Hewlett Packard Enterprise

Hewlett Packard Enterprise

HP-UX Kernel Cryptographic Module

Software Version: 1.0

FIPS 140-2 Non-Proprietary Security Policy

FIPS Security Level: 1
Document Version: 1.3

Hewlett Packard Enterprise Development LP

USA Address:
Hewlett Packard Enterprise Company
6280 America Center Drive,
San Jose, CA 95002

India Address:
Hewlett Packard India Software Operations Private Ltd
Survey #192 , Whitefield Road
Mahadevapura Post
Bangalore 560048

Email: info@hpe.com
https://www.hpe.com/

Table of Contents

Table of Figures

List of Tables

1 Introduction

1.1 Purpose

This is a non-proprietary Cryptographic Module Security Policy for the HP-UX Kernel Cryptographic Module (Software Version: 1.0) from Hewlett Packard Enterprise Development LP. This Security Policy describes how the HP-UX Kernel Cryptographic Module meets the security requirements of Federal Information Processing Standards (FIPS) Publication 140-2, which details the U.S. and Canadian Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the National Institute of Standards and Technology (NIST) and the Communications Security Establishment Canada (CSEC) Cryptographic Module Validation Program (CMVP) website at https://csrc.nist.gov/projects/cryptographic-module-validation-program.

This document also describes how to run the module in a secure FIPS-Approved mode of operation. This policy was prepared as part of the Level 1 FIPS 140-2 validation of the module. The HP-UX Kernel Cryptographic Module is referred to in this document as the KCM, the crypto-module, or the module.

1.2 References

This document deals only with operations and capabilities of the module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the module from the following sources:

- The HPE website (https://www.hpe.com/) contains information on the full line of products from HPE.
- The CMVP website (https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules) contains ceritificate for this module having contact information for any quereies

1.3 Document Organization

The Security Policy document is one document in a FIPS 140-2 Submission Package. In addition to this document, the Submission Package contains:

- Vendor Evidence document
- Finite State Model document
- Other supporting documentation as additional references

Under contract to HPE, Leidos worked with HPE on the development of this Security Policy. With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Submission Package is proprietary to HPE and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact HPE.

2 HP-UX KCM

2.1 Overview

The HP-UX Kernel Cryptographic Module (also called "KCM") is an extraction and integration of the cryptographic functionality of various kernel-mode HP-UX software programs into a single kernel library. The various kernel-mode HP-UX software applications that utilize the cryptographic services of the module are:

- **Encrypted Volume and File System (EVFS)** – EVFS is a volume and file-level encryption utility developed by HPE and designed to run in HP-UX's kernel space. EVFS relies on AES[1] with 128, 192, and 256-bit keys in both CBC[2] mode to encrypt files on disk or entire logical disk volumes. When encrypted files or volumes are requested of EVFS by a user, the user establishes a secure session with the software and utilizes the cryptographic functionalities.

- **Whitelisting Infrastructure (WLI)** – WLI is HPE's file-level whitelisting infrastructure. This application protects files based on administrator established policy, providing file-level integrity checking and some level of user-granular access control to the files. Policies are user-specific, as each user is issued a unique RSA[3] key pair which is identified by WLI using the user's Crypto Identifier, a block of meta data created using the public key of the user.

- **IPsec[4]** – HPE's implementation of IPsec in HP-UX is proprietary and based on several modular components that execute in both user space and kernel space. User space modules include a Policy daemon, IKE[5] daemon, and a Command and Utilities module. Kernel space modules include the IPsec core within the XPORT[6] kernel of the TCP/IP[7] stack, and the IPsec ISU[8] kernel.

2.1.1 KCM Architecture Overview

HPE KCM is an integration of algorithms and functionalities used by EVFS, WLI, and IPsec into a single library. The library is written in C and exports an API[9] within which a unique call exists for each cipher-specific operation. Applications requiring KCM services invoke those services using the module's exported API.

The KCM is validated at the following FIPS 140-2 Section levels listed in Table 1 below.

