NRK. If the signature is verified, then the Application (User) role authentication was successful.

6.3 Services

All services implemented by the Module are listed in the tables below.

NOTE: For “Bound Module” services, there are references to Officers and Segments. These are defined in the [IBM-HSM]:
(IBM 4767 Cryptographic Coprocessor Security Module Non-Proprietary Security Policy) (FIPS 140-2 Cert. #3164)
For Officers, see Section 4.1 Assumption of Roles, Table 8 Role Description, pages 13 and 14.
For Segments, see Section 2.2 Firmware and Logical Cryptographic Boundary, Figure 4 Module software architecture – example usage, page 10.

Table 10 – IBM 4767 Bound Module CO Services

Service	Description
Perform Self-Tests	(Bound Module service) Perform the Power-up Self-Tests Note: This is not a callable service from the Host but is initiated upon Power-up.
Establish Officer 2	(Bound Module service) Register new Officer 2
Establish Officer 3	(Bound Module service) Register new Officer 3
Surrender Officer 2	(Bound Module service) Clear Layer 2 and 3 parameters, public keys, and persistent data
Surrender Officer 3	(Bound Module service) Clear Layer 3 parameters, public key, and persistent data
Ordinary Burn 1	(Bound Module service) Load Layer 1 (owner) public key; optionally clear Layer 2 and 3 parameters and persistent data, as defined by Segment 2/3 persistent object definitions
Ordinary Burn 2	(Bound Module service) Use the Officer2 public key; optionally clear Layer 3 parameters and persistent data; write Segment 2 code (over previous active one)
Emergency Burn 2	(Bound Module service) Clear Layer 2 and 3 persistent data; write Segment 2 code
Ordinary Burn 3	(Bound Module service) Use the Officer3 public key; write Segment 3 code (over previous active one)
Emergency Burn 3	(Bound Module service) Write Segment 3 code; clear Layer 3 persistent data
Software-induced tamper	(Bound Module service) Destroy all card-resident secrets, rendering the card unusable. The additional Software-induced tamper service is not the same thing as the actual physical tamper response mechanism, but rather, a rarely used software command to render the card inoperable by triggering the tamper response mechanism to zeroize the module. It's more like a zeroize command. It doesn't require opening the hardware to zeroize. Note that this command must be targeted to particular cards, requires IBM cooperation to create (instances are unique), and is therefore not expected to be used during the lifetime of a typical deployment.

The table below lists the two authenticated services, which are used to manage the Master Key (MK). The two services both require the passage of the MK from and to the HSM in a secure manner. The messages to carry out the specific requests are authenticated with a signature by the Neopost Root Key.

Table 11 CO Services

Service	Description
Generate Master Key Candidate	Creates an AES-256 secret Master Key Candidate which is split into two shares and exported to the Host in an encrypted format using an AES256 key.
Restore MK	Restores a Master Key to the HSM from a Host's backup of the key shares that are decrypted using an AES256 key.

Table 12 – Application (User) Services

Service	Description	Authentication
Activate New MK	Activates new Master Key that had been previously imported to the HSM.	1) Message envelope signature is checked as being signed by HAK. 2) The valid context for this command is that there be a "new MK" as a result of successfully importing a new MK. If there is no new MK, the command is rejected with an error.
Generate candidate HSM Identity Key	Generates a candidate HSM Identity Key (HIK): An EC p-521 key pair. The private key never leaves the HSM.	1) Message envelope signature is checked as being signed by HAK. 2) A "candidate" HIK and its associated CSPs is never used and has to be signed by the NRK in order to be promoted to a "usable" HIK. This service has to be followed by the "Promote Candidate HSM Identity key" service, without which this service has no material effect.
Promote Candidate HSM Identity key	After the certificate has been signed by the NRK, the certificate is returned to the HSM and the HIK candidate is made active.	1) Message envelope signature is checked as being signed by HAK. 2) The data being passed in contains a HIK certificate which is checked for a valid signature by the NRK. If this authentication fails, the command is rejected with an error. (**Strength of authentication:** Digital Signature ECC P-521)

