T 613-454-2222
E INFO@CRYPTO4A.CA

WWW.CRYPTO4A.COM

QASM CRYPTOGRAPHIC MODULE:
NON-PROPRIETARY SECURITY POLICY

DATED: JUNE 5, 2022

VERSION: 1.6

QASM 1.0

F/W: 2.1.0

Version	Date	Author	Description of Change
0.4	06/05/20	Olivier Couillard	Last GDoc version of security policy
0.5	04/06/20	Olivier Couillard	Changes to reflect comments from review by Lightship, following call on May 21 2020
0.6	14/07/20	Olivier Couillard	Changes to reflect comments from review by Lightship of version 0.5, following call on July 16 2020. Also added first version of Roles, Services and Authentication.
0.7	25/08/20	Olivier Couillard	Changes to reflect comments from review by Lightship of version 0.6, following call on August 11 2020. Also added documentation about CSPs.
0.8	23/11/20	Olivier Couillard	General update of the security policy based on new development as well as answering new concerns expressed within the "Outstanding VEs and SP Observation" document provided by Lightship on November 20 2020.
Draft	30/11/20	Bruno Couillard	Official version. Expanded algorithm support info tables and minor editorial changes.
Draft-2	7/12/20	Olivier Couillard	Editorial changes
Draft-3	11/12/20	Olivier Couillard	Editorial changes
Draft-4	14/12/20	Olivier Couillard	More editorial changes
Draft-5	17/12/20	Olivier Couillard	More editorial changes
1.0	10/03/21	Olivier Couillard	Minor changes following functional testing
1.1	27/01/22	Olivier Couillard	Minor changes following round 1 review from CMVP
1.2	08/02/22	Olivier Couillard	Minor changes following comments from Lightship about version 1.1.
1.3	18/02/22	Olivier Couillard	Minor changes following comments from Lightship about version 1.2.
1.4	05/04/22	Jim Goodman	Minor changes to address comments from NIST round 2.
1.5	20/05/22	Bruno Couillard	Minor changes to address comments from NIST round 3.
1.6	5/6/22	Bruno Couillard	Minor changes to address comments from NIST round 4.

Trademarks, Copyrights, and Third-Party Software

Disclaimer

All information herein is either public information or is the property of and owned solely by Crypto4A Technologies Inc. which shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information. Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/or industrial property rights of or concerning any of Crypto4A Technologies Inc's information.

This document can be used for informational, non-commercial, internal and personal use only provided that:

> The copyright notice below, the confidentiality and proprietary legend and this full warning notice appear in all copies.

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided "AS IS" without any warranty of any kind. Unless otherwise expressly agreed in writing, Crypto4A Technologies Inc. makes no warranty as to the value or accuracy of information contained herein.

Crypto4A Technologies Inc hereby disclaims all warranties and conditions with regard to the information contained herein, including all implied warranties of merchantability, fitness for a particular purpose, title and non-infringement. In no event shall Crypto4A Technologies Inc. be liable, whether in contract, tort or otherwise, for any indirect, special or consequential damages or any damages whatsoever including but not limited to damages resulting from loss of use, data, profits, revenues, or customers, arising out of or in connection with the use or performance of information contained in this document.

Crypto4A Technologies Inc. does not and shall not warrant that this product will be resistant to all possible attacks and shall not incur, and disclaims, any liability in this respect. Even if each product is compliant with current security standards in force on the date of their design, security mechanisms' resistance necessarily evolves according to the state of the art in security and notably under the emergence of new attacks. Under no circumstances, shall Crypto4A Technologies Inc. be held liable for any third-party actions and in particular in case of any successful attack against systems or equipment incorporating Crypto4A Technologies Inc. products. Crypto4A Technologies Inc. disclaims any liability with respect to security for direct, indirect, incidental or consequential damages that result from any use of its products. It is further stressed that independent testing and verification by the person using the product is particularly encouraged, especially in any application in which defective, incorrect or insecure functioning could result in damage to persons or property, denial of service or loss of privacy.

TABLE OF CONTENTS

1 PREFACE

This document deals with operations and capabilities of the Crypto4A Quantum Assured Security Module (QASM) Cryptographic Module in the terminology specified by the Federal Information Processing Standards (FIPS) Publication number 140-2 (also known as FIPS PUB 140-2), 'Security Requirements for Cryptographic Modules', dated March 12th, 2002

General information on Crypto4A's QASM HSM, the QxEDGE and other Crypto4A products is available from the following sources:

> the Crypto4A internet site contains information on the full line of available products at

https://crypto4a.com.

> product manuals and technical support literature is available from the Crypto4A Customer Support Portal

at https://support.crypto4a.com.

> technical or sales representatives of Crypto4A can be contacted through one of the channels listed on

https://crypto4a.com/about-us/support/

NOTE You require an account to access the Customer Support Portal.

2 INTRODUCTION

2.1 PURPOSE

This non-proprietary document describes the security policies enforced by the QASM Cryptographic Module in order to meet the level 3 security requirements of the FIPS 140-2 [1]. This document applies to Hardware Versions QASM 1.0 with Firmware Versions 2.1.0.

2.2 SCOPE

The security policies described in this document applies to the public key authentication (FIPS Level 3) configuration of the QASM Cryptographic Module, and does not include any security policy that may be enforced by the host appliance or an attached server. Any firmware loaded into this module that is not shown on the module certificate is out of the scope of this validation and requires a separate FIPS 140-2 validation.

2.3 DETAILS

Module Name: QASM v1.0
Standard: FIPS 140-2
Status: Active
Overall Level: 3
Module Type: Hardware
Embodiment: Multi-Chip Standalone

Table 1 summarizes the different elements that will be covered in section 3 with their associated security level.

