FIPS 140-2 Non-Proprietary Security Policy

CryptoComply

Software Version 3.0

Document Version 1.4

October 15, 2018

SafeLogic Inc.
530 Lytton Ave., Suite 200
Palo Alto, CA 94301
www.safelogic.com

Abstract

This document provides a non-proprietary FIPS 140-2 Security Policy for CryptoComply.

SafeLogic's CryptoComply is designed to provide FIPS 140-2 validated cryptographic functionality and is available for licensing. For more information, visit https://www.safelogic.com/cryptocomply/.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1 About FIPS 140

Federal Information Processing Standards Publication 140-2 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic modules to be deployed in a Sensitive but Unclassified environment. The National Institute of Standards and Technology (NIST) and Communications Security Establishment Canada (CSE) Cryptographic Module Validation Program (CMVP) run the FIPS 140 program. The NVLAP accredits independent testing labs to perform FIPS 140 testing; the CMVP validates modules meeting FIPS 140 validation. *Validated* is the term given to a module that is documented and tested against the FIPS 140 criteria.

More information is available on the CMVP website at:
http://csrc.nist.gov/groups/STM/cmvp/index.html.

1.2 About this Document

This non-proprietary Cryptographic Module Security Policy for CryptoComply from SafeLogic provides an overview of the product and a high-level description of how it meets the security requirements of FIPS 140-2. This document contains details on the module's cryptographic keys and critical security parameters. This Security Policy concludes with instructions and guidance on running the module in a FIPS 140-2 mode of operation.

CryptoComply may also be referred to as the "module" in this document.

1.3 External Resources

The SafeLogic website (https://www.safelogic.com) contains information on SafeLogic services and products. The Cryptographic Module Validation Program website contains links to the FIPS 140-2 certificate and SafeLogic contact information.

1.4 Notices

2. CryptoComply

2.1 Cryptographic Module Specification

CryptoComply is a standards-based, drop-in compliant cryptographic engine for servers, network appliances, mobile devices, and other deployments. The module delivers core cryptographic functions and features robust algorithm support, including Suite B algorithms. CryptoComply offloads secure key management, data integrity, data at rest encryption, and secure communications to a trusted implementation. The validated version of the library is *3.0*. For the purposes of the FIPS 140-2 validation, its embodiment type is defined as multi-chip standalone.

The cryptographic module was tested on the following operational environments on the general-purpose computer (GPC) platforms detailed below:

#	Operational Environment	Processor Family	Compiler
1	Ubuntu Linux 14.04 LTS	Intel Xeon E5 (without PAA)	Clang (4.0.0)
2	Ubuntu Linux 16.04	Intel Xeon E5	Clang (4.0.0)
3	Ubuntu Linux 15.04	POWER8 (without PAA)	Clang (4.0.0)
4	Ubuntu Linux 17.04	POWER8	Clang (4.0.0)
5	Ubuntu Linux 17.04	POWER9	Clang (4.0.0)

Table 1 – Tested Operational Environments

2.1.1 Validation Level Detail

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

FIPS 140-2 Section Title	Validation Level
Cryptographic Module Specification	1
Cryptographic 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
Electromagnetic Interference / Electromagnetic Compatibility	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A
Overall Level	1

Table 2 – Validation Level by FIPS 140-2 Section

2.1.2 Cryptographic Boundary

The module is a software library providing a C-language application program interface (API) for use by other processes that require cryptographic functionality. All operations of the module occur via calls from host applications and their respective internal daemons/processes. As such there are no untrusted services calling the services of the module.

The physical cryptographic boundary is the general-purpose computer on which the module is installed. The logical cryptographic boundary of the CryptoComply module is a single object file named cryptocomply.o, which is statically linked to the calling application. The module performs no communications other than with the calling application and the host operating system.