Table 1 – Security Level Per FIPS 140-2 Section

Section	Section Title	Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services, and Authentication	1
4	Finite State Model	1

[1] AES – Advanced Encryption Standard
[2] CBC – Cipher-Block Chaining
[3] RSA – Rivest Shamir Adleman
[4] IPsec – Internet Protocol Security
[5] IKE – Internet Key Exchange
[6] XPORT – Transport
[7] TCP/IP – Transmission Control Protocol/Internet Protocol
[8] ISC – Independent Software Unit
[9] API – Application Programming Interface

Section	Section Title	Level
5	Physical Security	N/A[10]
6	Operational Environment	1
7	Cryptographic Key Management	1
8	EMI/EMC[11]	1
9	Self-tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A

2.2 Module Specification

The HP-UX Kernel Cryptographic Module is a software module (Software Version: 1.0) with a multi-chip standalone embodiment. The overall security level of the module is 1. The cryptographic boundaries of the module consists of the KCM library as shown by the red-colored dotted line in Figure 1 (for the physical boundary) and Figure 2 (for the logical boundary). The module was tested and found compliant in the following operational environments:

- HP-UX 11i v3 running on an HPE Integrity BL860c i2 server blade with an Intel Itanium Processor 9350
- HP-UX 11i v3 running on an HPE Integrity BL890c i6 server blade with an Intel Itanium Processor 9740
- HP-UX 11i v3 on HPE Portable HP-UX on Red Hat Enterprise Linux Server 7.5 (x86-64) running on an HPE ProLiant DL580 Gen10 with an Intel Xeon Gold 6146

Security functions offered by the module (and their associated algorithm implementation certificate numbers) are listed in Table 2 below.

Table 2 – FIPS-Approved Algorithm Implementations

Algorithm	Cert. Number
Symmetric Key Algorithm	
FIPS 197 AES encryption and decryption in CBC mode (128/192/256 bits) as defined in SP 800-38A	2488
Asymmetric Key Algorithm	
FIPS 186-4 RSA key generation and PKCS#1 v1.5 signature generation/verification (2048-bit) with SHA-256, SHA-384, SHA-512	1277
Secure Hashing Standard (SHS)	
FIPS 180-4 SHA[12]-256, SHA-384, SHA-512	2106
Message Authentication Code (MAC) Function	
FIPS 198-1 HMAC[13] using SHA-256, SHA-384, and SHA-512	1530
Random Bit Generation (RBG)	

[10] N/A – Not Applicable
[11] EMI/EMC – Electromagnetic Interference / Electromagnetic Compatibility
[12] SHA – Secure Hash Algorithm
[13] HMAC – (Keyed-) Hash Message Authentication Code

Algorithm	Cert. Number
SP[14] 800-90A Hash-based DRBG[15]	346
Cryptographic Key Generation (CKG)	
SP 800-133 Cryptographic Key Generation for Symmetric Keys and Asymmetric Keys	Vendor Affirmed

The module also implements the following non-Approved but allowed algorithm:

- RSA[16] (key wrapping; key establishment methodology provides 112 bits of encryption strength)

2.2.1 Physical Cryptographic Boundary

As a software cryptographic module, there are no physical protection mechanisms implemented. Therefore, the module must rely on the physical characteristics of the host system. The physical boundary of the cryptographic module is defined by the hard enclosure around the host platform on which it executes. The module supports the physical interfaces of host system. These interfaces include the integrated circuits of the system board, processor, network adapters, RAM[17], hard disk, device case, power supply, and fans. See Figure 1 for a generic block diagram of the test systems.

