IBM® Security QRadar® Cryptographic Security Kernel Cryptographic Module

Software Version 7.2

FIPS 140-2

Non-Proprietary Security Policy

Policy Version 1.1

January 29, 2016

IBM Corporation
80 Bishop Dr. Unit B
Fredericton, NB
E3C 1B2
Canada

Table of Contents

List of Tables

List of Figures

1 Introduction

This document is the Security Policy for the IBM QRadar® Cryptographic Security Kernel library Version 7.2. This Security Policy specifies the security rules under which the module shall operate to meet the requirements of FIPS 140-2 Level 1. It describes how the module functions to meet the FIPS requirements, and the actions that operators must take to maintain the security of the module. The module is referred to in this document as the Cryptographic Security Kernel, the CSK, the cryptographic module, or the module.

This Security Policy describes the features and design of the IBM QRadar® Cryptographic Security Kernel library using the terminology contained in the FIPS 140-2 specification. *FIPS 140-2, Security Requirements for Cryptographic Modules* specifies the security requirements that will be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information. The NIST Cryptographic Module Validation Program (CMVP) validates cryptographic modules to FIPS 140-2 and other cryptography-based standards. Validated products are accepted by the Federal agencies of both the USA and Canada for the protection of sensitive or designated information.

The FIPS 140-2 standard and information on the CMVP can be found at http://csrc.nist.gov/groups/STM/cmvp.

This Security Policy contains only non-proprietary information. This document may be freely reproduced and distributed whole and intact. All other documentation submitted for FIPS 140-2 conformance testing and validation is “IBM - Proprietary” and is releasable only under appropriate non-disclosure agreements.

The IBM QRadar® Cryptographic Security Kernel library (the cryptographic module) meets the overall requirements applicable to Level 1 security for FIPS 140-2 as shown in Table 1.

Table 1: Cryptographic Module Security Requirements.

Security Requirements Section	***Level***
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles and Services and Authentication	2
Finite State Machine Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A
Cryptographic Module Security Policy	1

1.1 Acronyms and Abbreviations

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CMVP	Cryptographic Module Validation Program
CBC	Cipher-Block Chaining
CFB	Cipher Feedback
CSE	Communications Security Establishment
CSP	Critical Security Parameter
CTR	Counter Mode
DES	Data Encryption Standard
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
HMAC	Keyed-Hashing for Message Authentication
KAT	Known Answer Test
NIST	National Institute of Standards and Technology
OFB	Output Feedback
OS	Operating System
PUB	Publication
RAM	Random Access Memory
RSA	Rivest, Shamir and Adleman
SHA	Secure Hash Algorithm

2 IBM QRadar® Cryptographic Security Kernel Library

2.1 Functional Overview

The IBM QRadar® Cryptographic Security Kernel library (CSK) is used by the several product families within the Q1 Labs product line as the underlying cryptographic provider. A typical usage for the module is to provide the core cryptographic services necessary to implement the handshaking, establishment and management of TLS-secured connections between IBM appliances over WAN and LAN links. The module is distributed only as an integrated subcomponent of IBM appliances.

The CSK provides security functions for encryption, decryption, random number generation, hashing, getting the status of the integrity test, and running the self-tests. The library is used by the application.

2.2 Module Description

The CSK is provided as a shared library that runs on Linux operating systems. In FIPS 140-2 terminology, the CSK executes within a multi-chip standalone embodiment, such as a General Purpose Computer (GPC). The overall security level of the module is 1. The boundaries of the Cryptographic Security Kernel include the following components also depicted in Figure 1.