Service	Description	Authentication
Promote Candidate MK	After the successful completion of "Generate Master Key Candidate", the MK is promoted to active or current.	1) Message envelope signature is checked as being signed by HAK. 2) This service is called subsequent to the CO role service "Generate Master Key Candidate". The digest of the MK being promoted from Candidate to Active is passed in. This indicates that the MK had been successfully decrypted and backed-up. If a wrong digest is passed in, the command is rejected with an error. This command poses no security threat to the Module as it in no way divulges the MK.
Export MK	Exports a MK in an encrypted form to a target HSM.	1) Message envelope signature is checked as being signed by HAK. 2) The data being passed in is a "target HSM" identity comprised of a HIK certificate. This certificate is ensured that it had been signed by the NRK. If this authentication fails, the command is rejected with an error. (**Strength of authentication:** Digital Signature ECC P-521)
Import MK	Imports a MK, which was encrypted by the originating HSM.	1) Message envelope signature is checked as being signed by HAK. 2) The data being passed in has "source" HIK and "target" HIK certificates. 3) Both certificates are validated for correct signature by the current NRK. Then verification is made that the "target" HIK is the importing Module's HIK. 4) A decryption key is derived (via ECDH key agreement) using the two certificates. 5) The encrypted MK being imported is decrypted and checked against its digest. If any of these authentications fails, the command is rejected with an error. (**Strength of authentication:** Digital Signature ECC P-521)

Service	Description	Authentication
Create PSD	1) Creates a unique PSD Verification key pair (ECC p521) as well as a Indicia key pair (ECC p256). These two keys are stored in a data structure (Security Package) that is then encrypted by the MK. 2) A plaintext data structure representing the Virtual PSD is created and signed by the PSD Verification key. 3) Both of these structures are then returned to the Host for storage.	1) Message envelope signature is checked as being signed by HAK. 2) When the two keys are created, they are encrypted along with their digests. (**Strength of authentication:** AES-256 encryption and SHA-512 digest). All subsequent PSD services will use the PSD Verification key pair to authenticate.
Verify PSD State	Verifies that the Virtual PSD data structure has not been modified.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Rekey PSD	Generates a new Indicia key as well the PSD Verification Key for a Virtual PSD.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Re-encrypt PSD	Re-encrypts the Virtual PSD keys with a new MK.	1) Message envelope signature is checked as being signed by HAK. 2) Further Checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)

Service	Description	Authentication
Update Registration	Updates the registration ZIP code for the Virtual PSD.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Withdraw PSD	The virtual PSD is withdrawn from service.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Add funds	Adds funds to the Virtual PSD.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Create IMI MAX Indicia	Creates a MAX indicia. MAX refers to a format defined by the US Postal Service.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)

Service	Description	Authentication
Create IMI STD Indicia	Creates a STD indicia. STD refers to a format defined by the US Postal Service.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Reset and Create IMI MAX Indicia	First adds funds to the Virtual PSD, then creates a MAX indicia. MAX refers to a format defined by the US Postal Service.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)
Reset and Create IMI STD Indicia	First adds funds to the Virtual PSD, then creates a Standard (STD) indicia. STD refers to a format defined by the US Postal Service.	1) Message envelope signature is checked as being signed by HAK. 2) Further checked: A) AES-256 decryption of the Security package, B) SHA-512 verification of its digest, C) signature verification of the PSD state via P521 ECC signature. If any of the above authentications fails, the command is rejected with an error. (**Strength of authentication:** AES-256 decryption and SHA-512 digest)

Table 13 – IBM 4767 Bound Module Unauthenticated User Services

Service	Description
Cold Boot	(Bound Module service) Reboots the module and performs power-on self-tests, triggered by the strobing of a bit in the HRCSR by a host device driver.
Query Status	(Bound Module service) Read infrastructure status, including layer owners. Reset the Module CPU (MCPU) (OS/application).
Query Status/Noreset	(Bound Module service) Read infrastructure status, including layer owners. Do not reset Module CPU.
Query Signed Health ("Get Health")	(Bound Module service) Read status, including owner identities and public keys. Resets Module CPU. It does so conditionally (only if segment 2 or segment 3 has been updated since the MCPU was last reset [in practice this is only possible for segment 3])

Service	Description
Query Signed Health/Noreset (“Query Firmware”)	(Bound Module service) Read status, including owner identities and public keys. Do not reset Module CPU.
Query Certificate	(Bound Module service) Returns the entire Seg1 certificate list, one certificate at a time (repeated calls to MB1).
Algorithm test	(Bound Module service) Hashes host-supplied data as an interactive communications/infrastructure self-test. Does not access CSPs.
Continue to Segment 1	(Bound Module service) Advance into Segment 1 code if status permits
Continue to Segment 2	(Bound Module service) Advance into Segment 2 code if possible. POST 2 self-test must have completed successfully.