Table 1 - Summary of Security Level

Security Requirement	Level
Cryptographic Module Specification	3
Cryptographic Ports and Interfaces	3
Roles, Services and Authentication	3
Finite State Machine	3
Physical Security	3+EFP
Operating Environment	N/A
Cryptographic Key Management	3
EMI/EMC	3
Self-tests	3
Design Assurance	3
Mitigation of Other Attacks	3

2.4 VENDOR

Crypto4A Technologies Inc.
1550A Laperriere Ave.
Ottawa ON K1Z 7T2
Canada

2.5 RELATED DOCUMENTS

1 **“FIPS PUB 140-2: Security Requirements for Cryptographic Modules”. NIST, Mars, 2002.**

2.6 ACRONYMS AND ABBREVIATIONS

CA	Certificate Authority
CSP	Critical Security Parameter
ECC	Elliptic Curve Cryptography
FIPS	Federal Information Processing Standards
GPIO	General Purpose I/O
HMAC	Hash based Message Authentication Code
HSM	Hardware Security Module
HSS	Hash-based Signature Scheme
I/O	Input/Output
I2C	Inter-Integrated Circuit
KE	Key Exchange
KEM	Key Encapsulation Mechanism
KW	Key Wrapping
KWP	Key Wrapping with Padding
OS	Operating System
PCIe	Peripheral Component Interconnect express
PE	Processing Engine
PKC	Public Key Certificate
PKCS	Public Key Cryptographic Standard
PMSS	Platform Management Sub-System
PPSIN	Pulse per Second Input
PPSOUT	Pulse per Second Output
PU	Public User

QASM	Quantum Assured Security Module
REFCLKIN	Reference Clock Input
RSA	Rivest Shamir Adelman
SBC	Single Board Computer
TA	Trust Anchor
TKA	Tamper Key All
TKU	Tamper Key Unit
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
ZKU	Zeroize Key Unit

3 MODULE OVERVIEW

The QASM Cryptographic Module is a high-assurance, tamper-resistant Hardware Security Module (HSM) which secures sensitive data and critical applications by storing, protecting and managing cryptographic keys. It provides end users with industry-leading security and performance. The QASM can be embedded into the QxEDGE family of security appliances for FIPS 140-2 validated key security or be used as a standalone co-processor for an external computing system using a docking-station adapter.

The QxEDGE is a 1U, 19-inch rack mountable appliance form factor. In its base configuration, a QxEDGE comprises four Single Board Computers (SBCs) and a central QASM HSM. The QxEDGE SBCs are based on Intel i7 quad core processors, and are configured with 16 GB of RAM and 256 GB of SSD storage. Each one of the SBCs implements a hardened Linux-based Operating System (OS). Each one of the SBCs have a direct and independent connectivity with the QxEDGE central QASM. This allows local software applications running on these independent SBCs to interact with the QASM as four independent computing systems.

3.1 MODULE SPECIFICATION

The QASM is a security level 3 multi-chip standalone cryptographic hardware module. Figure 1 shows the QASM as well as the cryptographic boundary in red. The aluminum enclosure containing the QASM acts as a physical boundary, and the cryptographic boundary is contained within this physical boundary.

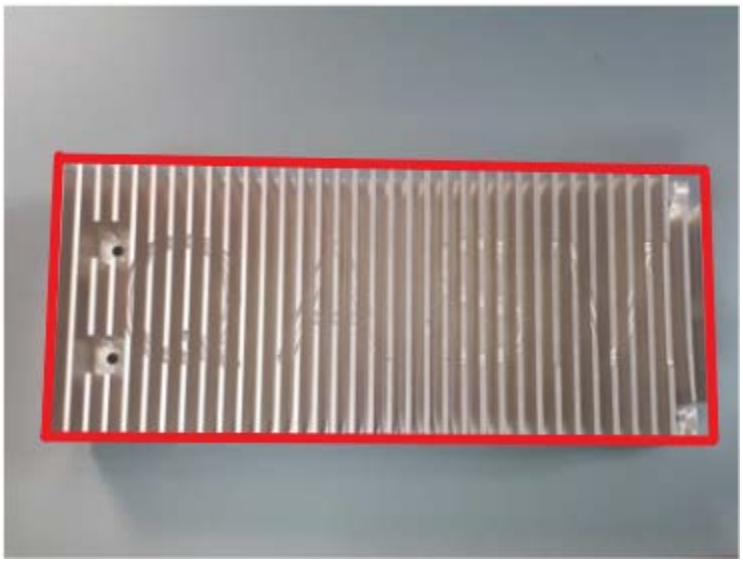

Figure 1 - Picture of QASM and Physical Boundary

3.2 PORTS AND INTERFACES

The QASM implements a single external port (referred to as the “AMI”) consisting of a rectangular grid connector designed to contain a variety of signals and buses to support a

variety of interfaces, from UART to PCIe. Control inputs, data inputs, data outputs, status outputs and power inputs all go through this one connector. Figure 2 shows the AMI connector.

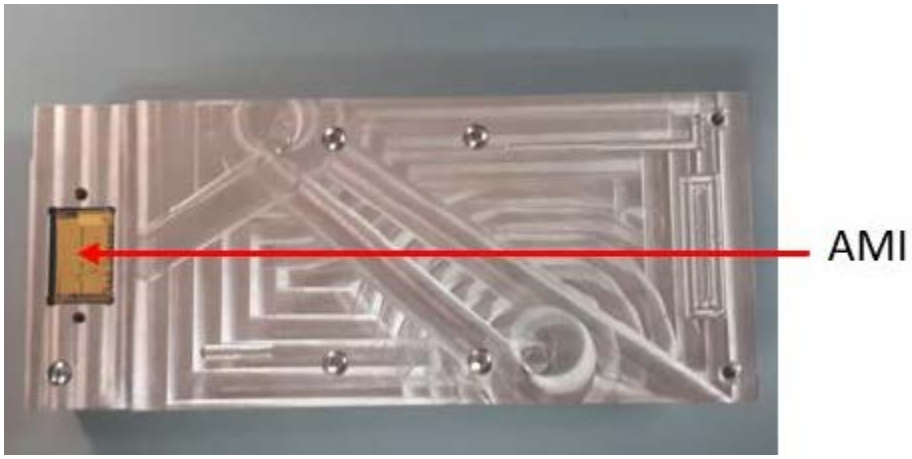

Figure 2 - Port of the QASM (AMI connector)

The main interfaces implemented by the QASM AMI connector are: a Universal Asynchronous Receiver-Transmitter (UART) interface, four Universal Serial Bus (USB) version 3.0 interfaces and six Peripheral Component Interconnect express (PCIe) dual lane buses that allow the QASM to interact with external system management support[1], the SBCs and other external interfaces. Physical security in the form of fuse circuits is implemented to protect the QASM from a power surge on power and tamper signals. Table 2 shows the various interfaces.