Figure 1 shows the logical relationship of the cryptographic module to the other software and hardware components of the computer:

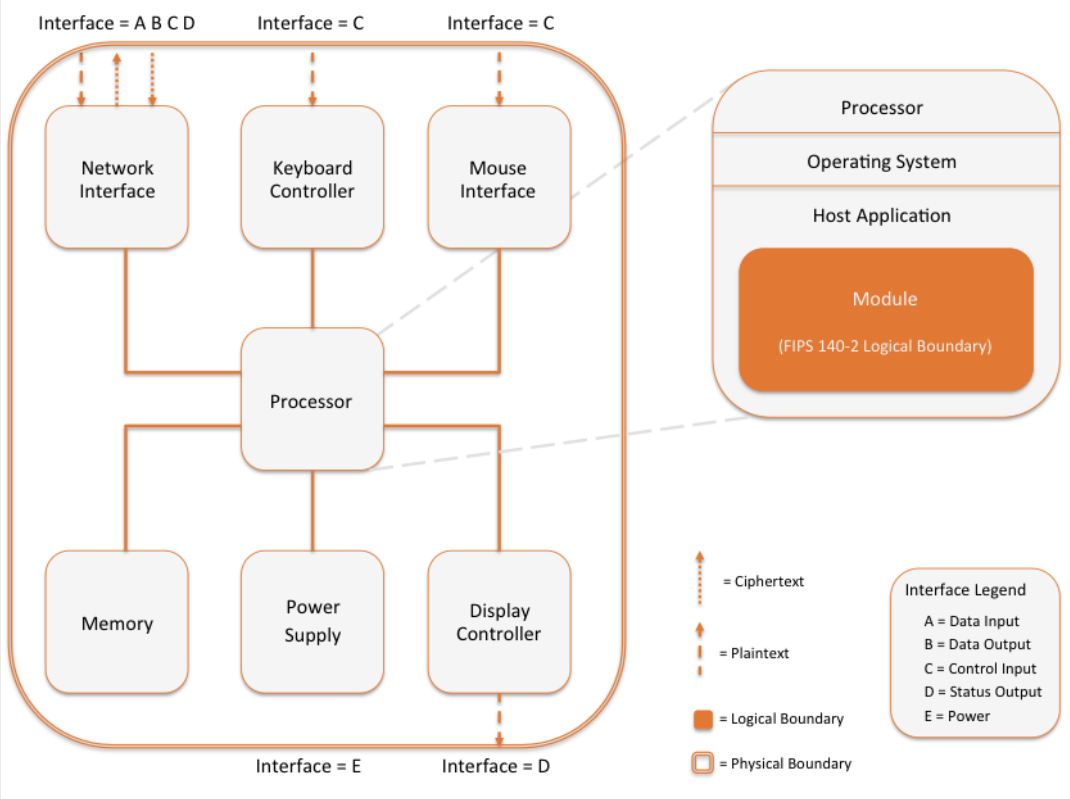

Figure 1 – Logical Boundary

2.1.3 Modes of Operation

The module supports two modes of operation: Approved and Non-Approved. The module will be in FIPS-Approved Mode when all power-on self-tests have completed successfully and only Approved algorithms are invoked. See Table 7 below for a list of the supported Approved algorithms and Table 8 for allowed algorithms. The non-Approved mode is entered when a non-Approved algorithm is invoked. See Table 9 for a list of non-Approved algorithms.

2.1.4 Cryptographic Module Ports and Interfaces

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 input parameters. The Status Output interface includes the return values of the API functions. These are listed in the table below:

FIPS Interface	Physical Ports	Logical Interfaces
Data Input	Physical ports of the tested platforms	API input parameters
Data Output	Physical ports of the tested platforms	API output parameters and return values
Control Input	Physical ports of the tested platforms	API input parameters
Status Output	Physical ports of the tested platforms	API return values
Power Input	Physical ports of the tested platforms	N/A

Table 3 – Ports and Interfaces

As a software module, control of the physical ports is outside module scope. However, when the module is performing self-tests, or is in an error state, all output on the module's logical data output interfaces is inhibited.

2.2 Roles, Authentication and Services

The cryptographic module implements both User and Crypto Officer (CO) roles. The module does not support user authentication. The User and CO roles are implicitly assumed by the entity accessing services implemented by the module. A user is considered the owner of the thread that instantiates the module and, therefore, only one concurrent user is allowed.