The following services do not pose any vulnerabilities and hence are considered as Unauthenticated User services.

Table 14 – Unauthenticated User Services

Service	Description
Startup UDX	Used to perform application level initialization of the HSM.
Load Configuration	Loads certain performance (non-security related) configurable parameters to the HSM.
Set Trace Debug	Sets a tracing level for the HSM application software for messages to be logged to the host.
Reset UDX	Clears the CSPs from the HSM and places the HSM into an initial state. Also known as Zeroization.
Set System Time	Sets the clock on the system so that it can be synchronized with the host system.
Import Root (Public) Key	Imports a new Root key certificate (containing the public key) which is signed by the previous Root Key
Import HAK (Public) Key	Imports a new HAK key certificate (containing the public key) which is signed by the previous HAK Key. If this is the first HAK Key being imported, then it will be signed by the Root Key.
Get HSM Information (Show Status)	Obtains information about the logical status of the HSM application.
Algorithm Test (ACVP and Debug)	Runs tests for 1) ECC CDH primitive 2) ECC KAS Initiator 3) ECC KAS responder 4) ECC CDH debugging. No CSPs accessed.

Table 15 defines the relationship between access to Security Parameters and the different module services. The modes of access shown in the table are defined as:

- G = Generate: The service generates the CSP.
- O = Output: The service outputs the CSP.
- E = Execute: The service uses the CSP in an algorithm.
- E*= Conditional execution
- I = Input: The service inputs the CSP.
- Z = Zeroize: The service zeroizes the CSP.

Table 15 – Security Parameters Access by Service

Service	CSPs and Keys															
	NDRBG Seed	DRBG-State	NRK Public Key	Candidate HIK Private Key	Candidate HIK Public Key	HIK Private Key	HIK Public Key	Candidate MK	New MK	Active MK	Previous MK	Indicia Private Key	Indicia Public Key	PSDSVK Private Key	PSDSVK Public Key	HAK Public Key
Perform Self-tests (not a callable service from the Host, but is initiated upon Power-up)	*1	*1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Establish Officer 2	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Establish Officer 3	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Surrender Officer 2	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Surrender Officer 3	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ordinary Burn 1	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ordinary Burn 2	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Emergency Burn 2	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ordinary Burn 3	_	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Service	CSPs and Keys															
	NDRBG Seed	DRBG-State	NRK Public Key	Candidate HIK Private Key	Candidate HIK Public Key	HIK Private Key	HIK Public Key	Candidate MK	New MK	Active MK	Previous MK	Indicia Private Key	Indicia Public Key	PSDSVK Private Key	PSDSVK Public Key	HAK Public Key
Emergency Burn 3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Software induced tamper	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
Generate Master Key Candidate	*1	*1	E	-	-	-	E	G,O	-	-	-	-	-	-	-	-
Restore MK	-	-	E	-	-	-	E	-	-	I	-	-	-	-	-	-
Activate New MK	-	-	-	-	-	-	-	-	Z	-	-	-	-	-	-	E
Generate candidate HSM Identity Key	*1	*1	-	G,E*	G,O	E*	-	-	-	-	-	-	-	-	-	E
Promote Candidate HSM Identity key	-	-	E	Z	Z	E	-	-	-	-	-	-	-	-	-	E
Promote Candidate MK	-	-	-	-	-	-	-	Z	-	-	-	-	-	-	-	E
Export MK	*1	*1	E	-	-	E	O,E	-	-	O	-	-	-	-	-	E
Import MK	-	-	E	-	-	E	E	-	I	-	-	-	-	-	-	E
Create PSD	*1	*1	-	-	-	-	-	-	-	E	-	G,O	G,O	G,O	G,O	E
Verify PSD State	-	-	-	-	-	-	-	-	E*	E*	E*	I	I	I,E	I,E	E
Rekey PSD	*1	*1	-	-	-	-	-	-	E*	E*	E*	I,G,O	I,G,O	I,G,E,O	I,G,E,O	E
Re-encrypt PSD	*1	*1	-	-	-	-	-	-	E*	E*	E*	I,O	I	I,E,O	I	E
Update Registration	-	-	-	-	-	-	-	-	E*	E*	E*	I	I	I,E	I,E	E
Withdraw PSD	-	-	-	-	-	-	-	-	E*	E*	E*	I,E	I	I,E	I,E	E
Add Funds	-	-	-	-	-	-	-	-	E*	E*	E*	I	I	I,E	I,E	E
Create IMI MAX Indicia	-	-	-	-	-	-	-	-	E*	E*	E*	I,E	I	I,E	I,E	E