Table 2 - Description of Ports and Interfaces

FIPS 140-2 Interface	Physical Interface	Description
Data Input	6x 2-lane PCIe	Data I/O with the SBCs
	4x USB 3.0	Data I/O with the SBCs
	2x I2C	Data I/O with Platform Management Sub-System (PMSS)
	1x UART	Data I/O with PMSS
Data Output	6x 2-lane PCIe	Data I/O with the SBCs
	4x USB 3.0	Data I/O with the SBCs
	2x I2C	Data I/O with PMSS
	1x UART	Data I/O with PMSS
Control	6x 2-lane PCIe	Data I/O with the SBCs
	4x USB 3.0	Data I/O with the SBCs

[1] The QASM is typically connected to Crypto4A's QxEDGE, which has a Platform Management Sub-System (PMSS) to control power, fans, and other components of the QxEDGE. Note that the PMSS falls outside the scope of this validation.

FIPS 140-2 Interface	Physical Interface	Description
	1x REFCLKIN	Reference clock input signal (from atomic clock)
	1x PPSIN	Pulse per second (from atomic clock)
	1x REFCLKOUT	Reference clock output signal (from FPGA's internal clock)
	1x PPSOUT	Pulse per second (to PMSS)
	2x I2C	Data I/O with PMSS
	1x UART	Data I/O with PMSS
	24x GPIO	2 pins are used as a signal loop to detect the presence of the PMSS, 9 pins can be used to trigger zeroization, 1 pin is used to reset the PMSS, 1 is used to enable the 12V input, 11 are reserved for future use.
Status	6x 2-lane PCIe	Data I/O with the SBCs
	4x USB 3.0	Data I/O with the SBCs
	2x I2C	Data I/O with PMSS
	1x UART	Data I/O with PMSS
	6x GPIO	3 pins are used to indicate if a zeroization or tamper event occurred, 1 pin indicates if the power input is good, 1 pin indicates that the FPGA has finished loading its image, 1 pin indicates if the 5V supply is provided by the battery
Power	5V and 12V	Power inputs from PMSS

3.3 ROLES, SERVICES AND AUTHENTICATION

3.3.1 Roles

The Crypto4A QASM Cryptographic Module supports the following roles:

Table 3 – QASM Cryptographic Module Roles

Role (C4A)	Role (FIPS)	Responsibilities
OWNER (O)	Crypto Officer (CO)	There is a single owner per QASM. The QASM owner may relinquish its role to a different owner under a secure owner authority transfer. The owner is in charge of the proper operations of the QASM including its ability to perform firmware update and zeroizing the QASM. The owner is authenticated using an ECDSA P-384 digital signature.
Security Officer (SO)	Crypto Officer (CO)	There is a single security officer per QASM. The QASM owner may change the SO under a secure SO authority transfer. The SO manages the access policy and is responsible for adding and removing users. The

Role (C4A)	Role (FIPS)	Responsibilities
		SO is authenticated using ECDSA P-224, P-256 or P-384 digital signatures.
Access User (AU)	User (U)	There can be many access users. Each AU has a unique set of credentials they can use to login the QASM and perform various cryptographic operations. The SO can add or remove AUs as needed. AUs are authenticated using ECDSA P-224, P-256 or P-384 digital signatures
Public User (PU)	Unauthenticated User (UU)	Public users have access to a limit number of services since they are not authenticated to the QASM.

Since the QASM uses identity-based and signature-enabled authentication, an identity must possess an appropriate signing private key in order to assume the responsibilities associated with a role. Note that there is no maintenance role because there is no maintenance interface for the QASM.

3.3.2 Services

Table 4 summarizes all the different services provided by the QASM. For each service, we define the cryptographic keys and CSPs that are concerned, along with the type of access (read, write, use and erase). We mark with an "X" the roles for which the corresponding service is allowed, and we indicate the authentication data required for performing the service. Note that in order to login, one must go through the authorization process described in section 3.3.4[2]. The underlying assumption for the services described in Table 4 is that the QASM has not been tampered or has not entered an error state.

For some operations, a more thorough authentication process may be required in order to provide additional security on some specific objects. For instance, users can customize the restrictions applied to various cryptographic objects upon their generation such that only they will be allowed to use those objects. Furthermore, they can establish quorums of users to share the control of an object with multiple entities. Regardless, the following table focuses on authenticated services rather than expand on object-specific restrictions.

[2] To be clear, the login process with the QASM does not follow the traditional username/password scheme. Instead, it relies on a challenge-response signature-based scheme which is described in section 3.3.4.

Table 4 - List of Services

Service	Cryptographic Keys and CSPs	Type(s) of Access	Roles				Authentication Data (If not N/A, strength is ECDSA P-384)
			O	SO	AU	PU	
Zeroize module	All cryptographic keys and CSPs[3]	Erase	X				Owner public key
Change mode of operation	Owner Policy	Write	X				Owner public key
	Owner public key	Use					
Change policy	Owner Policy	Write	X				Owner public key
	Owner public key	Use					
Transfer ownership	Owner public key (new)	Write	X				Owner public key (old)
	Owner public key (old)	Use					
Firmware update	C4A_TA_FW, QASM_FW_UPDATE, C4A_FW_UPDATE	Use	X				Owner public key
Add SO	Access policy	Write	X				Owner public key
	Owner public key	Use					
Remove SO	Access policy	Write	X				Owner public key
	Owner public key	Use					
Add AU	Access policy	Write	X	X			O/SO public key
	SO public key	Use					
Remove AU	Access policy	Write	X	X			O/SO public key
	SO public key	Use					
Login	Access policy	Write	X	X	X		O/SO/AU public key
	O/SO/AU public key	Use					
Logout	Access policy	Write	X	X	X		O/SO/AU public key
	O/SO/AU public key	Use					
Get status	N/A	N/A	X	X	X	X	N/A
Self-test	N/A	N/A	X	X	X	X	N/A
Generate random data	DRBG C, DRBG entropy input, DRBG instantiation nonce, DRBG	Use, Read	X	X	X		Requires login

[3] CSPs associated to the Secure Boot service are zeroized as part of physical protection responses described in section 3.4.