Logical Cryptographic Boundary

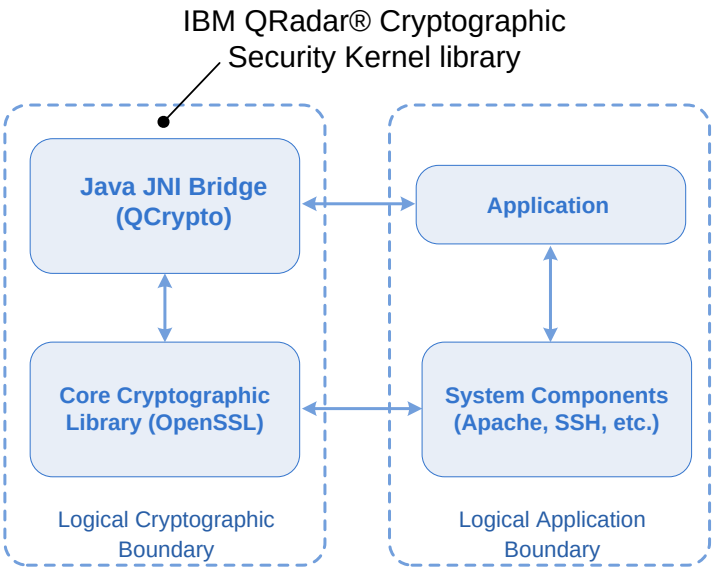
- Java JNI Bridge implemented by **libqcrypto_bridge.so**
- Core Cryptographic Library (OpenSSL) implemented by **libcrypto_bridge.so.10**

Logical Application Boundary

- Application
- System Components

The cryptographic module was validated on Red Hat Enterprise Linux (RHEL) 6.5

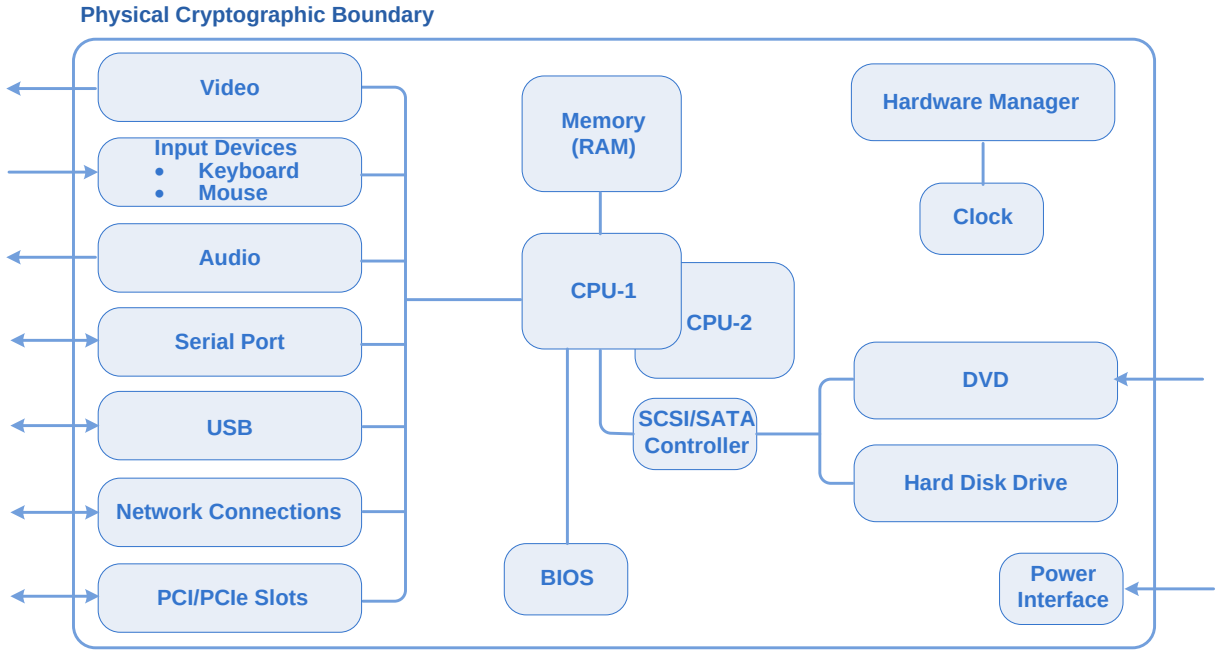
Figure 1 – FIPS 140-2 Logical Block Diagram



2.3 Module Interfaces

The module's interfaces are provided by the logical application programming interface (API), which provides the data input, data output, control input, and status output logical interfaces defined by FIPS 140-2. The module is installed on a GPC with physical ports consistent with that of a GPC as depicted in Figure 2.