Service	CSPs and Keys															
	NDRBG Seed	DRBG-State	NRK Public Key	Candidate HIK Private Key	Candidate HIK Public Key	HIK Private Key	HIK Public Key	Candidate MK	New MK	Active MK	Previous MK	Indicia Private Key	Indicia Public Key	PSDSVK Private Key	PSDSVK Public Key	HAK Public Key
Create IMI STD Indicia	-	-	-	-	-	-	-	-	E*	E*	E*	I,E	I	I,E	I,E	E
Reset And Create IMI MAX Indicia	-	-	-	-	-	-	-	-	E*	E*	E*	I,E	I	I,E	I,E	E
Reset And Create IMI STD Indicia	-	-	-	-	-	-	-	-	E*	E*	E*	I,E	I	I,E	I,E	E
Startup UDX	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Load Configuration	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Set Trace Debug	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Reset UDX	-	-	-	Z	Z	Z	Z	Z	Z	Z	Z	-	-	-	-	-
Set System Time	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Import Root (Public) Key	-	-	I,E	-	-	-	-	-	-	-	-	-	-	-	-	-
Import HAK (Public) Key	-	-	E	-	-	-	-	-	-	-	-	-	-	-	-	I,E
Get HSM Information (Show Status)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Algorithm Test (ACVP and Debug)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Cold Boot	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Query Status	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Query Status/Noreset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Query Signed Health (“Get Health”)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Service	CSPs and Keys															
	NDRBG Seed	DRBG-State	NRK Public Key	Candidate HIK Private Key	Candidate HIK Public Key	HIK Private Key	HIK Public Key	Candidate MK	New MK	Active MK	Previous MK	Indicia Private Key	Indicia Public Key	PSDSVK Private Key	PSDSVK Public Key	HAK Public Key
Query Signed Health/Noreset ("Query Firmware")	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Query Certificate	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Algorithm test	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Continue to Segment 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Continue to Segment 2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

- *[1] = Used, Generated and Zeroized as provided by the bound IBM 4767 Cryptographic Coprocessor Security Module (Cert. #3164).

7 Self-Tests

Each time the Module is powered on, it tests that the cryptographic algorithms still operate correctly, and that sensitive data have not been damaged. Power on self–tests are available on demand by power cycling the Module. On power on or reset, the Module performs the self-tests described below. All KATs must be completed successfully prior to any other use of cryptography by the Module. If one of the KATs fails, the Module halts and a POST error code is generated. In addition to power on self-tests, the Module executes the conditional self-tests described below.

Power on Self-tests

- AES KATs: Encryption, Decryption; Modes: ECB, CBC; Key sizes: 256 bits
- ECDSA PCT: Signature Generation, Signature Verification; Curves/Key sizes: P-521 w/ SHA 512
- ECC CDH KAT: Shared secret computation; Curve size: P-521
- KAS ECC KAT: Primitive "Z" Computation KAT; Curve size: P-521
- SHA KATs: SHA-256, SHA-512
- DRBG Health Checks
- DRBG KATs: NIST SP800-90A Rev 1

- Firmware Integrity Test is verified by the bound module.

Conditional Self-tests

- DRBG Continuous Test performed when a random value is requested from the DRBG.
- NDRNG Continuous Test performed when a random value is requested from the NDRNG.
- Firmware Load using ECC P-521 signature verification.
- ECDSA self-test for mathematical functions used. PCT for all ECC keys generated.

For additional details on the self-tests, please see Table 13 - Power on Self-tests and Table 14 - Conditional Self-Tests in the *IBM 4767 Cryptographic Coprocessor Security Module Non-Proprietary Security Policy* (CMVP Cert. #3164).