Service	Cryptographic Keys and CSPs	Type(s) of Access	Roles				Authentication Data (If not N/A, strength is ECDSA P-384)
			O	SO	AU	PU	
	personalization string, DRBG seed, and DRBG V						
Key generation	DRBG C, DRBG entropy input, DRBG instantiation nonce, DRBG personalization string, DRBG seed, and DRBG V	Use	X	X	X		Requires login
	Symmetric Keys	Write					
Key pair generation	DRBG C, DRBG entropy input, DRBG instantiation nonce, DRBG personalization string, DRBG seed, and DRBG V	Use	X	X	X		Requires login
	Asymmetric Keys	Write					
Wrap key	Wrapping key	Use	X	X	X		Requires login
	Unwrapped key	Write					
Unwrap key	Unwrapping key	Use	X	X	X		Requires login
	Wrapped key	Write					
Export key (MofN)	Exported key	Read	X	X	X		Requires login
	Custodians' public keys	Use					
Import key (MofN)	Imported key	Write	X	X	X		Requires login
Hash	N/A	N/A	X	X	X		Requires login
Keyed-hash	Symmetric keys	Use	X	X	X		Requires login
Symmetric encryption/decryption	DRBG C, DRBG entropy input, DRBG instantiation nonce, DRBG personalization string, DRBG seed, and DRBG V and symmetric keys	Use	X	X	X		Requires login
Key derivation (ECDH)	ECC public keys	Use	X	X	X		Requires login
Signature generation	DRBG C, DRBG entropy input, DRBG instantiation nonce, DRBG personalization string, DRBG seed, and DRBG V	Use	X	X	X		Requires login

Service	Cryptographic Keys and CSPs	Type(s) of Access	Roles				Authentication Data (If not N/A, strength is ECDSA P-384)
			O	SO	AU	PU	
	and private key of signature						
Signature verification	Public key of signature	Use	X	X	X		Requires login
Store data object	Data object	Write	X	X	X	X	N/A
Read data object	Data object	Read	X	X	X	X	N/A
Archive	Symmetric key	Read	X	X	X		Requires login
	Asymmetric keys	Use					
Unarchive	Asymmetric keys	Use	X	X	X		Requires login
	Symmetric keys	Write					
Create authority	Authority public key	Write	X	X	X		Requires login
Create quorum	Authorities	Use	X	X	X		Requires login
Secure Boot	TKA-0, TKA-1, TKH, TKU, ZKU, PSK, PPK, SSK, SPK, XEK, TWK-0, TWK-1, XWK, ALTEK-0, ALTEK-1, AL-Key, BO-Key, CH-Key, PUF, TMK	Use	X	X	X	X	N/A

3.3.3 Unauthenticated Roles

As shown in Table 4, there are only four services which do not require authentication. Those services can be performed by a public user. Specifically, the services are to get the status information of the QASM, to perform the self-tests, to store and to read a data object.

3.3.4 Authentication

The QASM performs authentication services in two steps to ensure identity-based authentication. The first step consists of preparing a request containing information about the service to be performed[4] and the second step consists of signing the request with the appropriate authority or authorities and submitting the signature(s). Once the signature(s) is/are submitted and validated, the QASM will perform the corresponding service. The OWNER must use an ECDSA P-384 digital signature for this process. The SO/AU may utilise ECDSA P-224, P-256 or P-384 digital signatures.

[4] Such services include switching in and out of FIPS mode, login/logout, adding/removing a SO, adding/removing a AU, zeroizing the module, performing a firmware upgrade, and changing owner.

Table 5 – Strength of Authentication

Authentication Mechanism	Type of authentication	Strength of Mechanism
ECDSA P-384 signature	Identity based, owner	This authentication mechanism uses ECDSA P-384. According to NIST's SP800-57 Table 2[1], the security strength of ECDSA is 192 bits. An attacker would therefore need to guess the user's key to defeat the QASM authentication method. Each guess would have a probability of $1/2^{192}$ of success, which is well below the threshold set by NIST of $1/10^6$. Given that an authorizationprocess requires the verification of at least one signature, and that the QASM cannot verify more than 2^{16} signatures per minute, the QASM cannot process more than 2^{16} authorization processes per minute. This gives the probability of success of an attacker guessing the appropriate authentication data in a minute to be $1/2^{176}$. This is well below the threshold set by NIST of $1/10^6$.
ECDSA P-224 signature (worst case)	Identity based, authorities	This authentication mechanism uses ECDSA P-224. According to NIST's SP800-57 Table 2[5], the security strength of ECDSA is 112 bits. An attacker would therefore need to guess the user's key to defeat the QASM authentication method. Each guess would have a probability of $1/2^{112}$ of success, which is well below the threshold set by NIST of $1/10^6$. Given that an authorizationprocess requires the verification of at least one signature, and that the QASM cannot verify more than 2^{16} signatures per minute, the QASM cannot process more than 2^{16} authorization processes per minute This gives the probability of success of an attacker guessing the appropriate authentication data in a minute to be $1/2^{96}$. This is well below the threshold set by NIST of $1/10^6$.

3.4 PHYSICAL SECURITY

The QASM is contained in a strong aluminum enclosure that provides tamper-evidence. Various sensors inside the enclosure monitor a number of event signals. The QASM enclosure is formed by a two pieces cover set with a single opening that provides connectivity to up to six (6) external computing system(s). The QASM enclosure forms the cryptographic boundary and also helps to isolate the QASM from an electromagnetic point of view, thereby minimizing side channel information leakage. The QASM enclosure also integrates a built-in heatsink. Figure 3 shows the QASM with its factory applied tamper seals. The Crypto Officer shall establish a regular cadence for the inspection of the tamper seals, to ensure that they have not been

[5] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf

breached. In the event that any of the tamper seals have been breached, the module should be returned to Crypto4A for service.