The Approved services supported by the module and access rights within services accessible over the module's public interface are listed in the table below:

Service	Approved Security Functions	Keys and/or CSPs	Roles	Access Rights to Keys and/or CSPs
Module Initialization	N/A	N/A	CO	N/A
Symmetric encryption/decryption	AES, TDES	AES, TDES symmetric keys	User, CO	Execute
Keyed hashing	HMAC-SHA	HMAC key	User, CO	Execute
Hashing	SHS	None	User, CO	N/A
Random Bit Generation	CTR_DRBG	DRBG seed, internal state V and Key values	User, CO	Write/Execute
Signature generation/ verification	CTR_DRBG RSA ECDSA	RSA, ECDSA private key	User, CO	Write/Execute
Key Transport	RSA	RSA private key	User, CO	Write/Execute
Key Generation	CTR_DRBG RSA ECDSA	RSA, ECDSA private key	User, CO	Write/Execute
On-Demand Self-test	None	None	User, CO	Execute
Zeroization	None	All keys	User, CO	Write/Execute
Show status	None	None	User, CO	N/A

Table 4 – Approved Services, Roles and Access Rights

The module provides the following non-Approved services that utilize algorithms listed below in Table 9:

Service	Non-Approved Functions	Roles	Keys and/or CSPs
Symmetric encryption/decryption	AES, DES, TDES	User, CO	N/A
Hashing	MD4, MD5, POLYVAL	User, CO	N/A
Signature generation/ verification	RSA ECDSA	User, CO	N/A
Key Transport	RSA	User, CO	N/A
Key Generation	RSA ECDSA	User, CO	N/A

Table 5 – Non-Approved or Non-Security Relevant Services

The module also provides the following non-Approved or non-security relevant services over a non-public interface:

Service	Approved Security Functions	Roles	Access Rights to Keys and/or CSPs
Large integer operations	None	User, CO	N/A
Disable automatic generation of CTR_DRBG "additional_input" parameter	CTR_DRBG	User, CO	N/A
Wegman-Carter hashing with POLYVAL	None	User, CO	N/A

Table 6 - Non-Security Relevant Services

2.3 Physical Security

The cryptographic module is comprised of software only and thus does not claim any physical security.

2.4 Operational Environment

The cryptographic module operates under Ubuntu Linux 14.04 LTS, 15.04, 16.04 and 17.04. The module runs on a GPC running one of the operating systems specified in Table 1. Each approved operating system manages processes and threads in a logically separated manner. The module's user is considered the owner of the calling application that instantiates the module.

The cryptographic module is also supported on the following operating environments for which operational testing and algorithm testing was not performed:

- Android 6.0 and later
- CentOS 6.3 and later
- Red Hat Enterprise Linux 6.3 and later

As per FIPS 140-2 Implementation Guidance G.5, compliance is maintained for other versions of the respective operational environments where the module binary is unchanged. No claim can be made as to the correct operation of the module or the security strengths of the generated keys if any source code is changed and the module binary is reconstructed.

The GPC(s) used during testing met Federal Communications Commission (FCC) FCC Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part 15, Subpart B. FIPS 140-2 validation compliance is maintained when the module is operated on other versions of the GPOS running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

2.5 Cryptographic Algorithms & Key Management

2.5.1 Approved Cryptographic Algorithms

The module implements the following FIPS 140-2 Approved algorithms:

CAVP Cert #	Algorithm	Standard	Mode/Method	Use
4558	AES	SP 800-38A FIPS 197 SP 800-38F SP 800-38D	CBC, ECB, CTR, GCM, KW	Encryption, Decryption, Key Wrapping, Key Unwrapping, Decryption, Authentication
2428	Triple-DES	SP 800-20	TCBC, TECB	Encryption, Decryption
1112 1240 (CVL)	ECDSA	FIPS 186-4	Signature Generation Component, Key Pair Generation, Signature Generation, Signature Verification, Public Key Validation	Digital Signature Services
3011	HMAC	FIPS 198-1	HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	Generation, Authentication
3736	SHA	FIPS 180-4	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	Digital Signature Generation, Digital Signature Verification, non-Digital Signature Applications
1507	DRBG	SP 800-90Arev1	CTR_DRBG	Random Bit Generation
2485	RSA	FIPS 186-4	Key Generation, Signature Generation, Signature Verification	Digital Signature Services

Table 7 – Approved Algorithms and CAVP Certificates

2.5.2 Allowed Cryptographic Algorithms

The module supports the following non-FIPS 140-2 Approved but allowed algorithms that may be used in the Approved mode of operation:

Algorithm	Use
RSA Key Transport	RSA key wrapping (key establishment methodology provides between 112 and 256 bits of encryption strength)
NDRNG	Used only to seed the Approved DRBG

Table 8 – Allowed Algorithms

2.5.3 Non-Approved Cryptographic Algorithms

The module employs the methods listed in Table 9, which are not allowed for use in a FIPS-Approved mode. Their use will result in the module operating in a non-Approved mode.

