Mocana Cryptographic Loadable Kernel Module

Software Version 6.4.1f

Non-Proprietary Security Policy

Document Version 3.5

Mocana Corporation

May 16, 2017

Table of Contents

List of Tables

List of Figures

1. Module Overview

The Mocana Cryptographic Loadable Kernel Module (Software Version 6.4.1f) is a software only, multi-chip standalone cryptographic module that runs on a general purpose computer. The primary purpose of this module is to provide FIPS Approved cryptographic routines to consuming applications via an Application Programming Interface. The physical boundary of the module is the case of the general purpose computer. The logical boundary of the cryptographic module is the kernel module, moc_crypto.ko, as well as the signature file, moc_crypto.ko.sig.

The cryptographic module runs on the following operating environment:

Table 1 – Operational Environment

SW Version	Operating System	Platform
6.4.1f	Wind River Linux 6.0	Intel Atom E3800

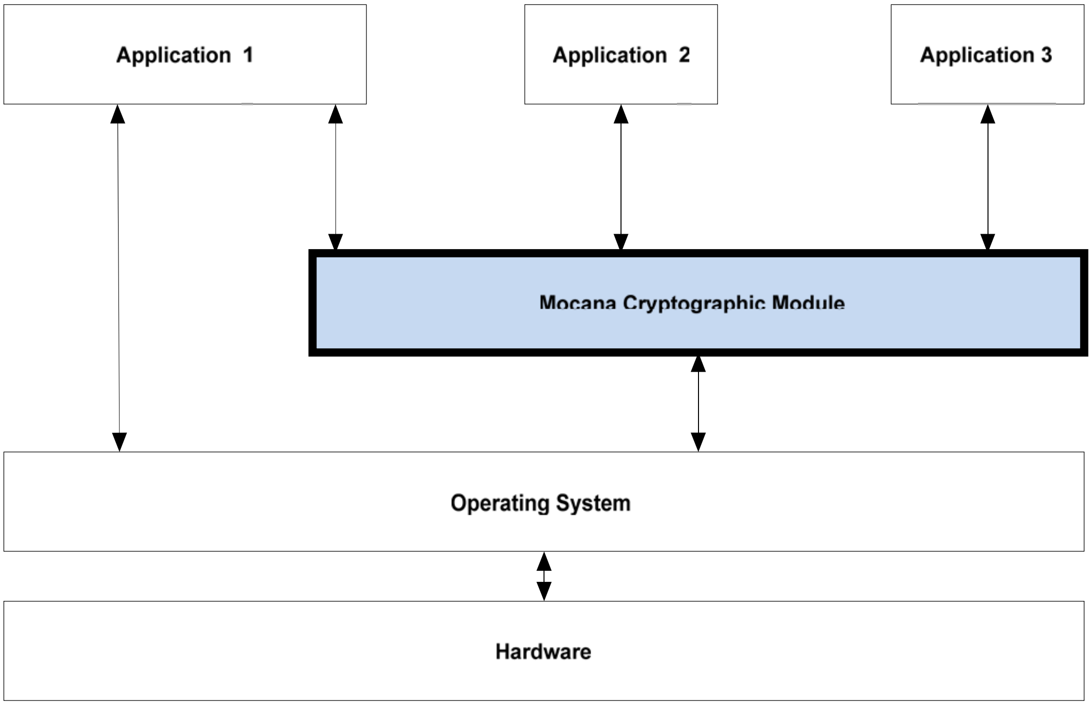

Figure 1 - Cryptographic Module Interface Design

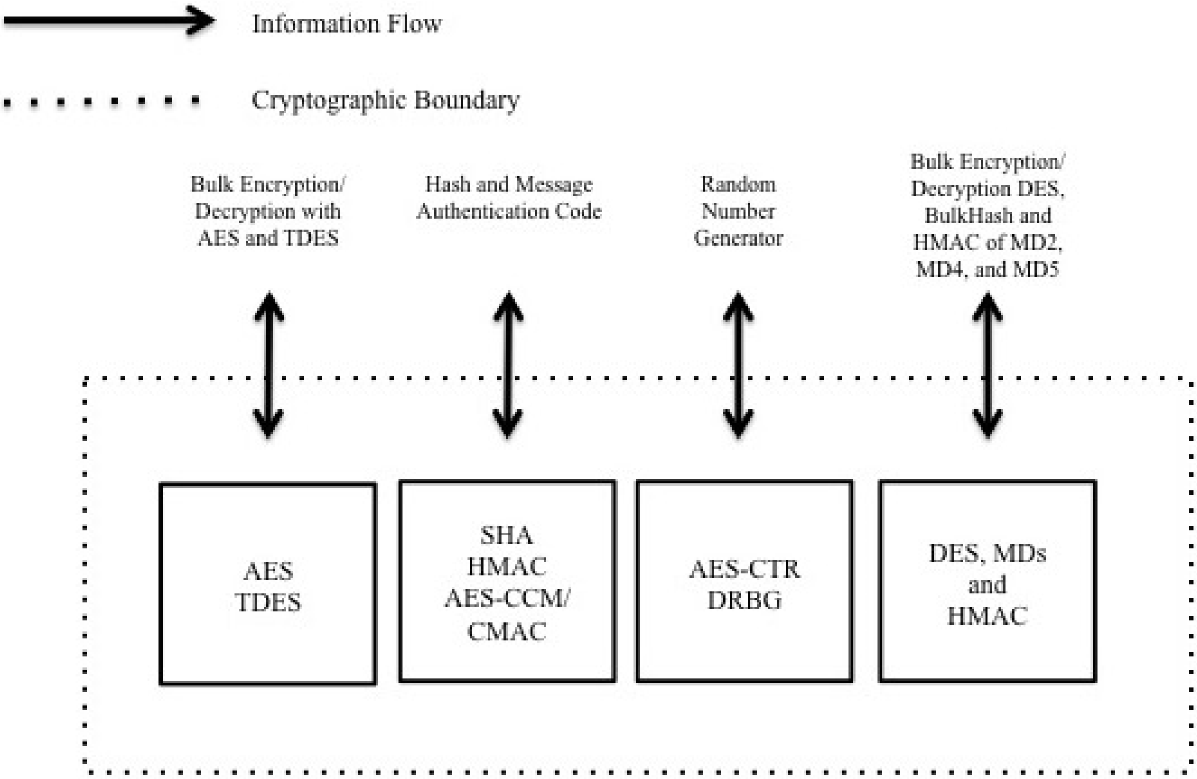
Information Flow
Cryptographic Boundary
Bulk Encryption/ Decryption with AES and TDES
Hash and Message Authentication Code
Random Number Generator
Bulk Encryption/ Decryption DES, BulkHash and HMAC of MD2, MD4, and MD5
AES TDES
SHA HMAC AES-CCM/ CMAC
AES-CTR DRBG
DES, MDs and HMAC

Figure 2 - Logical Cryptographic Boundary

2. Security Level

The cryptographic module meets the overall requirements applicable to Security Level 1 of FIPS 140-2.

Table 2 - Module Security Level Specification

Security Requirements Section	Level