8 Physical Security Policy

Intrusions, which destroy card secrets through an internal, independent action, are host-observable as system administration events. System administrators may notice tamper detection through unusual Module start-up, e.g., the card failing to initialize. It is recommended that all types of tamper events be investigated by crosschecking the tamper event with other logs.
The types of tamper events are listed in the following table:

Table 16 – Physical Security Inspection Guidelines

Physical Security Mechanism	Severity/Effect	Recommended Frequency of Inspection	Test Guidance
Hard Tamper	Zeroization	N/A (Automatic)	N/A
Soft Tamper	Module Reset	N/A (Automatic)	N/A
External Warning	Warning	Module Restart	Warning is exposed to the host
Low Battery	Warning	As frequent as possible	Replace as soon as possible

Physical security is constantly monitored through a tamper detection / response envelope with tamper response and zeroization circuitry. No external physical monitoring is required. Environmental failure protection (EFP) is included.

9 Operational Environment

The Module is designated as a non-modifiable operational environment under the FIPS 140-2 definitions. The Module includes a firmware load service to support necessary updates. New firmware versions within the scope of this validation must be validated through the FIPS 140-2 CMVP. Any other firmware loaded into this Module is out of the scope of this validation and require a separate FIPS 140-2 validation.

10 Mitigation of Other Attacks Policy

N/A – the Module does not claim to mitigate other attacks.

11 Security Rules and Guidance

The Module design corresponds to the Module security rules. This section documents the security rules enforced by the cryptographic Module to implement the security requirements of this FIPS 140-2 Level 3 Module.

1. The Module will provide three (3) distinct operator roles: Cryptographic Officer, Application (User), and Unauthenticated User.
2. The Module will provide identity-based authentication.
3. The Module will clear previous authentications on power cycle. This is accomplished by clearing RAM and all running applications.
4. When the Module has not been placed in a valid role, the operator will not have access to any cryptographic services.
5. The operator will be capable of commanding the bound Module to perform the power on self-tests by cycling power or resetting the Module.
6. Power on self-tests do not require any operator action.
7. Data output will be inhibited during key generation, self-tests, zeroization, and error states. This is accomplished by the Custom Communication Hardware in the PCIe interface path.
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 CSPs are zeroized by the zeroization service.
10. The module supports concurrent operators.
11. The Module does not support a maintenance interface or role.
12. The Module does not support manual key entry.
13. The Module does not have any 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 store any plaintext CSPs.
16. The Module does not output intermediate key values.
17. An authentication limit counter is not applicable because there is no operator authentication. Messages are authenticated using message signatures.

18. The module is seeded and reseeded automatically by the IBM hardware.
19. The module does not provide bypass services or ports/interfaces.

12 References and Definitions

The following standards are referred to in this Security Policy.

Table 17 – References

Abbreviation	Full Specification Name
[FIPS140-2]	*Security Requirements for Cryptographic Modules*, May 25, 2001
[IG]	*Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*
[108]	*NIST Special Publication 800-108, Recommendation for Key Derivation Using Pseudorandom Functions (Revised), October 2009*
[131A]	*NIST Special Publication 800-131A Rev. 2, Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, 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, December 2012*
[135]	*National Institute of Standards and Technology, Recommendation for Existing Application-Specific Key Derivation Functions, Special Publication 800-135rev1, December 2011.*
[186-4]	*National Institute of Standards and Technology, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-4, July, 2013.*
[186-2]	*National Institute of Standards and Technology, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-2, January 2000.*
[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*

Abbreviation	Full Specification Name
[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*
[38B]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, Special Publication 800-38B, October 2016*
[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 800-56A Revision 2, Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography, March 2019*
[90Ar1]	*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.*
[IBM-HSM]	*IBM 4767 Cryptographic Coprocessor Security Module* *Non-Proprietary Security Policy, Version SP 1.3, January 15, 2021.* *https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp3164.pdf*

Table 18 – Acronyms and Definitions

Acronym	Definition
NRK	Neopost Root Key – Neopost managed key for signing HIKs
HIK	HSM Identification Key – Uniquely identifies a particular HSM and is used for authenticating a HSM as being a Quadient HSM. Also used for ECDH key agreement.
MK	Master Key - Used to encrypt and decrypt Virtual PSDs