Cisco Common Cryptographic Module (C3M) - Hybrid

FIPS 140-2 Non Proprietary Security Policy
Level 1 Validation

Version 0.3

December, 11

Table of Contents

1 Introduction

1.1 Purpose

This document is the non-proprietary Cryptographic Module Security Policy for the Cisco Common Cryptographic Module (C3M) - Hybrid. This security policy describes how the Cisco Common Cryptographic Module (C3M) - Hybrid (Software Version: 0.9.8r.1.1) meets the security requirements of FIPS 140-2, and how to operate it in a secure FIPS 140-2 mode. This policy was prepared as part of the Level 1 FIPS 140-2 validation of the Cisco Common Cryptographic Module (C3M) - Hybrid.

FIPS 140-2 (Federal Information Processing Standards Publication 140-2 — *Security Requirements for Cryptographic Modules*) details the U.S. Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the NIST website at http://csrc.nist.gov/groups/STM/index.html.

1.2 Module Validation Level

The following table lists the level of validation for each area in the FIPS PUB 140-2.

No.	Area 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
5	Physical Security	1
6	Operational Environment	1
7	Cryptographic Key management	1
8	Electromagnetic Interface/Electromagnetic Compatibility	1
9	Self-Tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A
	Overall module validation level	**1**

Table 1 Module Validation Level

1.3 References

This document deals only with operations and capabilities of the Cisco Common Cryptographic Module (C3M) - Hybrid in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available from the following sources:

For answers to technical or sales related questions please refer to the contacts listed on the Cisco Systems website at www.cisco.com.

The NIST Validated Modules website (http://csrc.nist.gov/groups/STM/cmvp/validation.html) contains contact information for answers to technical or sales-related questions for the module.

1.4 Terminology

In this document, the Cisco Common Cryptographic Module (C3M) - Hybrid is referred to as C3M, the library, or the module.

1.5 Document Organization

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

Vendor Evidence document
Finite State Machine
Other supporting documentation as additional references

This document provides an overview of the Cisco Common Cryptographic Module (C3M) - Hybrid and explains the secure configuration and operation of the module. This introduction section is followed by Section 2, which details the general features and functionality of the module. Section 3 specifically addresses the required configuration for the FIPS-mode of operation.

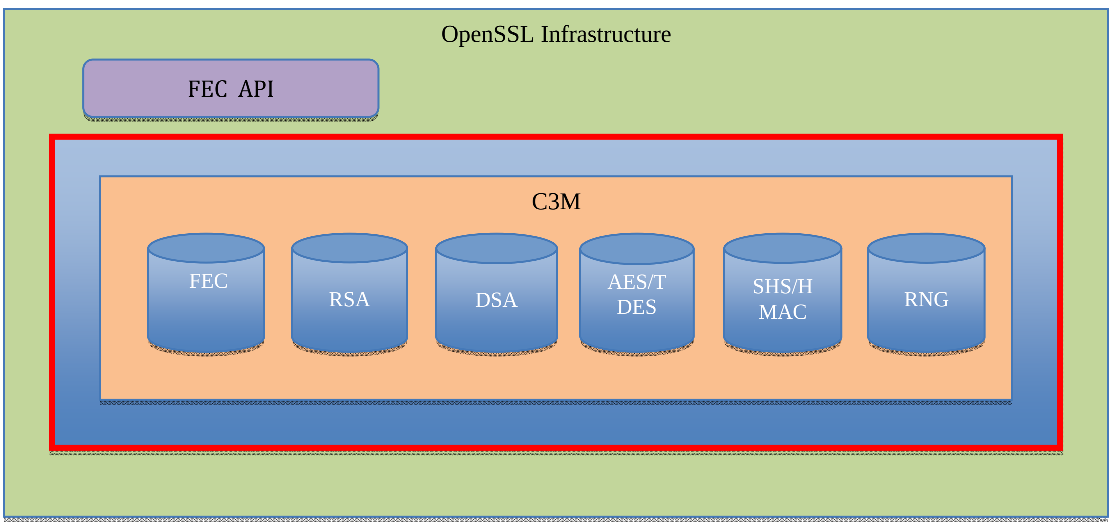
With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Validation Submission Documentation is Cisco-proprietary and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact Cisco Systems.

2 Cisco Common Cryptographic Module (C3M) - Hybrid

The Cisco Common Cryptographic Module (C3M) is a software-hybrid library that provides cryptographic services to a vast array of Cisco's networking and collaboration products. Additionally it provides hardware acceleration for AES encryption-decryption by leveraging AES-NI implemented on a number of processors. The module provides FIPS validated cryptographic algorithms for services such as sRTP, SSH, TLS, 802.1x etc. The module does not implement any of the protocols directly. Instead, it provides the cryptographic primitives and functions to allow a developer to implement various protocols.