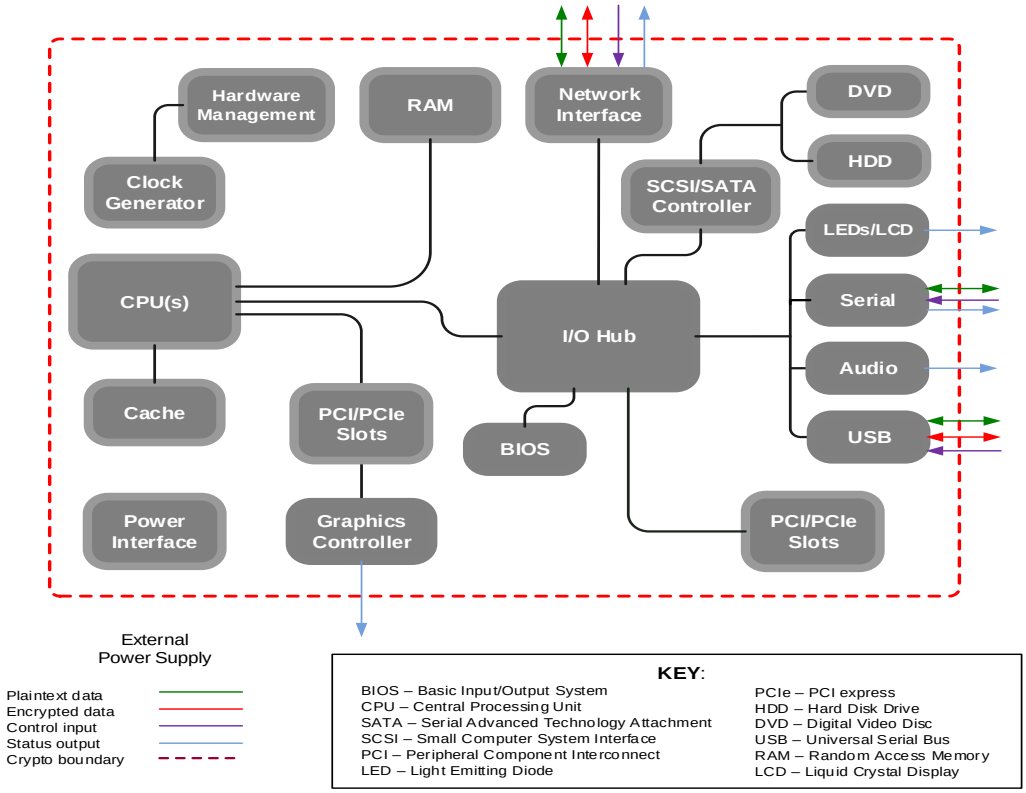

Figure 1 – Test Platform Block Diagram

[14] SP – Special Publication
[15] DRBG – Deterministic Random Bit Generator
[16] RSA key wrapping (which uses PKCS#1 v1.5 padding) is allowed for use in the Approved mode of operation[16] through 2023 per SP 800-131A Rev. 2. After 2023 RSA key wrapping is disallowed. The calling application may use this to implement a key transport scheme, which is allowed for use in FIPS mode.
[17] RAM – Random Access Memory

2.2.2 Logical Cryptographic Boundary

Figure 2 shows a logical block diagram of the module, where "Calling Application" represents any other software/firmware component loaded on the host platform that employs the module's services. The module's logical cryptographic boundary (also illustrated in Figure 2) encompasses all functionality provided by the module as described in this document.

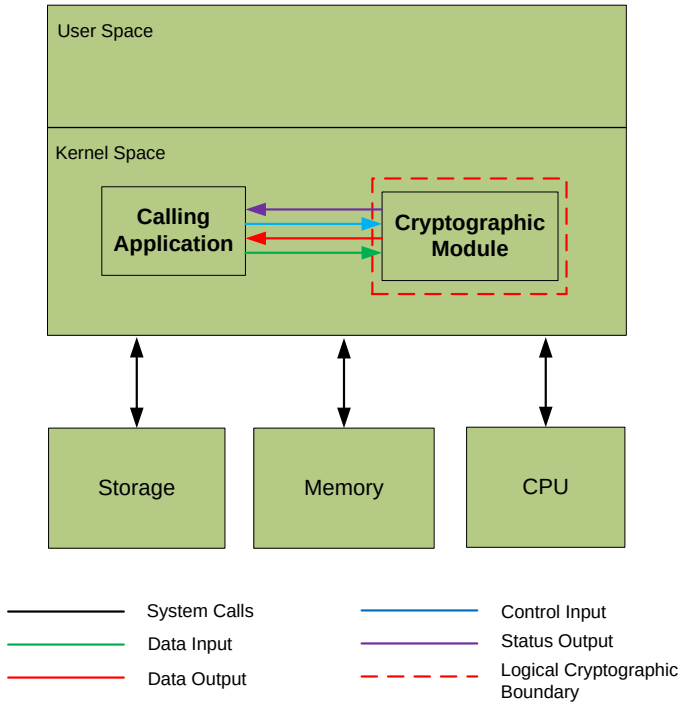

Figure 2 – HP-UX Kernel Cryptographic Module Logical Cryptographic Boundary

The cryptographic module is a shared object that provides cryptographic and secure communication services to the various HPE-developed kernel-space applications. In this document, those applications will be referred to collectively as the "calling application". The module is used by the calling application to provide symmetric key encryption/decryption, digital signature generation and verification, hashing, assymmetric keypair generation, random bit generation, and message authentication functions.