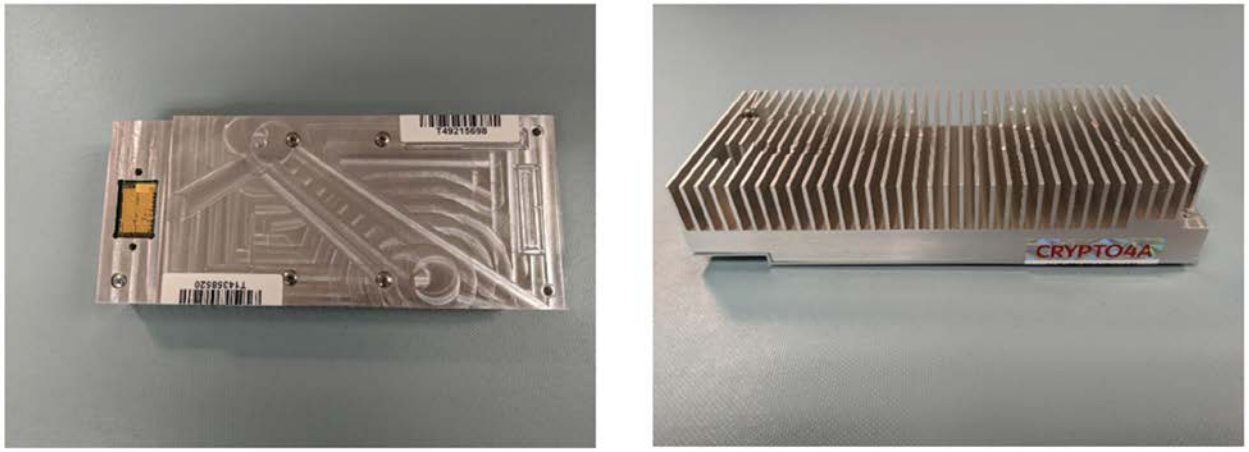

Figure 3 - Tamper Seals

3.4.1 External Event

The QASM support the input of an external event signal which is monitored in both power-on and power-off state.

If a status change of the signal is detected the QASM will zeroize its internal User Key Encryption Key (aka ZKU), reset itself, clear all working memory and log the event. The module can be reset and placed back into operation when the external event signal is removed. Once re-instated, all previously held user keys will have been erased.

3.4.2 Fault Tolerance

The QASM can detect the loss of the main power input. During the powered-off condition, the QASM continues to monitors a series of event signals that could indicate loss of physical integrity. Unless an event indicates a breach of its integrity, the QASM maintains itself in a state that can be placed back into operation when power is restored without compromise of its functionality or permanently stored data.

3.4.3 Removal Detection

The QASM detects removal from its attached external computing system in both the powered-on state and the powered-off state.

If the QASM is removed from the attached computing system it will zeroize its internal appliance binding key (aka TKU) reset itself, clear all working memory and log the event. The module will require a factory reactivation in order to be usable again.

3.4.4 EFP

The QASM is designed to sense and respond to out-of-range temperature[6] as well as voltage conditions[7]. Temperature and voltage sensing are performed in both power-on and powered-off state.

In the event of an excursion outside the operational temperature or voltage range, the QASM will erase its internal TKU, ZKU and Master Tamper Key (aka TKA), reset itself, clear all of its working memory and log the event.

Exposure to out-of-range temperatures or over voltage conditions is a non-recoverable event and the QASM cannot be placed back into operation.

Note, under-voltage conditions are treated as a power cycle. Due to its fault tolerant features, the QASM will return to its initial operating state once the under-voltage condition is cleared.

3.4.5 Physical Protection

The QASM is designed to sense and respond to breach attempts at its physical integrity that may compromise the QASM or the keys it protects. Cutting, drilling, milling, grinding, or dissolving attempts at the QASM enclosure will result in a permanently damaged QASM and a very high probability of being detected, resulting in the immediate zeroization of all plaintext CSPs.

3.5 OPERATING ENVIRONMENT

The QASM is a hardware security module running a non-modifiable firmware. The requirements for a modifiable operating environment do not apply.

3.6 CRYPTOGRAPHIC KEY MANAGEMENT

3.6.1 FIPS-Approved Algorithm Implementations

The FIPS-Approved algorithms implemented by the module are listed in the table below:

Table 6 – FIPS-Approved Algorithms

Approved Cryptographic Algorithms	Certificate Number
Symmetric Encryption	
AES: GCM[8], ECB, CBC, Key Wrap (KW), Key Wrap with Padding (KWP) (128, 192, 256 bits)	A893
Hash	

[6] Temperatures below -11 degrees Celsius and above 71 degrees Celsius are considered out-of-range and will result in a shutdown.
[7] Voltages above 13.6V are considered out-of-range and will result in a tamper. Voltages below 8.3V will result in a shutdown.
[8] The module generates IVs internally using the Approved DRBG. The IVs are at least 96 bits in length.

Approved Cryptographic Algorithms	Certificate Number
SHA: SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512-224, SHA2-512-256, SHA3-224, SHA3-256. SHA3-384, SHA3-512	A893
Message Authentication	
HMAC: HMAC-SHA1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512	A893
Asymmetric	
RSA: Key Generation, Signature Generation PSS and PKCS1.5, Signature Verification PSS and PKCS1.5 Wrap OAEP and Unwrap OAEP (2048, 3072, 4096 and 8192 modulus[9]) Signature Verification PSS and PKCS1.5 (1024)	A893
ECDSA: Key Generation, Signature Generation, Signature Verification Curves: P-192 (Signature verification only), P-224, P-256 and P-384	A893
KAS-ECC: staticUnified | Domain Parameter Generation: P-224, P-256, P-384 (provides between 112 and 192 bits of encryption strength) KDFs: ANSI X9.63 and NIST SP-800 56C.	A893
KTS-RSA: Modulus size: 2048, 3072, 4096 and 8192 (provides between 112 and 196 bits of encryption strength)	A893
KTS (AES KW / AES KWP): Key Sizes: 128, 192, 256 (provides between 128 and 256 bits of encryption strength)	A893
Random Number Generation	
DRBG (Hash) ENT (P) CKG[10]	A893 Vendor Affirmed
Key Derivation Function	
CVL (Cert. #A893) (ANSI X9.63 KDF)	A893

[9] Modulus 8192 is vendor affirmed as test vectors were not available at the time of writing this security policy.
[10] Resulting symmetric keys and seeds used for asymmetric key generation are an unmodified output from an Approved DRBG.

3.6.2 Non-FIPS Approved Algorithm Implementations

The following Non-FIPS Approved algorithms are either not implemented by the module when operating in FIPS-approved mode or they are implemented but complemented with FIPS-approved algorithms. This is the case for the two Post Quantum Algorithms, HSS and McEliece which remain implemented in FIPS approved mode, but are always used in tandem with a FIPS approved algorithms. See 3.10 Mitigation Of Other Attacks for more information on this approach. This is in conformance with the Implementation Guidance (IG) 1.23 scenario 3. Table 7 lists the algorithms that are not FIPS approved, but that are allowed in FIPS mode. Table 8 lists the algorithms that are not FIPS approved, and that are not allowed in FIPS mode.