Figure 2 – FIPS 140-2 GPC Block Diagram



The mapping of logical interfaces to the physical ports of the GPC is provided in Table 2 below.

Table 2: FIPS 140-2 Logical Interface Mappings.

Logical Interface	Physical Ports	Interface Description
Data input	Network1, Network2, DVD, SCSI/SATA Controller, Serial, USB, Keyboard, Mouse, Host Bus Adapter	Data entering the module using these interfaces.
Data output	Network1, Network2, SCSI/SATA Controller, Serial, USB, Graphics Controller, Host Bus Adapter	. Data exiting the module using these interfaces.
Control input	Network1, Network2, Host Bus Adapter, DVD, Serial, USB	Control parameters entering the module using these interfaces.
Status output	Network1, Network2, Host Bus Adapter, Audio, Graphics Controller	Return values from module operations including error messages.
Power	Power Interface	N/A

2.4 Roles and Services

The Cryptographic Security Kernel (CSK) is a software only module that provides an API for applications to perform general cryptography. The module supports authentication for both the Crypto-Officer and Admin Role. The module supports two roles in the module that operators may assume: a Crypto-Officer role and a User role (named Admin in this module), which are explicitly assumed. The module does not support concurrent operators.

2.4.1 Crypto-Officer Role

The Crypto-Officer role has the ability to install the module, to query the module for status information, and to force the module to perform startup self-tests.

Please note that the keys and critical security parameters (CSPs) listed in the table indicate the type of access required using the following notation:

- R – Read: The CSP is read.
- W – Write: The CSP is established, generated, modified, or zeroized.
- X – Execute: The CSP is used within a FIPS-Approved or Allowed security function or authentication mechanism.

Table 3: Crypto-Officer Services

Service	Description	Input	Output	CSP and Type of Access
Installing the module	Installation tasks include loading the software components onto the system and configuring the module to ensure proper operation.	Pathname	Status	None
Commit	Apply any changes made to a system file of your FIPS enabled system.	commit	Command Response	None
Deploy	Start a full deploy on the appliance. Restarts all services.	deploy	Command Response and Status Output	None
Disable FIPS configure-ation	Takes the module out of a configured state by allowing root access.	disable_ verified_mode	Command Response and Status Output	Advanced Encryption Standard (AES) - W Triple Data Encryption Standard (Triple-DES) – W RSA public/private keys – W Diffie-Hellman (DH) – W (keyed) Hash Message Authentication Code (HMAC) – W
Display status	Displays the status of the operating system, required RPM files, log settings, and FIPS mode in the command line.	fips_self_check	Status Output	AES – W Triple-DES – W RSA public/private keys – W DH – W HMAC – W
Get logs	Collects system data for your FIPS appliance.	get_logs	Command response	None
Modify log source	Modifies log sources by using the command-line interface of a FIPS enabled appliance.	mod_log4j	Command response	None

Service	Description	Input	Output	CSP and Type of Access
Reboot	Restarts a FIPS enabled appliance.	reboot	Command response and status output	AES – W Triple-DES – W RSA public/private keys – W DH – W HMAC – W
Start, stop, or restart a service	Changes the status of a service on your QRadar appliance. For a list of services that can be restarted by the crypto user, type service --list.	service <service name> <start | stop | restart>	Command response and status output	AES – W Triple-DES – W
Perform self-test	Trigger a power-on self test.	Module reboot	Command response and status output	AES – W Triple-DES – W RSA public/private keys – W DH – W HMAC – W
Zeroize key	Zeroizes and de-allocates memory containing sensitive data	API call parameters	Status	AES key – W Triple DES key – W HMAC key – W RSA private/public key – W DH components – W

2.4.2 User Role

The User role has the ability to perform basic cryptographic operations. Descriptions of the services available to the User role are provided in Table 4 below.