2.3 Module Interfaces

The module's physical ports can be categorized into the following logical interfaces defined by FIPS 140-2:

- Data input
- Data output
- Control input
- Status output

As a software module, the module has no physical characteristics. The module's physical and electrical characteristics, manual controls, and physical indicators are those of the host system. The mapping of the module's logical interfaces in the software to FIPS 140-2 logical interfaces is described in Table 3 below.

Table 3 – FIPS 140-2 Logical Interface Mappings

FIPS 140-2 Interface	Physical Interface	Module Interface (API)
Data Input	Network/Serial/USB port, DVD/CD, Keyboard port, and Mouse port	Function calls that accept, as their arguments, data to be used or processed by the module
Data Output	Network/Serial/USB port, DVD/CD, Graphics/Video port, and Audio	Arguments for a function that specify where the result of the function is stored
Control Input	Network/Serial/USB port, Keyboard port and Mouse port, Power button	Function calls utilized to initiate the module and the function calls used to control the operation of the module
Status Output	Network/Serial/USB port, Graphics/Video, LED indicators, and Audio	Return values for function calls; module-generated error messages
Power Input	Power plug/adapter, Power Switch	Not Applicable

2.4 Roles and Services

There are two roles in the module that operators may assume: a Crypto Officer role and User role. The Crypto Officer is responsible for managing the module and monitoring the module's status, while the User accesses the services implemented by the module. The available functions are utilized to provide or perform the cryptographic services.

The various services offered by the module are described in Table 4 and Table 5. The Critical Security Parameters (CSPs) used by each service are also listed. Please note that the keys and CSPs listed in the tables use the following notation to indicate the type of access required:

- R – Read: The keys and CSPs are read.
- W – Write: The keys and CSPs are established, generated, modified, or zeroized.
- X – Execute: The keys and CSPs are used within an Approved or Allowed security function or authentication mechanism.

2.4.1 Crypto Officer Role

The Crypto Officer (CO) role is responsible for initializing module, zeroizing keys and CSPs, executing self-tests, and monitoring status. Descriptions of the services available to the Crypto Officer role are provided in Table 4.

Table 4 – Crypto Officer Services

Service	Description	Input	Output	CSP and Type of Access
Initialize module	Performs integrity check and power-up self-tests	API call parameters	Status	None
Show status	Returns the current mode of the module	API call	Status	None
Run self-tests on demand	Performs power-up self-tests	None	Status	None

Service	Description	Input	Output	CSP and Type of Access
Zeroize keys	Zeroizes and de-allocates memory containing sensitive data	API call, reboot command or cycling power	None	All keys and CSPs – W

2.4.2 User Role

The User role can utilize the module's cryptographic functionalities. Descriptions of the services available to the User role are provided in Table 5.

Table 5 – User Services

Service	Description	Input	Output	CSP and Type of Access
Random number generation (Hash_DRBG)	Returns the specified number of random bits to calling application	API call parameters	Status, random bits	DRBG seed – RWX Hash_DRBG V value – RWX Hash_DRBG C value – RWX
Message digest generation (SHS)	Compute and return a message digest using SHS algorithms	API call parameters, message	Status, hash	None
Keyed hash (HMAC) generation	Compute and return a message authentication code using HMAC-SHA	API call parameters, key, message	Status, hash	HMAC key – RX
Symmetric encryption	Encrypt plaintext using supplied key and algorithm specification (AES)	API call parameters, key, plaintext	Status, ciphertext	AES key – RX
Symmetric decryption	Decrypt ciphertext using supplied key and algorithm specification (AES)	API call parameters, key, ciphertext	Status, plaintext	AES key – RX
Symmetric key generation	Generate and return the specified type of symmetric key (AES)	API call parameters	Status, key	AES key – W
Asymmetric key pair generation	Generate and return the specified type of asymmetric key pair (RSA)	API call parameters	Status, key pair	RSA private/public key – W
RSA key wrapping	Perform key wrapping using RSA public key (used for key transport)	API call parameters, key, plaintext	Status, ciphertext	RSA public key – RX
RSA key unwrapping	Perform key unwrapping using RSA private key (used for key transport)	API call parameters, key, ciphertext	Status, plaintext	RSA private key – RX