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles, Services and Authentication	1
Finite State 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

3. Modes of Operation

Approved mode of operation

During module initialization, a consuming application can configure the module to utilize all of the following FIPS Approved algorithms:

Table 3 - Algorithms and Software Versions

Algorithm	**Mode/Method/Strength**	**CAVP Cert #**
AES [FIPS 197]	ECB, CBC, OFB, CFB128 and CTR modes; E/D; 128, 192 and 256	4265
AES [FIPS 197]	CCM and CMAC 128, 192 and 256	4265
DRBG [SP 800-90A]	AES-CTR based DRBG- AES-128, AES-192, AES-256	1336
HMAC [FIPS 198]	HMAC-SHA-1; HMAC-SHA-224; HMAC-SHA-256; HMAC-SHA-384; HMAC-SHA-512	2810
SHS [FIPS180-4]	SHA-1 SHA-2: SHA-224; SHA-256; SHA-384; SHA-512	3511
Triple-DES [SP 800-67]	3-key; TCBC; E/D)	2306

During module initialization, a consuming application can configure the module to utilize all, or any subset of the above Approved algorithms. The module's FIPS_powerupSelfTest_Ex() function, which is called during module startup, takes a parameter that points to a configuration table data structure. This data structure contains an array of booleans indexed by an internal Algorithm-ID that will indicate to the module which FIPS algorithms should be initialized for use. The only configuration that was tested as part of the FIPS validation is the configuration which utilized ALL of the Approved algorithms. The CMVP makes no statement as to the correct operation of the module for all other configurations for which operational testing was not performed.