The module is based on OpenSSL FIPS canister with additions to support Suite B algorithms and AES NI instruction set.

2.1 Cryptographic Module Physical Characteristics



FEC: Fundamental Elliptical Curve cryptography

Figure 1 C3M-Hybrid logical block diagram

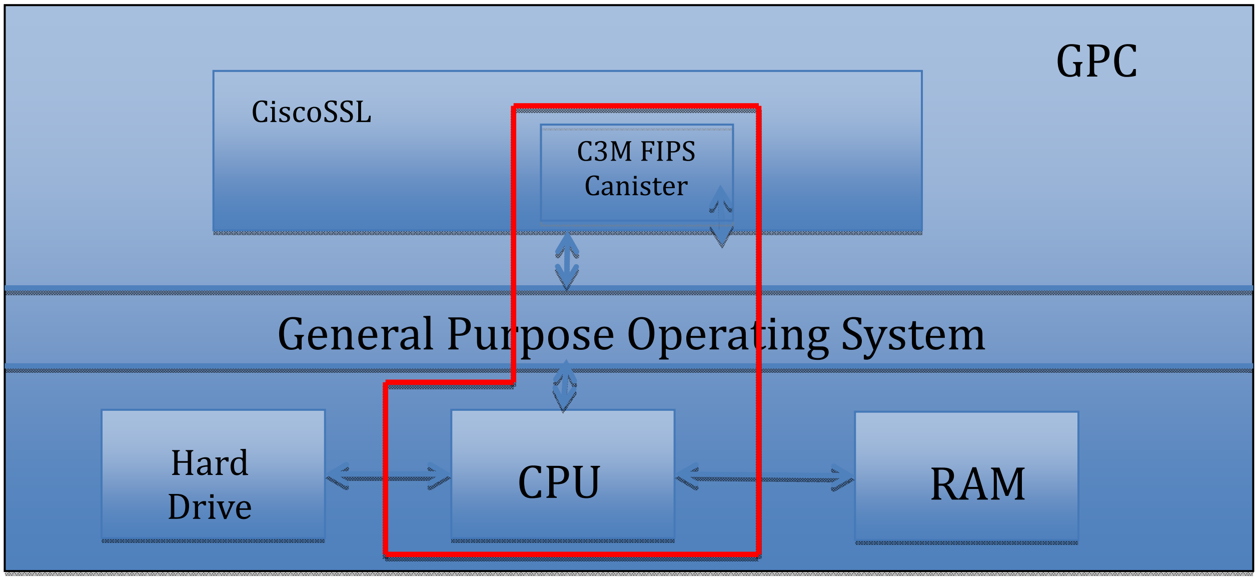


Figure 2 C3M =-Hybrid Physical block diagram

The module is a multiple-chip standalone cryptographic module. For the purposes of the FIPS 140-2 level 1 validation, the C3M module is a software-hybrid composed of a single object module file named fipscanister.o (Linux/Unix) or fipscanister.lib (Microsoft Windows) and the processor implementing AES-NI.

The module's logical block diagram is shown in Figure 1 above. The red border denotes the logical cryptographic boundary of the module. The physical cryptographic boundary of the module is the enclosure of the system on which it is executing.

This module was tested on the following platforms for the purposes of this FIPS validation:

#	Operating System	Processor
1	FreeBSD 8.2 (AES-NI)	Intel Core i5
2	Red Hat Enterprise Linux v5 (AES-NI)	Intel Xeon
3	Windows 7 SP1 (AES-NI)	Intel Core i5
4	Red Hat Enterprise Linux v5 (AES-NI)	Intel Core i7

Table 2 Tested Operational Environment

2.2 Module Interfaces

The physical ports of the Module are the same as the system on which it is executing. The logical interface is a C-language application program interface (API).

The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the actual API functions. The Status Output interface includes the return values of the API functions.

The module provides a number of physical and logical interfaces to the device, and the physical interfaces provided by the module are mapped to the following FIPS 140-2 defined logical interfaces: data input, data output, control input, status output, and power. The logical interfaces and their mapping are described in the following table:

Module Interface	FIPS 140-2 Logical Interface
API input parameters	Data Input Interface
API output parameters	Data Output Interface
API function calls	Control Input Interface
API return codes	Status Output Interface
N/A	Power Interface

Table 3 – FIPS 140-2 Logical Interfaces

2.3 Roles and Services

The Module meets all FIPS 140-2 level 1 requirements for Roles and Services, implementing both Crypto-User and Crypto-Officer roles. As allowed by FIPS 140-2, the Module does not support user authentication for those roles. Only one role may be active at a time and the Module does not allow concurrent operators.