Table 4: User Services

Service	Description	Input	Output	CSP and Type of Access
Commit	Apply any changes made to a system file of your FIPS enabled system.	commit	Command Response	None
Deploy	Start a full deploy on the appliance. Restarts all services	deploy	Command Response and Status Output	None

Get logs	Collects system data for your FIPS appliance.	get_logs	Command Response	None
Modify log source	Modifies log sources by using the command-line interface of a FIPS enabled appliance.	mod_log4j	Command Response	None
Reboot	Restarts a FIPS enabled appliance.	reboot	Command Response and Status Output	AES – W Triple-DES – W RSA public/private keys – W DH – W HMAC – W
Generate random bits (SP-800-90A)	Returns the specified number of random bits to calling application	API call parameters	Status, random bits	SP-800-90A DRBG seed-RWX DRBG V Value, RX
Generate keyed hash (HMAC)	Compute and return a message authentication code using HMAC-SHA1, HMAC-SHA256, HMAC-SHA512	API call parameters, key, message	Status, hash	HMAC key – RX
Zeroize key	Zeroizes and de-allocates memory containing sensitive data	API call parameters	Status	AES key – W Triple DES key – W HMAC key – W RSA private/public key – W DH components – W
Symmetric encryption	Encrypt plaintext using supplied key and algorithm specification (Triple-DES ECB/CBC/CFB /OFB or AES ECB /CBC/CFB/CTR/OFB)	API call parameters, key, plaintext	Status, ciphertext	AES key – RX Triple-DES key – RX
Symmetric decryption	Decrypt ciphertext using supplied key and algorithm specification (Triple-DES ECB/CBC/CFB /OFB or AES ECB /CBC/CFB/CTR/OFB)	API call parameters, key, ciphertext	Status, plaintext	AES key – RX Triple-DES key – RX
Generate asymmetric key pair	Generate and return an RSA asymmetric key pair	API call parameters	Status, key pair	RSA private/public key – W
RSA encryption	Encrypt plaintext using RSA public key (used for key transport)	API call parameters, key, plaintext	Status, ciphertext	RSA public key – RX
RSA decryption	Decrypt ciphertext using RSA private key (used for key transport)	API call parameters, key, ciphertext	Status, plaintext	RSA private key – RX

Diffie-Hellman key agreement	Perform key agreement using Diffie-Hellman algorithm	API call parameter	Status, key components	DH components – W
Signature Generation	Generate a signature for the supplied message using the specified key and algorithm (RSA)	API call parameters, key, message	Status, signature	RSA private key – RX
Signature Verification	Verify the signature on the supplied message using the specified key and algorithm (RSA)	API call parameters, key, signature, message	Status	RSA public key – RX

2.5 Operator Authentication

The cryptographic module supports role-based authentication for access to user services and crypto officer services, explicitly authenticating users and crypto officers.

An operator attempting to access user services must enter a password that is known to the module.

A crypto officer sets the password during the initial enabling of FIPS mode. A crypto officer may change the password by logging in as the root user and setting a new password. The password change is completed when the module is in the configured state.

Strength of Authentication Mechanism

CO and FIPS Admin passwords are required to be at least 6 characters long. The maximum password length is 64 characters. Case-sensitive alphanumeric characters and at least one special character (‘$’, ‘.’, ‘,’, ‘!’, ‘%’, ‘^’, ‘*’) must be used, which gives a total of 69 characters to choose from. The chance of a random attempt falsely succeeding is 1:69^6, or 1:107,918,163,081.

When an incorrect password is entered locally or remotely the module injects a 2 second delay before issuing the failure and allowing another attempt. Remote connections disconnect after 6 consecutive failed attempts.