Table 7 - Non-FIPS Approved and Allowed Algorithms

Non-Approved Cryptographic Algorithms	Certificate Number
Post-Quantum Algorithms	
McEliece Key Encapsulation Mechanism (KEM) (Only if used together with KTS-RSA)	N/A
HSS digital signature (Only if used together with ECDSA/RSA signature)	N/A

Table 8- Non-FIPS Approved and Non-Allowed Algorithms

Non-Approved Cryptographic Algorithms	Certificate Number
Post-Quantum Algorithms	
McEliece Key Encapsulation Mechanism (KEM) (Used alone)	N/A
HSS digital signature (Used alone)	N/A
Miscellaneous	
RSA-1024 (Key Generation, Signature Generation, encryption and decryption)	N/A
ECDSA P-192 (Key generation, signature generation)	N/A
HMAC counter mode (SP800-108)	N/A

3.6.3 M of N Based Key Archival & Restore

Highly critical keying material may be exported from the QASM for long term archival using a secure technique based on an MofN process designed around the Shamir Secret Sharing scheme.

3.7 CRITICAL SECURITY PARAMETERS

Table 9 shows the list of CSPs used in the module. Note that two types of owners are shown. During the manufacturing process, C4A is designated as the owner of the QASM. However, upon receiving the QASM, a customer can request that the ownership be transferred to them, at which point C4A would submit a signature using its private key accepting the transfer of ownership. After the transfer, the customer is now the owner of the QASM, and the C4A owner is removed. Apart from this minor exception, all other CSPs listed in Table 9 are always present on the QASM (assuming the QASM is operating in the FIPS-approved mode of operation, and hasn't been tampered).

Table 9 - List of CSPs

Keys/CSPs	Type	Generation/Input	Output	Description
C4A_TA_FW	Certificate /TAF	Loaded at Manufacturing	Certificate output in plaintext.	TA used for validating signature on firmware update. Contains both HSS and ECDSA public key.
C4A_FW_UPDATE	Secret Key AES-256	Loaded at Manufacturing	Not output	Valid for F/W update security for QASM built.
QASM_FW_UPDATE	Secret Key AES-256	Generated during manufacturing	Not output	F/W update key specific to a given QASM.
Owner Public Key	ECDSA P-384	Generated during manufacturing or imported during a transfer of owner	Output in plaintext	Public key used to authenticate the owner.
Owner Policy	Policy	Generated during manufacturing	Output in plaintext	Document used to define policies related to the QASM. Only the owner can modify this document
Access Policy	Policy	Generated by the owner	Output in plaintext	Document used to keep track of the login/logout state and the list of users.
DRBG Personalization string	88-bytes value	Randomly generated during the manufacturing process (pre-configured)	N/A	Personalization string used to instantiate the DRBG
DRBG Instantiation nonce	64-bytes value	Randomly generated during the previous instantiation	N/A	Nonce used to instantiate the DRBG
DRBG Entropy Input	256-bytes	Randomly generated during instantiation	N/A	Entropy value used to instantiate the DRBG
DRBG V	888-bits	Randomly generated	N/A	Part of the secret state of the DRBG.
DRBG C	888-bits	Randomly generated	N/A	Part of the secret state of the DRBG.
DRBG seed	888-bits	Randomly generated	N/A	Random seed data generated from conditioned output from the module's TRNGs

Keys/CSPs	Type	Generation/Input	Output	Description
TKA-0	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TKA-1	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TKH	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TKU	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
ZKU	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
PSK	Private RSA-4096	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
PPK	Public RSA 4096	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
SSK	Private RSA-4096	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
SPK	Public RSA 4096	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process

Keys/CSPs	Type	Generation/Input	Output	Description
XEK	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TWK-0	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TWK-1	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
XWK	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
ALTEK-0	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
ALTEK-1	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
AL-Key	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
BO-Key	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
CH-Key	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process

Keys/CSPs	Type	Generation/Input	Output	Description
PUF	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process
TMK	Secret Key AES-256	Randomly generated during the manufacturing process (pre-configured)	N/A	Used for the module secure boot process

During the lifecycle of the QASM, new keys and CSPs can be generated, removed, or replaced, according to the services supported by the QASM. Table 10 provides a list of these “transitional” keys and CSPs.

Table 10 - List of Transitional Keys and CSPs

Keys/CSPs	Type	Generation/Input	Output	Description
SO Public Key	ECDSA P-224, P-256 or P-384	Imported by the Owner to instantiate a SO	Output in plaintext	Public key used to validate signatures to authenticate the SO
AU Public Key	ECDSA P-224, P-256 or P-384	Imported by the SO to instantiate a user	Output in plaintext	Public key used to validate signatures to authenticate the AU. Note that there can be multiple AU public keys.
Symmetric Keys	See supported algorithms in Table 6	Generated or imported during normal operation	May be exported under wrap or MofN.	Secret keys imported and generated to support the services listed in Table 4.
Asymmetric Keys	See supported algorithms in Table 6	Generated or imported during normal operation	Public key exported in plaintext; private key may be exported under wrap.	Private/public key pairs imported and generated to support the services listed in Table 4.
Wrapping key	See supported algorithms in Table 6	Generated or imported during normal operation	If an RSA public key, can be exported in plaintext. If an AES key, can only be exported under wrap or MofN.	Wrapping key used to securely export other keys.

Keys/CSPs	Type	Generation/Input	Output	Description
Unwrapping key	See supported algorithms in Table 6	Generated or imported during normal operation	May be exported under wrap. Additionally, if the unwrapping key is an AES key, it can be exported under MofN.	Unwrapping key used to securely import other keys.
Wrapped key	Any private/secret key	Generated or imported during normal operation	Output encrypted with wrapping key	Any private/secret key that is marked as “extractable”.
Unwrapped key	Any private/secret key	Imported during normal operation	Output encrypted with wrapping key	Any private/secret key that was previously wrapped.
Exported key	AES key	Generated or imported during normal operation	Output exported using MofN (each split being encrypted).	A secret key that was marked as “extractable”.
Imported key	AES key	Imported during normal operation	Output exported using MofN (each split being encrypted).	A secret key that was previously exported using MofN.
ECC public keys	Se supported EC algorithms in Table 6	Generated or imported during normal operation	Output in plaintext	An ECC public key.
Public key of signature	See supported algorithms in Table 6	Generated or imported during normal operation	Output in plaintext	Public key used to verify digital signatures.
Private key of signature	See supported algorithms in Table 6	Generated or imported during normal operation	May be exported under wrap.	Private key used to generate a digital signature.
Authority	Single user or Quorum	Generated during normal operation	N/A	As described in Section 3.3.2, authorities can be used to provide additional authentication requirements on specific keys.