The User and Crypto Officer roles are implicitly assumed by the entity accessing services implemented by the Module. The Crypto Officer can install and initialize the Module. The Crypto Officer role is implicitly entered when installing the Module or performing system administration functions on the host operating system.

- User Role: Loading the Module and calling any of the API functions. This role has access to all of the services provided by the Module.
- Crypto-Officer Role: Installation of the Module on the host computer system. This role is assumed implicitly when the system administrator installs the Module library file.

Service	Role	CSP	Access
Symmetric encryption/decryption	User, Crypto Officer	Symmetric keys AES, Triple-DES	Execute
Key transport	User, Crypto Officer	Asymmetric private key RSA	Execute

Key agreement	User, Crypto Officer	DH and ECDH private key	Execute
Digital signature	User, Crypto Officer	Asymmetric private key RSA, DSA, ECDSA	Execute
Symmetric key generation	User, Crypto Officer	Symmetric key AES, Triple-DES	Write/execute
Keyed Hash (HMAC)	User, Crypto Officer	HMAC SHA-1 key, HMAC-SHA-1	Execute
Message digest (SHS)	User, Crypto Officer	None SHA-1, SHA-2	N/A
Random number generation	User, Crypto Officer	Seed key, seed AES	Write/execute
Show status	User, Crypto Officer	None	N/A
Module initialization	User, Crypto Officer	None	N/A
Self-test	User, Crypto Officer	None	N/A
Zeroize	User, Crypto Officer	Symmetric key, asymmetric key, HMAC-SHA-1 key, seed key AES	Zeroize

Table 4 Roles, Services and Keys

2.4 Physical Security

The module is a software-hybrid which is designated as multi-chip standalone. The hardware component is the GPC processor that is production grade and manufactured by well-known companies such as Intel.

2.5 Cryptographic Algorithms

The module implements a variety of approved and non-approved algorithms.

2.5.1 Approved Cryptographic Algorithms

The routers support the following FIPS 140-2 approved algorithm implementations:

Algorithm	Algorithm Certification Number
AES	1758
DSA	550
ECDSA	234
HMAC	1031
RNG	937
RSA	876
SHS	1544
Triple-DES	1139

Table 5 Approved Cryptographic Algorithms

2.5.2 Non-FIPS Approved Algorithms Allowed in FIPS Mode

The module supports the following non-FIPS approved algorithms which are permitted for use in the FIPS approved mode:

- Diffie-Hellman (key agreement; key establishment methodology provides between 80 and 152 bits of encryption strength)
- EC Diffie-Hellman (key agreement; key establishment methodology provides between 80 and 256 bits of encryption strength)
- RSA (key wrapping; key establishment methodology provides between 80 and 152 bits of encryption strength)

2.5.3 Non-Approved Cryptographic Algorithms

The module does not implement any non-approved cryptographic algorithms.

2.6 Cryptographic Key Management

2.6.1 Key Generation

The Module supports generation of DH, ECDH, DSA, RSA, and ECDSA public-private key pairs. The Module employs an ANSI X9.31 compliant random number generator for creation of asymmetric and symmetric keys.

The developer shall use entropy sources that contain at least 128 bits of entropy to seed the RNG as the module is not capable of detecting randomness or quality of the seeding material provided.

2.6.2 Key Storage

Public and private keys are provided to the Module by the calling process, and are destroyed when released by the appropriate API function calls. The Module does not perform persistent storage of keys.

2.6.3 Key Access

An authorized application as user (the Crypto-User) has access to all key data generated during the operation of the Module.

2.6.4 Key Protection and Zeroization

Keys residing in internally allocated data structures can only be accessed using the Module defined API. The operating system protects memory and process space from unauthorized access. Zeroization of sensitive data is performed automatically by API function calls for intermediate data items, and on demand by the calling process using Module provided API function calls provided for that purpose.

Only the process that creates or imports keys can use or export them. No persistent storage of key data is performed by the Module. All API functions are executed by the invoking process in a non- overlapping sequence such that no two API functions will execute concurrently.

The calling process can perform key zeroization of keys by calling an API function. Additionally keys can be zeroized by power-cycling the module.