Service	Description	Input	Output	CSP and Type of Access
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

2.4.3 Authentication

The module does not support any authentication mechanism. Operators of the module implicitly assume a role based on the service of the module being invoked. Since all services offered by the module can only be used by either the Crypto Officer or the User, the roles are mutually exclusive. Thus, when the operator invokes a Crypto Officer role service, he implicitly assumes the Crypto Officer role. When the operator invokes a User role service, he implicitly assumes the User role.

2.5 Physical Security

Since this is a software module, the module relies on the target system to provide the mechanisms necessary to meet FIPS 140-2 physical security requirements. The module was tested and found compliant on the systems identified in the section Module Specification. All components of the platforms are made of production-grade materials, and all integrated circuits coated with commercial standard passivation.

2.6 Operational Environment

The KCM was tested and found compliant with the applicable FIPS 140-2 requirements when running on the following operational environment(s):

- HP-UX 11i v3 running on an HPE Integrity BL860c i2 server blade with an Intel Itanium Processor 9350
- HP-UX 11i v3 running on an HPE Integrity BL890c i6 server blade with an Intel Itanium Processor 9740
- HP-UX 11i v3 on HPE Portable HP-UX on Red Hat Enterprise Linux Server 7.5 (x86-64) running on an HPE ProLiant DL580 Gen10 with an Intel Xeon Gold 6146

All cryptographic keys and CSPs are under the control of the operating system (OS), which protects the keys and CSPs against unauthorized disclosure, modification, and substitution. The module only allows access to keys and CSPs through its well-defined APIs. The module performs a Software Integrity Test using an Approved RSA 2048 w/ SHA-256 digital signature verification.

The vendor also affirms under the guidance of FIPS 140-2 IG G.5 that the module executes in its FIPS-Approved manner on other platforms equipped with an Intel Itanium Processor running the HP-UX 11i v3 operating system or platforms equipped with an Intel Xeon Processor running the HPE Portable HP-UX product. No claim can be made as to the correct operation of the module or the security strengths of the generated keys when ported to an operational environment which is not listed on the validation certificate.

2.7 Cryptographic Key Management

The module supports the critical security parameters listed below in Table 6. Please note that the “Input” and “Output” columns in Table 6 are in reference to the module’s logical boundary.

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

CSP/Key	CSP/Key Type	Input	Output	Storage	Zeroization	Use
AES key	AES 128, 192, 256-bit key	Input electronically in plaintext	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	Encryption, decryption
		Internally generated via DRBG	Output in plaintext via API call parameter			Used by calling application
HMAC key	HMAC 512, 1024-bit key	Input electronically in plaintext	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	Message Authentication with SHS
		Internally generated via DRBG	Output in plaintext via API call parameter			Used by calling application
RSA private key	RSA 2048-bit key	Input electronically in plaintext	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	Signature generation, decryption
		Internally generated via DRBG	Output in plaintext via API call parameter			Used by calling application
RSA public key	RSA 2048-bit key	Input electronically in plaintext	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	Signature verification, encryption
		Internally generated via DRBG	Output in plaintext via API call parameter			Used by calling application
DRBG seed	440-bit random value	Internally generated using nonce, personalization string along with entropy input string	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	FIPS-Approved random number generation
Entropy input string	256-bit random value	Externally generated using an NDRNG[18] and input in plaintext	Never exits the module	Plaintext in volatile memory	By API call, power cycle or host reboot	FIPS-Approved random number generation