This limits local password attempts to no more than 30 in a one-minute period (30/69^6) with a probability of far less than one in 100,000 that a random attempt will succeed or a false acceptance will occur.

2.6 Physical Security

The Cryptographic Security Kernel is a software only module, which operates on a multi-chip standalone device, such as a GPC. As such, it does not include physical security mechanisms and the FIPS 140-2 requirements for physical security are not applicable.

2.7 Operational Environment

The module was validated with FIPS 140-2 requirements on each of the following operating system platforms:

- Red Hat Enterprise Linux (RHEL) 6.5.

2.8 Cryptographic Key Management

The module implements the FIPS-Approved algorithms listed in Table 5 below and uses these algorithms in FIPS 140-2 Approved mode.

Table 5 – FIPS-Approved Algorithm Implementations

Algorithm	Certificate Number
AES 128/192/256 in ECB/CBC/CFB/CTR/OFB modes	3131
Triple-DES 128/192 in ECB/CBC/CFB/OFB modes	1794
RSA (X9.31, PSS, PKCS#1) for signing, signature generation, and verification – 2048- and 3072-bit	1686
SHA-1, SHA-256, SHA-512	2600
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512	1981
SP-800-90A Deterministic Random Bit Generator (DRBG) CTR-DRBG AES-256	753

TLS(TLS1.0/1.1) SHA Val#2600 HMAC Val#1981 SSH (SHA 1 , 256) SHA Val#2600	397

The operating system must be configured for single user mode for FIPS 140-2 compliance (see section 3 for guidance).

All cryptographic keys and CSPs are under the control of the OS or calling applications, which is responsible for protection of the CSPs against unauthorized disclosure, modification, and substitution. The module only allows access to CSPs through its well-defined APIs. The module performs a Software Integrity Test using the HMAC-SHA-256 algorithm.

The module utilizes the following non FIPS-Approved algorithm implementations:

- Diffie-Hellman 2048 bit key size (key agreement, key establishment methodology provides between 112 and 202 bits of encryption strength).
- HMAC MD5 and MD5 within TLS only
- Non-deterministic random number generators for seeding the SP-800-90A DRBG.
    - The minimum number of bits of entropy requested per each GET function is 256 bits.
    - The NDRNG is outside the logical boundary but still within the physical boundary.
- RSA (key wrapping; key establishment methodology provides between 112 and 256 bits of encryption strength)

The TLS and SSH protocols have not been reviewed or tested by the CAVP and CMVP. Please see NIST document SP800-131A for guidance regarding the use of non FIPS-approved algorithms.

The module supports the critical security parameters (CSPs) listed in Table 6.

Table 6 – List of Cryptographic Keys, Cryptographic Key Components, and CSPs

CSP (Key Type)	Generation / Input	Output	Storage	Zeroization	Use
Crypto-Officer Password, FIPS Admin Password (Passphrase of at least six characters)	Entered by a CO or FIPS Admin locally.	Never	Stored in encrypted format	Zeroized when the password is updated with a new password	Used for authenticating all COs and FIPS Admin using CLI[1]
AES Keys (AES 128, 192, 256 bit keys)	API call parameter	Never	Plaintext in volatile memory	By API call, power cycle, host reboot.	Keys are passed as arguments to the module API for cryptographic processing.
Triple-DES Keys (128, 192 bit key)	API call parameter	Never	Plaintext in volatile memory	By API call, power cycle, host reboot.	Keys are passed as arguments to the module API for cryptographic processing.
RSA private key (RSA 2048, 3072 bit key)	API call parameter	Never	Plaintext in volatile memory	By API call, power cycle, host reboot	Signature generation, decryption
	Internally generated	API call parameter			Used by host application
RSA Public Key (RSA 1024[2], 2048, 3072 bit key)	API call parameter	Never	Plaintext in volatile memory	By API call, power cycle, host reboot	Signature verification, encryption
	Internally generated	API call parameter			Used by host application

[1] CLI – Command Line Interface

[2] RSA 1024 bit keys allowed for legacy use only for signature verification purposes.