Keys/CSPs	Type	Generation/Input	Output	Description
Authority Public Key	ECDSA P-224, P-256 or P-384	Imported when creating an authority	Output in plaintext	Public key associated to an authority to validate its signatures.
Custodian Public Keys	RSA-4096	Imported in the QASM during the "MofN Export" service	Output in plaintext	Public keys used with the "MofN Export" service.
Data Objects	Any	Imported during normal operation	Output in plaintext	Data objects may be stored on the QASM. Note that those are not security relevant.

3.8 EMI/EMC

The cryptographic module conforms to the EMI and EMC requirements specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital Devices, Class B.

3.9 SELF TESTS

This section describes the self tests performed automatically by the module. If any of the power-up self-tests fail or any of the critical function tests, the HSM enters an error state and doesn't allow any cryptographic operation to be performed. The HSM behaves similarly for conditional tests except for the random number generator tests and for pairwise consistency. Indeed, because there are five independent entropy sources on the QASM, we allow for the failure of one before entering in an error state. As for pairwise key consistency, we return an error indicating that the algorithm failed, and we abort the operation being performed. Note that the data output interface is inhibited when performing self-tests.

3.9.1 Power-Up Self-Tests

Table 11 provides a list of the power-up self-tests performed on the QASM.

Table 11 - Power-Up Self-Tests

Name	Description
Cryptographic Algorithm Tests	
Known-Answer Tests	Each cryptographic algorithm implemented in the QASM has only one implementation. They are all tested using known-answer tests during power-up. Here is the list of tests: - AES primitive block encryption (blocks of 128, 192 and 256 bits) - AES primitive block decryption (blocks of 128, 192 and 256 bits)

Name	Description
	- AES-GCM encryption (128 bits key, 192 bits key and 256 bits key - AES-GCM decryption (128 bits key, 192 bits key and 256 bits key) - AES-Key wrap (128 bits KEK, 192 bits KEK and 256 bits KEK) - AES-Key unwrap (128 bits KEK, 192 bits KEK and 256 bits KEK) - AES-Key wrap with padding (128 bits KEK, 192 bits KEK and 256 bits KEK) - AES-Key unwrap with padding (128 bits KEK, 192 bits KEK and 256 bits KEK) - RSA-OAEP encryption - RSA-OAEP decryption - RSA signature generation (PSS and PKCS1.5) - RSA signature verification (PSS and PKCS1.5) - ECDH KAT (aka Primitive Z KAT) - NIST SP800-56C KDF - ANSI X9.63 KDF - ECDSA signature generation (P-224, P-256, P-384) - ECDSA signature verification (P-224, P-256, P-384) - Digest KAT (SHA1, SHA224, SHA256, SHA384, SHA512, SHA512_224, SHA512-256, SHA3_224, SHA3_256, SHA3_384, SHA3_512) - HMAC KAT (SHA1, SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384, SHA3_512)
Software/Firmware Tests	
Firmware Signature	When powering-up, the operational image's integrity must be validated before proceeding with the boot. Each image is signed with an RSA4096 key (unique to each QASM) which is itself signed by another RSA4096 to ensure that the image is not only valid, but also comes from Crypto4A. Furthermore, the operational image must be decrypted with a key unique to each QASM.
Random Number Generator Tests	
DRBG	Known-Answer Tests of the three main DRBG functions (instantiate, generate and reseed).
Entropy Sources	Each of the three entropy sources are tested during power-up to ensure they each provide a rich source of entropy to the DRBG. They must pass the tests described in NIST's SP800-90B.
Other Self-Tests	
FRAM Self-Test	Tests the drivers of the FRAM.

3.9.2 Conditional Self-Tests

Table 12 provides a list of the conditional self-tests performed on the QASM.

Table 12 - Conditional Self-Tests

Name	Description
Pairwise Consistency Tests	
Asymmetric key pairs	Each asymmetric key pair (RSA, ECC) are tested before being used, whether they will be used to sign/verify or encrypt/decrypt. For RSA keys, we validate each field of the key and then perform a raw pairwise test. For ECC keys, we validate the public key and perform a raw pairwise test.
Software/Firmware Load Tests	
Firmware Update	Firmware image that are loaded in the QASM are signed using an HSS key as well as an ECDSA key. They are also encrypted using AES256-GCM. Before accepting a new firmware image, the HSS and ECDSA signatures must be validated and it must be decrypted using a key loaded during the manufacturing process. Then, the QASM re-encrypts and re-signs the firmware image to its unique local key set.
Random Number Generator Tests	
Continuous Tests	Each entropy source is constantly tested using the continuous tests described in NIST's SP800-90B. These tests are meant to detect catastrophic failures.
Periodic Tests	In addition to continuous tests, the QASM periodically performs a full test of the quality of each entropy source following the full NIST's SP800-90B test suite. Similarly, the DRBG's output is also tested.
Other Self-Tests	
Key Integrity	Whenever a symmetric key is decrypted and about to be used, the module validates its integrity with a hashing function.

3.9.3 Critical Function Tests

Table 13 provides a list of the critical function self-tests performed on the QASM.

Table 13 – Critical Function Tests

Name	Description
DRBG	Known-Answer tests are performed to validate that the DRBG works properly. Furthermore, a critical error is returned if the instantiation of the DRBG fails. Here are the tests performed: - DRBG instantiate - DRBG generate - DRBG reseed
Physical Integrity	From the moment of its manufacturing until its decommission, the QASM constantly monitors its physical integrity. Unless a breach is detected, the QASM always perform a complete firmware decryption and validation during a power-up

	sequence. If the QASM booths up properly, one can assume its physical integrity has been validated.

3.10 MITIGATION OF OTHER ATTACKS

The QASM implements two post-quantum cryptographic algorithms, namely McEliece and HSS, both based on NIST's recommendations[11]. Some particularly sensitive operations such as firmware updates are handled from manufacturing to support composite signatures and composite encryption, thus providing a protection against eventual quantum attacks while simultaneously maintaining the requirements of a FIPS-Approved mode of operation. This follows NIST's recommended approach as described in their Frequently Asked Questions (FAQ[12]) under the following two questions:

1- Is it possible for a hybrid key-establishment mode to be performed in a FIPS 140 approved mode of operation? (added 1/28/20); and
2- Is it possible for dual signature generation or verification to be performed in a FIPS 140 approved mode of operation? (added 1/28/20).