[18] NDRNG – Non-Deterministic Random Number Generator

CSP/Key	CSP/Key Type	Input	Output	Storage	Zeroization	Use
Hash_DRBG V value	440-bit internal hash DRBG state value	Internally generated	Never	Plaintext in volatile memory	Reboot, power cycle	Used for SP 800-90A Hash_DRBG
Hash_DRBG C value	440-bit internal hash DRBG state value	Internally generated	Never	Plaintext in volatile memory	Reboot, power cycle	Used for SP 800-90A Hash_DRBG

2.7.1 Key Generation

The module uses an SP 800-90A hash-based DRBG implementation to generate cryptographic keys and seeds; and the key generation process follows example 1 of SP 800-133 Rev.1 section 4, using the unmodified output of the DRBG. The module's DRBG is FIPS-Approved (as shown in Annex C to FIPS PUB 140-2). The module also supports the generation of the RSA public/private keys using the RSA key generation function specified in FIPS 186-4. It is the responsibility of the calling applications to ensure that the module is supplied with enough entropy (256 bits) to meet the security strength requirement of the hash-based DRBG. This entropy is supplied by means of callback functions. Those functions must return an error if the minimum entropy strength cannot be met[19].

2.7.2 Key Entry and Output

Keys are passed into the module's logical boundary in plaintext via the exposed APIs, but only from applications resident on the host platform. However, the cryptographic module does not support key entry or key output across the host platform's physical boundary. Similarly, keys and CSPs exit the module in plaintext (but remain in the physical boundary) via the well-defined exported APIs.

2.7.3 Key/CSP Storage and Zeroization

As a software module, the module does not provide for the persistent storage of keys and CSPs. Keys and CSPs stored in RAM can be zeroized by a power cycle or a host platform reboot. Additionally, symmetric and asymmetric 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. The DRBG seed is initialized by the module (internally generated) at power-up and remains stored in RAM until the module is uninitialized by a host platform reboot or power cycle. The keys can also be zeroized by the calling application by invoking an API function.

The key zeroization techniques used for clearing volatile memory, once invoked, take effect immediately and do not allow sufficient time to compromise any plaintext secret and private keys and CSPs stored by the module.

2.8 EMI/EMC

The HP-UX Kernel Cryptographic Module is a software module. Therefore, the only electromagnetic interference produced is that of the host platform on which the module resides and executes.

The test platforms for the module are:

- HPE Integrity BL860c i2 server blade

[19] Caveat: The module generates cryptographic keys whose strengths are modified by available entropy. Thus, there is no assurance of the minimum strength of generated keys.

- HPE Integrity BL890c i6 server blade
- HPE ProLiant DL580 Gen10

This equipment has been tested and found to comply with 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.

2.9 Self-Tests

The HP-UX Kernel Cryptographic Module performs a set of self-tests upon power-up and conditionally during operation as required in FIPS 140-2.

2.9.1 Power-Up Self-Tests

The HP-UX Kernel Cryptographic Module performs the following self-tests at power-up (these tests can also be performed on demand by cycling the power on the host platform, by calling the function *libkcm_FIPS_selftest(),* or by reinitializing the module by using the *libkcm_core_load()* function):

- Software integrity check using RSA 2048 w/ SHA-256 digital signature verification
- Known Answer Tests (KATs) for:
    - AES CBC encrypt and decrypt
    - SHA-256
    - SHA-384
    - SHA-512
    - HMAC SHA-256
    - HMAC SHA-384
    - HMAC SHA-512
    - RSA 2048 w/ SHA-256 signature generation
    - RSA 2048 w/ SHA-256 signature verification
    - DRBG

2.9.2 Conditional Self-Tests

The module performs the following conditional self-tests:

- Continuous DRBG test (CRNGT) for FIPS-Approved random number generator
- RSA Pairwise Consistency test

2.9.3 Health Tests

The HP-UX Kernel Cryptographic Module implements the SP 800-90A Hash_DRBG as its random number generator. The SP 800-90A specification requires that certain health tests be performed conditionally to ensure the security of the DRBG. Therefore, the following health tests are implemented by the cryptographic module:

- DRBG Instantiate Health Test
- DRBG Reseed Health Test
- DRBG Generate Health Test
- DRBG Uninstantiate Health Test

2.9.4 Error Behavior

If any power-up self-test fails, the module will enter a critical error state, during which cryptographic functionality and all data output is inhibited and the module does not get loaded. To clear the error state, the CO must reboot the host platform.