CSP (Key Type)	Generation / Input	Output	Storage	Zeroization	Use
DH Public Components (Public components of DH protocol)	Internally generated	API call parameter	Plaintext in volatile memory	By API call, power cycle, host reboot	Used by host application
DH Private Components (Private components of DH protocol)	Internally generated	API call parameter	Plaintext in volatile memory	By API call, power cycle, host reboot	Used by host application
DRBG Seed Key (AES 256 bit key)	Imported from dev/urandom	Never	Plaintext in volatile memory	By API call, power cycle, host reboot	Generate random number
NDRNG Seed Value (128 bit random value)	Imported from dev/random	Never	Plaintext in volatile memory	By API call, power cycle, host reboot	Generate random number
HMAC Keys (HMAC keys SHA-1, SHA-256, SHA-512)	API call parameter	Never	Plaintext in volatile memory	By API call, power cycle, host reboot	Message Authentication
Software Integrity Check Key (HMAC-SHA-256 key)	Never	Never	On host system as plaintext file	By uninstalling the module	Used to perform the software integrity test at power-on.

2.8.1 Key Generation

The module uses an SP 800-90A DRBG implementation to generate cryptographic keys. This DRBG is FIPS-Approved as shown in Annex C to FIPS PUB 140-2.

2.8.2 Key Entry and Output

The cryptographic module itself does not support key entry or key output from its physical boundary. However, keys are passed to the module as parameters from the applications resident on the host platform via the exposed APIs. Similarly, keys and CSPs exit the module in plaintext via the well-defined exported APIs.

2.8.3 Key/CSP Storage and Zeroization

Symmetric, asymmetric, and HMAC keys are either provided by or delivered to the calling process, and are subsequently destroyed by the module at the completion of the API call. Keys and CSPs stored in RAM can be zeroized by a power cycle or a host system reboot. The SP 800-90A DRBG seed is initialized by the module at power-up and remain stored in RAM until the module is uninitialized by a host system reboot or power cycle. The HMAC key that is used to verify the integrity of the module is stored in a file residing on the host system.

2.9 EMI/EMC

The Cryptographic Security Kernel is a software module. Therefore, the only electromagnetic interference produced is that of the host platform on which the module resides and executes. FIPS 140-2 requires that the host systems on which FIPS 140-2 testing is performed meet the Federal Communications Commission (FCC) EMI and EMC requirements for business use as defined in Subpart B, Class A of FCC 47 Code of Federal Regulations Part 15. However, all systems sold in the United States must meet these applicable FCC requirements.

2.10 Self-Tests

This section describes the power-up and conditional self-tests performed by the module. If any of the tests listed below fails to complete successfully, the module enters into a critical error state where all cryptographic operations and output of any data is prohibited. An error message is logged for the CO to review and requires action on the CO's part to clear the error state.

2.10.1 Power-Up Self-Tests

At module power-up, the Cryptographic Security Kernel performs power-on self-tests without requiring any involvement from the operator using a default entry point mechanism.

The module employs a DEP-like initialization mechanism as described in IG 9.10 - Note 7). Support for this is provided on 2 levels: kernel and the module.

The kernel contains a special FIPS parameter that once enabled, runs additional integrity check on the kernel image to make sure it was not modified.

In addition, the parameter also creates a read-only flag within the proc file system in **/proc/sys/crypto/fips_enabled**.

Prior to initial use, the module is initialized. During the initialization, the fips_enabled flag is checked and if present, the module forces itself to perform all necessary start-up tests.

Furthermore, the system's kernel is statically configured to always be in FIPS mode. It cannot be taken out of that mode without significant changes.

The following self-tests are performed at power-up:

- Software integrity checks (HMAC SHA-256) over each component of the module.