The module supports the following keys and critical security parameters (CSPs):

ID	Algorithm	Size	Description
Symmetric Keys	AES, Triple-DES	AES: 128, 192, 256 bits Triple-DES: 112, 168 bits	Used for symmetric encryption/decryption
Asymmetric Keys	RSA, DSA, ECDSA	RSA: 1024-4096 bits DSA: 1024 bits ECDSA: P-192, P-256, P-384, P-521	Used for signature generation/verification RSA: Also used for key transport
Diffie-Hellman/ EC Diffie-Hellman private exponent	DH, ECDH	DH: 1024-4096 bits ECDH: P-192, P-256, P-384, P-521	Used for key agreement
RNG Seed	X9.31	16 bytes	This is the seed used for X9.31 RNG

ID	Algorithm	Size	Description
RNG Seed key	X9.31	128, 192 and 256 bits	This is the seed key used for X9.31 RNG
Keyed Hash key	HMAC	All supported key sizes for HMAC	Used for keyed hash

Table 6 Cryptographic Keys and CSPs

2.7 Self-Tests

The Module performs both power-up self tests at module initialization[1] and continuous condition tests during operation. Input, output, and cryptographic functions cannot be performed while the Module is in a self-test or error state as the module is single threaded and will not return to the calling application until the power-up self tests are complete. If the power-up self tests fail subsequent calls to the module will fail and thus no further cryptographic operations are possible.

2.7.1 Self-tests performed

- POST tests
    - AES Known Answer Test
    - AES-CCM Known Answer Test
    - AES-GCM Known Answer Test
    - AES-CMAC Known Answer Test
    - Triple-DES Known Answer Test
    - DSA Sign/Verify Test
    - RSA Signature Known Answer Test
    - ECDSA Sign/Verify Test
    - RNG Known Answer Test
    - HMAC Known Answer Test (performed for each supported SHA)
    - SHA-1 Known Answer Test
    - SHA-2 Known Answer Test (includes SHA-224, SHA-256, SHA-384 and SHA-512)
    - Software Integrity Test

- Conditional tests
    - Pairwise consistency test for RSA, DSA, and ECDSA signature keys
    - Continuous random number generation test for approved and non-approved RNGs

[1] The FIPS mode initialization is performed when the application invokes the FIPS_mode_set() call which returns a "1" for success and "0" for failure

A single initialization call, FIPS_mode_set(), is required to initialize the Module for operation in the FIPS 140-2 Approved mode. When the Module is in FIPS mode all security functions and cryptographic algorithms are performed in Approved mode.

The FIPS mode initialization is performed when the application invokes the FIPS_mode_set() call which returns a "1" for success and "0" for failure. Interpretation of this return code is the responsibility of the host application. Prior to this invocation the Module is uninitialized in the non- FIPS mode by default.

The FIPS_mode_set() function verifies the integrity of the runtime executable using a HMAC-SHA-1 digest computed at build time. If this computed HMAC-SHA-1 digest matches the stored known digest then the power-up self-test, consisting of the algorithm specific Pairwise Consistency and Known Answer tests, is performed. If any component of the power-up self-test fails an internal global error flag is set to prevent subsequent invocation of any cryptographic function calls. Any such power-up self test failure is a hard error that can only be recovered by reinstalling the Module[2]. If all components of the power-up self-test are successful then the Module is in FIPS mode. The power-up self-tests may be performed at any time with a separate function call, FIPS_selftest(). This function call also returns a "1" for success and "0" for failure, and interpretation of this return code is the responsibility of the host application.

A power-up self-test failure can only be cleared by a successful FIPS_mode_set() invocation. No operator intervention is required during the running of the self-tests.

3 Secure Operation

The tested operating systems segregate user processes into separate process spaces. Each process space is an independent virtual memory area that is logically separated from all other processes by the operating system software and hardware. The Module functions entirely within the process space of the process that invokes it, and thus satisfies the FIPS 140-2 requirement for a single user mode of operation.

The Module is installed using one of the set of instructions in CiscoSSL_DeveloperGuide.pdf appropriate to the target system. A complete revision history of the source code from which the Module was generated is maintained in a version control database[3]. The HMAC-SHA-1 of the Module distribution file as tested by the CMT Laboratory is verified during installation of the Module file as described in CiscoSSL_DeveloperGuide.

The HMAC fingerprint of the validated distribution file is
d592c5af90566c18fca75b0d8883e92f65b0aa97

[2] The FIPS_mode_set() function could be re-invoked but such re-invocation does not provide a means from recovering from an integrity test or known answer test failure.
[3] This database is internal to Cisco since the intended use of this crypto module is by Cisco dev teams

Upon initialization6 of the Module, the module will run its power-up self tests. Successful completion of the power-up self tests ensures that the module is operating in the FIPS mode of operation.

The self-tests can be called on demand by reinitializing the module using the FIPS_mode_set() function call, or alternatively using the FIPS_selftest() function call.

CISCO EDITOR'S NOTE: You may now include all standard Cisco information included in all documentation produced by Cisco. Be sure that the following line is in the legal statements at the end of the document:

By printing or making a copy of this document, the user agrees to use this information for product evaluation purposes only. Sale of this information in whole or in part is not authorized by Cisco Systems.