If any conditional self-test or on-demand power-up self-test fails, the module will enter a critical error state, during which cryptographic functionality and all data output is inhibited. Subsequent calls to the module

will not result in data output; rather, the module will only return an error code indicating that it is in an error state. To clear the error state, the CO must explicitly unload the module and reboot the host platform.

2.10 Mitigation of Other Attacks

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

3 Secure Operation

The HP-UX Kernel Cryptographic Module meets level 1 requirements for FIPS 140-2. The sections below describe how to place and keep the module in its FIPS-Approved mode of operation. Section 3.1 below provides guidance to the Crypto Officer for managing the module.

3.1 Secure Management

The following sections provide the necessary guidance to ensure that the module is running in its FIPS-Approved mode of operation.

3.1.1 Installation and initialization

The module will be provided as a binary to the Crypto Officer by HPE. The module is installed after the process of installing the HP-UX kernel which is achieved via HP-UX OS installation. Upon delivery, the Crypto Officer will also receive the HP-UX KCM Installation Procedure documentation that lists the steps and commands to be followed for successful installation of the module. In order to install and setup the module, the following steps must be performed by an authorized CO:

1. Install the module on the system by using the command "*swinstall –s <location _of_depot>/HP-KCM.depot*".
2. Check the success or failure of the module installation by using the command "*swlist | grep –i kcm*". If the module is successfully installed on the system, the following message will be displayed:

   "`HPUX-KCM  A.01.00.00  HP-UX Kernel Cryptography Module`"

3. The CO can also verify whether the module is properly installed or not by using the command "*swverify HPUX-KCM*".

If either "*swinstall*" or "*swverify*" shows any Errors or Warnings, then either the module is not installed or not installed properly. If the installation procedure reports error consistently, then HPE Customer Support should be contacted. Once installed correctly, the module is initialized by a call to a single initialization function *libkcm_core_load()*. When the module is installed and initialized, the module is considered to be running in its FIPS-Approved mode of operation.

3.1.2 Management

Since the Crypto Officer cannot directly interact with the module, no specific management activities are required to ensure that the module runs securely; the module only executes in an Approved mode of operation when operated according to this Security Policy. If any irregular activity is noticed or the module is consistently reporting errors, then HPE Customer Support should be contacted.

3.1.3 Zeroization

The module does not persistently store any key or CSPs. All ephemeral keys used by the module are zeroized upon session termination. Individual keys can be zeroized by API call. All keys can be zeroized by power cycling or rebooting the host platform.

3.2 User Guidance

It is the responsibility of the calling application developer to ensure that only appropriate algorithms, key sizes, and key establishment techniques are applied.

Although the User does not have any ability to modify the configuration of the module, they should notify the Crypto Officer if any irregular activity is noticed.

4 Acronyms

Table 7 below defines the acronyms used in this document.

Table 7 – Acronyms

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
BIOS	Basic Input/Output System
CBC	Cipher Block Chaining
CD	Compact Disc
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CPU	Central Processing Unit
CSEC	Communications Security Establishment Canada
CSP	Critical Security Parameter
DRBG	Deterministic Random Bit Generator
DVD	Digital Video Disc
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
EVFS	Encrypted Volume and File System
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
HDD	Hard Disk Drive
HMAC	(Keyed-) Hash Message Authentication Code
IG	Implementation Guidance
IKE	Internet Key Exchange
IPsec	Internet Protocol Security
ISU	Independent Software Unit
KAT	Known Answer Test
KCM	Kernel Cryptographic Module
LCD	Liquid Crystal Display
LED	Light Emitting Diode
N/A	Not Applicable
NDRNG	Non-Deterministic Random Number Generator

NIST	National Institute of Standards and Technology
OS	Operating System
PCI	Peripheral Component Interconnect
PCIe	PCI Express
PKCS	Public Key Cryptography Standards
PUB	Publication
RAM	Random Access Memory
RBG	Random Bit Generator
RNG	Random Number Generator
RSA	Rivest Shamir and Adleman
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SP	Special Publication
TCP/IP	Transmission Control Protocol/Internet Protocol
USB	Universal Serial Bus
WLI	Whitelisting Infrastructure
XPORT	Transport