- Known Answer Tests (KATs):
  - AES Decrypt
  - Triple-DES Decrypt
  - AES Encrypt
  - Triple-DES Encrypt
  - RSA. The following implementations are tested:
    - X9.31 Signature Generation, Signature Verification
    - PKCS#1 1.5 Signature Generation
    - PKCS#1 PSS Signature Generation, Signature Verification
  - SHA-1, SHA-256, SHA-512
  - HMAC SHA-1, HMAC SHA-256, HMAC SHA-512
  - SP 800-90A DRBG

2.10.2 Conditional Self-Tests

The Cryptographic Security Kernel performs the following conditional self-tests:

- Continuous DRBG Test
- Continuous NDRNG test for the non-approved NDRNG.
- RSA Pairwise Consistency Check (SIG (gen), SIG (ver), encrypt, decrypt). Implementations tested are PSS, PKCS1, X9.31.

2.11 Design Assurance

Source code and documentation are both managed and stored within Perforce, an automated configuration management system, and its associated server.

2.12 Mitigation of Other Attacks

This section is not applicable. The module does not claim to mitigate any attacks beyond the FIPS 140-2 Level 1 requirements for this validation.

3 Secure Operation

The Cryptographic Security Kernel meets Level 1 requirements for FIPS 140-2. The sections below describe how to place and keep the module in FIPS-Approved mode of operation.

The Crypto-Officer role is responsible for installing the module as part of its host application. The cryptographic module implements a software integrity test that consists of an HMAC-SHA-256 computed over each of the libraries. During the power-up self-tests phase, the signatures are verified over the stored CSK Module instance. If the stored signatures are verified, then the test is passed. Otherwise, the test is failed and the module enters an error state where no cryptographic functionality is allowed.

3.1 Initial Setup

When initialized and configured according to the Crypto-Officer guidance in this Security Policy, the module does not support a non-Approved mode of operation.

The module is installed when the IBM Security QRadar appliance is installed. After installing the appliance you must perform the following procedures:

1. Enable FIPS mode.
2. Disable automatic updates.

For procedure details, please see the *IBM Security QRadar FIPS Installation Guide*.

3.2 Secure Management

This section provides guidance which ensures that the module is always operated in a secure configuration.

3.2.1 Initialization

FIPS 140-2 mandates that a software cryptographic module at Security Level 1 shall be restricted to a single operator mode of operation. Prior to installing the module, the Crypto-Officer must ensure that the Linux operating system environment is in single-user mode.

3.2.2 Management

The Crypto-Officer should monitor the module's status regularly and make sure only the services listed in Table 3 and Table 4 are being used. If any irregular activity is noticed or the module is consistently reporting errors, then IBM customer support should be contacted.

3.2.3 Zeroization

The module does not provide for persistent storage of cryptographic keys. The calling applications are responsible for key management, protection, and zeroization. As a result, zeroization is not required by the module.

3.3 User Guidance

Only the module's cryptographic functionalities are available to the User. Users are responsible to use only the services that are listed in Table 4. Although the User does not have any ability to modify the configuration of the module, they should report to the Crypto-Officer if any irregular activity is noticed.

The User must not modify the configuration of the module as established by the Crypto-Officer.

4 References

The IBM website www.ibm.com contains information on the full line of solutions from IBM.

The following National Institute of Standards and Technology publications are available at URL csrc.nist.gov/groups/STM/cmvp/index.html:

- *FIPS PUB 140-2: Security Requirements for Cryptographic Modules*
- *FIPS 140-2 Annex A: Approved Security Functions*
- *FIPS 140-2 Annex B: Approved Protection Profiles*
- *FIPS 140-2 Annex C: Approved Random Number Generators*
- *FIPS 140-2 Annex D: Approved Key Establishment Techniques*
- *Derived Test Requirements (DTR) for FIPS PUB 140-2, Security Requirements for Cryptographic Modules* (a joint publication of the National Institute of Standards and Technology and Communications Security Establishment).
- *Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197
- *Secure Hash Standard (SHS)*, Federal Information Processing Standards Publication 180-3