Non-FIPS Approved mode

In addition to the above algorithms, the following algorithms are available in the non-FIPS Approved mode of operation:

AES EAX
AES GCM and GMAC (GCM IV generation not compliant with IG A.5)
AES XCBC
AES XTS (not compliant with IG A.9)
DES

HMAC-MD5
MD2, MD4, MD5
FIPS 186-2 RNG
Triple-DES, 2 key

Note: All the various AES modes, (e.g. EAX, XCBC, XTS, etc.) use the same underlying AES implementation as the approved AES Cert. #4265.

During operation, the module can switch service by service between an Approved mode of operation and a non-Approved mode of operation. The module will transition to the non-Approved mode of operation when one of the above non-Approved security functions is utilized in lieu of an Approved one. The module can transition back to the Approved mode of operation by utilizing an Approved security function.

4. Ports and Interfaces

The physical ports of the module are provided by the general-purpose computer on which the module is installed. The logical interfaces are defined as the API of the cryptographic module. The module's API supports the following logical interfaces: data input, data output, control input, and status output.

Table 4 - Logical Interface Mapping

FIPS 140-2 INTERFACE	Logical Interface
Data Input	Input parameters of API function calls
Data Output	Input parameters of API function calls
Control Input	API Function Calls
Status Output	For FIPS mode, function calls returning status information and return codes provided by API function calls.
Power	None

5. Identification and Authentication Policy

Assumption of Roles

The Mocana Cryptographic Loadable Kernel Module shall support two distinct roles (User and Cryptographic Officer). The cryptographic module does not provide any identification or authentication methods of its own. The Cryptographic Officer and the User roles are implicitly assumed based on the service requested.

Table 5 - Roles and Required Identification and Authentication

Role	Type of Authentication	Authentication Data
User	N/A	N/A
Cryptographic Officer	N/A	N/A

6. Access Control Policy

Roles and Services

Table 6 - Services Authorized for Use in the Approved Mode of Operation

Role	Authorized Services
User	• Self-tests • Show Status • Read Version
Cryptographic-Officer	• AES Encryption • AES Decryption • AES Message Authentication Code • Triple-DES Encryption • Triple-DES Decryption • SHA-1 • SHA-224/256 • SHA-384/512 • HMAC-SHA1 Message Authentication Code • HMAC-SHA224/256 Message Authentication Code • HMAC-SHA384/512 Message Authentication Code • AES-CTR DRBG Random Number Generation • Key Destruction

Other Services

Table 7 - Services Authorized for Use in the non-Approved Mode of Operation

Role	Authorized Services
User	• Self-tests • Show Status • Read Version
Cryptographic-Officer	• DES Encryption • DES Decryption • AES Message Authentication Code • MD2 Hash • MD4 Hash • MD5 Hash • AES EAX Encryption • AES EAX Decryption • AES XCBC Encryption • AES XCBC Decryption • AES XTS Encryption • AES XTS Decryption • FIPS 186-2 Random Number Generation

The cryptographic module supports the following service that does not require an operator to assume an authorized role:

- Self-tests: This service executes the suite of self-tests required by FIPS 140-2. It is invoked by loading the kernel module into executable memory.

Definition of Critical Security Parameters (CSPs)

The following are CSPs that may be contained in the module:

Table 8 - CSP Information

Key	**Description/Usage**	**Generation**	**Storage**	**Entry / Output**	**Destruction**
TDES Key	Used during Triple-DES encryption and decryption	Externally generated.	Temporarily in volatile RAM	Entry: Plaintext Output: N/A	An application program which uses the API may destroy the key. The Key Destruction service zeroizes this CSP.
AES Keys	Used during AES encryption, decryption, CMAC operations	Externally generated.	Temporarily in volatile RAM	Entry: Plaintext Output: N/A	An application program which uses the API may destroy the key. The Key Destruction service zeroizes this CSP.
HMAC Keys	Used during HMAC-SHA-1, 224, 256, 384, 512 operations	Externally generated.	Temporarily in volatile RAM	Entry: Plaintext Output: N/A	An application program which uses the API may destroy the key. The Key Destruction service zeroizes this CSP.
V and Key DRBG values	Used to seed the DRBG for key generation	Externally generated.	Temporarily in volatile RAM	Entry: Plaintext if generated externally Output: N/A	Automatically after use

Note: Key Entry and Output refers to keys crossing the logical boundary of the cryptographic module and not the physical boundary of the general purpose computer.

Definition of Public Keys

The module does not contain any public keys.