[11] See SP800-208 and NIST's list of third round candidates for post-quantum cryptography standardization here: "https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions".
[12] See NIST's web page at: "https://csrc.nist.gov/Projects/post-quantum-cryptography/faqs"

4 SECURE OPERATION

During its manufacturing process, the QASM is tested extensively and loaded up with its own set of CSPs. Then, the QASM is loaded with its operational firmware encrypted specifically for itself and re-tested to ensure of its correct functionality. At this point, the QASM can be assembled in a QxEDGE or mated to a special docking station. The following description is presented on the basis of the QASM being integrated in a QxEDGE.

4.1 SETUP

When the QxEDGE is received, perform the following steps:

1. Place the QxEDGE at its target location (either rack mounted or vertical stand).
2. Connect the power cord(s) and the ethernet cables.
3. Power up the QxEDGE by pressing the front panel push button. This will power-up the QASM and also perform the power-up self test sequence automatically.
4. The Crytpo4A's Universal Configuration Application (UCA) may be used to configure the QxEDGE. Another method is to load configuration files from a server and installed them on the QxEDGE which will configure itself automatically.
5. At this point, the QxEDGE is completely functional and its internal QASM is available to provide security services to applications that require cryptographic operation (e.g., create key, generate a signature, import data for encryption).
6. The management application allows the administrator of the QxEDGE to monitor the status of the QASM through a web interface provided by the QxEDGE. More documentation is provided to the administrator from our customer support portal to better understand the management and monitoring tools.
7. To put the module into FIPS mode, please refer to section 4.4 of the current document.

4.2 USER GUIDANCE

Customers will be given access to our support portal to access comprehensive documentation related to the QASM. Nonetheless, we provide here a few quick tips to get started.

Customers can log on one of the SBCs located on the QxEDGE and interact with the QASM from a terminal window using one of the various tools provided. The main tool is referred to as "spa-key-man". To see all available commands, run `spa-key-man help`:

```
$ spa-key-man help
SPA Key Man

This utility is used to manage keys on a HSM (key management).
This application is used to create, export and import keys.

Command Syntax:
  spa-key-man <command>

For more information about a command:
  spa-key-man help <command-name>

      atp
```

```
ca
certificate
comm
data
dbsec
delete
entropy
export
find
graph
help
hsm
import
key
list
menu
mofn
monty
openssl
policy
quorum
recite
request
rng
set
show
sign
spore
ta
trust-anchor
user
verify
version
```

As mentioned in the displayed output of `spa-key-man help`, you can also isolate a specific command by running `spa-key-man help <command>` to obtain more information:

```
$ spa-key-man help rng

SPA Key Man - rng Command

Command Syntax:
  spa-key-man <options> rng get <count> [ hex | bin ] [ try-reseed | no-
reseed | must-reseed ] [ add-input XXXXXX ]
    Fetch random data from HSM

  spa-key-man <options> rng reseed [ add-input XXXXXX ]
    Re-seed the random bytes generator from entropy sources

Options:
  --config (file)
    File where the system's configuration can be found
```

```
--debug (flag)
  When specified, the command reports more information to help
  developers with debugging.

--error (flag)
  When specified, the command reports only error logs.

--info (flag)
  When specified, the command reports logs at info, warn and error
  levels.

--warn (flag)
  When specified, the command reports logs at warn and error levels.

--logDir (file)
  If specified, a copy of the logs is kept in this directory.

--trace (flag)
  When specified, the command reports some information to help
  developers with tracing.

--journal (flag)
  When specified, the command prints logs in a format suitable for
  journalctl.
```

We bring the reader's attention on two specific commands. The "recite" command can be used to provide various lists to help construct commands. For instance, `spa-key-man recite key-gen-mech` will provide the list of key generation mechanisms supported by the module. For more information, use `spa-key-man help recite`. The second command that may be useful to get started is `spa-key-man menu`, which provides an interactive menu to select commands to run.

4.3 MANAGEMENT

One of the QxEDGE's PEs may be used as the management interface for the operator. It is therefore possible to perform a variety of commands with the proper level of assurance to monitor and maintain the appliance. The available services may depend on the configuration of the QxEDGE. It is also possible to perform different tests and services from the other ports of the QxEDGE.

An essential command to manage a QASM is spa-key-man hsm info. This command extracts useful information from the QASM, including the serial number, the firmware version, its current state and its mode of operation (highlighted in yellow):

```
$ spa-key-man hsm info
HSM Info:
Time                          2020-11-29 22:29:22 UTC
Command Protocol              2.0
HSM Serial Number             SomeSerialNumber
```

```
Platform Name                   SomePlatformName
Platform Serial Number          SomePlatformSerialNumber
Platform Model                  QxEDGE 1.0
Platform Description            SomePlatformDescription
FIPS Mode                       enabled
HSM Type                        PRODUCTION
State                           OPERATIONAL
Error state                     NONE
UUID                            80000000-0000-0000-0000-000000000000
Firmware Version                2.1.0.0
Firmware Build Timestamp        Nov 28 2020 20:13:59 (OP)
HSM Temperature                 35
ATP 0                           2.1.0.0
ATP 1                           2.1.0.0
Monty                           2.1.0.0
HSM Uptime                      0 days 00:00:00
```

4.4 APPROVED MODE OF OPERATION

Switching to FIPS mode is done via an authorization process performed by the owner, as described in section 3.3.4. First, the owner creates an authorization request to switch the mode of operation. Then, the owner must sign the request and submit the signature to the QASM. Once the signature is validated, the QASM processes the request and switches the mode of operation. Note that all the keys are zeroized prior to changing mode (going from FIPS mode to Non-FIPS mode or vice versa) to avoid any compromise between FIPS mode and non-FIPS mode. Python scripts with comprehensive documentation are provided to facilitate running through these steps. To verify the mode in which the module is currently operating, please refer to section 4.3.

The QASM remains in its configured mode of operation for as long as the owner (aka the Crypto Officer) does not change the mode of operation. A re-boot sequence (i.e. a power-up sequence) does not allow the mode of operation to be change.

If the owner requests a mode change, the QASM, upon having verified the request will proceed to zeroize all of the user's objects and re-boot itself in the new mode of operation.

Note that switching to non-FIPS mode is done via the exact same procedure as described earlier in this section.