Definition of CSPs Modes of Access

Table 9 defines the relationship between access to CSPs and the different module services.

Table 9 - CSP Access Rights within Roles & Services

Role		Service	Cryptographic Keys and CSPs Access Operation
C.O.	**User**		
X		AES Encryption	Use AES Key
X		AES Decryption	Use AES Key
X		AES Message Authentication	Use AES Key
X		Triple-DES Encryption	Use Triple-DES Key
X		Triple- ES Decryption	Use Triple-DES Key
X		SHA-1	Generate SHA-1 Output; no CSP access
X		SHA-224/256	Generate SHA-224/256 Output; no CSP access
X		SHA-384/512	Generate SHA-384/512 Output; no CSP access
X		HMAC-SHA-1 Message Authentication Code	Use HMAC-SHA-1 Key Generate HMAC-SHA-1 Output
X		HMAC-SHA-224/256 Message Authentication Code	Use HMAC-SHA-224/256 Key Generate HMAC-SHA-224/256 Output
X		HMAC-SHA-384/512 Message Authentication Code	Use HMAC-SHA-384/512 Key Generate HMAC-SHA-384/512 Output
X		AES-CTR DRBG Random Number Generation	Use V and Key values to generate random number Destroy V and Key values after use
X		Key Destruction	Destroy All CSPs
	X	Show Status	N/A
	X	Self-Tests	N/A
	X	Read Version	N/A

7. Operational Environment

The FIPS 140-2 Area 6 Operational Environment requirements are applicable because the Mocana Cryptographic Loadable Kernel Module operates in a modifiable operational environment. Please refer to Table 1 for a list of environments for which operational testing of the module was performed.

Integrity Check at Application Start

During the load of the kernel module, the integrity check of the kernel module and constants occurs in the module startup function. It verifies the integrity by executing the HMAC-SHA 1 fingerprint algorithm on the kernel module .ko file, and comparing the result with the signature file. This integrity check is performed as part of the function FIPS_powerupSelfTest(). This function is called automatically by the host O/S upon loading the kernel module into memory via the code snippet below.

```
static int __init
 mss_crypto_init(void)
 {
     int status = 0;
     PRINTDEBUG("moc_crypto_init.\n");

 #ifdef __ENABLE_FIPS_POWERUP_TEST__
     if (OK > (status = FIPS_powerupSelfTest()))
     {
         PRINTDEBUG("powerup test failed!\n");
         goto cleanup;
     }
     else
     {
         PRINTDEBUG("powerup test passed!\n");
         goto cleanup;
     }
 #else
     PRINTDEBUG("powerup test disabled!\n");
 #endif

 cleanup:
      return status;

 }

 static void __exit
 mss_crypto_fini(void)
 {
     PRINTDEBUG("moc_crypto_fini.\n");
 }

 module_init(mss_crypto_init);
 module_exit(mss_crypto_fini);
```

Figure 3 - Code Example for Self-Test

8. Security Rules

The Mocana Cryptographic Loadable Kernel Module design corresponds to the following security rules. This section documents the security rules enforced by the cryptographic module to implement the security requirements of this FIPS 140-2 Level 1 module.

1. The cryptographic module provides two (2) distinct roles. These are the User role and the Cryptographic Officer role.
2. The cryptographic module does not provide any operator authentication.
3. The cryptographic module shall encrypt/decrypt message traffic using the Triple-DES or AES algorithms.
4. The cryptographic module shall perform the following self-tests:

Table 10 - Power-up Self-Tests

Type	Detail
Software Integrity Check	• HMAC-SHA-1
Known Answer Tests	• AES-ECB, CBC, OFB. CFB, CCM, CMAC, CTR, GCM, GMAC and XTS • Triple-DES • HMAC-SHA-1 • HMAC-SHA-224 • HMAC-SHA-256 • HMAC-SHA-384 • HMAC-SHA-512 • SHA-1 • SHA-224 • SHA-256 • SHA-384 • SHA-512 • AES-CTR DRBG (including SP800-90A Health Checks)

Table 10 - Conditional Self-Tests

Type	Detail
Continuous RNG Tests	• AES-CTR DRBG Continuous Test

5. At any time, the operator shall be capable of commanding the module to perform the power-up self-tests by reloading the cryptographic module into memory.
6. The cryptographic module is available to perform services only after successfully completing the power-up self-tests.

7. Data output shall be inhibited during self-tests, zeroization, and error states. Because the logical interface is defined as the API of the crypto module and the API of the crypto module is single-threaded, zeroization must be complete before the API returns control to the calling application.
8. Status information shall not contain CSPs or sensitive data that if misused could lead to a compromise of the module.
9. In the event of a self-test or conditional test failure, the module will enter an error state and a specific error code will be returned indicating which self-test or conditional test has failed. The module will not provide any cryptographic services while in this state.
10. The operating system is restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded). The application that makes calls to the modules is the single user of the modules, even when the application is serving multiple clients.
11. The module does not support key generation.
12. The calling application of the module shall use entropy sources that meet the security strength required for the random bit generation mechanism. A minimum of 112 bits of entropy must be requested by the calling application.
13. DES, MD2, MD4, MD5, AES EAX, AES XCBC, FIPS 186-2 RNG, and Two-key Triple-DES are not allowed for use in the FIPS Approved mode of operation. When these algorithms are used, the module is no longer operating in the FIPS Approved mode of operation. It is the responsibility of the consuming application to zeroize all keys and CSPs prior to and after utilizing these non-Approved algorithms. CSPs shall not be shared between the Approved and non-Approved modes of operation.

9. Physical Security

The FIPS 140-2 Area 5 Physical Security requirements are not applicable because the Mocana Cryptographic Loadable Kernel Module is software only.

10. Mitigation of Other Attacks Policy

The module has not been designed to mitigate any specific attacks outside the scope of FIPS 140-2 requirements.

11. Key Management

The application that uses the module is responsible for appropriate destruction and zeroization of the keys. The library provides API calls for key allocation and destruction. These API calls overwrite the memory occupied by the key information with zeros before that memory is de-allocated. See Key Destruction Service paragraph below.

Key/CSP Zeroization

The application is responsible for calling the appropriate destruction functions from the API. These functions overwrite the memory with zeros and de-allocate the memory. In case of abnormal termination, the Linux kernel overwrites the keys in physical memory before the physical memory is allocated to another process.

12. Guidance

Cryptographic Officer Guidance

The operating system running the Mocana Cryptographic Loadable Kernel Module must be configured in a single-user mode of operation.

The Cryptographic Officer will install the kernel module and associated signature of the module into the proper location within the computer system. For example, the kernel module and signature file may be installed in the /usr/local/lib/module directory, which is protected by Linux access control mechanisms. The module is protected from modification by the integrity self-test performed during startup. The module is initialized by the operating system upon loading the module (kernel module or shared library) into memory for use by calling applications.

Key Destruction Service

There is a context structure associated with every cryptographic algorithm available in this module. Context structures hold sensitive information such as cryptographic keys. These context structures must be destroyed via respective API calls when the application software no longer needs to use a specific algorithm any more. This API call will zeroize all sensitive information including cryptographic keys before freeing the dynamically allocated memory. This will occur while the application process is still in memory, but no longer needs the specific algorithm, which sufficiently protects the keys from compromise. See the *Mocana Cryptographic API Reference* for additional information.

Random Number Generation

The module implements a CTR-based DRBG. The DRBG generates blocks of random numbers with more than 15 bits. During each generation of random numbers, the newly created bits are compared with the previously created bits. If they are not the same, then the newly created bits are saved to be used in a subsequent bit generation comparison test, however, if they are the same then the module enters the error state.

The module accepts input from entropy sources external to the cryptographic boundary for use as seed material for the module’s Approved DRBG’s. External entropy can be added via several APIs available to the crypto-module client application:

MOCANA_addEntropyBit () and MOCANA_addEntropy32Bits().

Module users (the calling applications) shall use entropy sources that meet the security strength required for the random number generation mechanism.

User Guidance

The module must be operated in FIPS Approved mode to ensure that FIPS140-2 validated cryptographic algorithms and security functions are used.

13. Definitions and Acronyms

Table 11 - Acronyms and Terms

Acronym	Term
AES	Advanced Encryption Standard
API	Application Program Interface
CO	Cryptographic Officer
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DES	Data Encryption Standard
DH	Diffie-Hellman
DLL	Dynamic Link Library
DRBG	Deterministic Random Bit Generator
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FIPS	Federal Information Processing Standard
HMAC	Keyed-Hash Message Authentication Code
LKM	Loadable Kernel Module
RAM	Random Access Memory
RNG	Random Number Generator
TDES	Triple-DES
SHA	Secure Hash Algorithm