MD5, MD4	DES
AES-GCM (non-compliant)	AES (non-compliant)
ECDSA (non-compliant)	RSA (non-compliant)
POLYVAL	Triple-DES (non-compliant)

Table 9 – Non-Approved Algorithms

2.5.4 Cryptographic Key Management

The table below provides a complete list of Private Keys and CSPs used by the module:

Key/CSP Name	Key Description	Generated/ Input	Output
AES Key	AES (128/192/256) encrypt / decrypt key	Input via API in plaintext	Output via API in plaintext
AES-GCM Key	AES (128/192/256) encrypt / decrypt / generate / verify key	Input via API in plaintext	Output via API in plaintext
AES Wrapping Key	AES (128/192/256) key wrapping key	Input via API in plaintext	Output via API in plaintext
Triple-DES Key	Triple-DES (3-Key) encrypt / decrypt key	Input via API in plaintext	Output via API in plaintext
ECDSA Signing Key	ECDSA (P-224/P-256/P-384/P-521) signature generation key	Internally Generated or input via API in plaintext	Output via API in plaintext
HMAC Key	Keyed hash key (160/224/256/384/512)	Input via API in plaintext	Output via API in plaintext
RSA Key (Key Transport)	RSA (2048 to 16384 bits) key decryption (private key transport) key	Internally Generated or input via API in plaintext	Output via API in plaintext
RSA Signature Generation Key	RSA (2048 to 16384 bits) signature generation key	Internally Generated or input via API in plaintext	Output via API in plaintext
CTR_DRBG V (Seed)	128 bits	Internally Generated	Does not exit the module
CTR_DRBG Key	256 bits	Internally Generated	Does not exit the module
CTR_DRBG Entropy Input	384 bits	Input via API in plaintext	Does not exit the module

Table 10 – Supported Keys and CSPs

2.5.5 Public Keys

The table below provides a complete list of the Public keys used by the module:

Public Key Name	Key Description
ECDSA Verification Key	ECDSA (P-224/P-256/P-384/P-521) signature verification key
RSA Key (Key Transport)	RSA (2048 to 16384 bits) key encryption (public key transport) key
RSA Signature Verification Key	RSA (1024 to 16384 bits) signature verification public key

Table 11 – Supported Public Keys

2.5.6 Key Generation

The module supports generation of ECDSA and RSA key pairs as specified in Section 5 of NIST SP 800-133. The module employs a NIST SP800-90A random number generator for creation of the seed for asymmetric key generation. The module requests a minimum number of 128 bits of entropy from its Operational Environment per each call.

The output data path is provided by the data interfaces and is logically disconnected from processes performing key generation or zeroization. No key information will be output through the data output interface when the module zeroizes keys.

2.5.7 Key Storage

The cryptographic module does not perform persistent storage of keys. Keys and CSPs are passed to the module by the calling application. The keys and CSPs are stored in memory in plaintext. Keys and CSPs residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the module defined API. The operating system protects memory and process space from unauthorized access.

2.5.8 Key Zeroization

The module is passed keys as part of a function call from a calling application and does not store keys persistently. The calling application is responsible for parameters passed in and out of the module. The Operating System and the calling application are responsible to clean up temporary or ephemeral keys.

2.6 Self-Tests

FIPS 140-2 requires the module to perform self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. Some functions require conditional tests during normal operation of the module. The supported tests are listed and described in this section.

2.6.1 Power-On Self-Tests

Power-on self-tests are run upon the initialization of the module and do not require operator intervention to run. If any of the tests fail, the module will not initialize. The module will enter an error state and no services can be accessed.

The module implements the following power-on self-tests:

Type	Test
Integrity Test	HMAC-SHA-512
Known Answer Test	AES KAT (encryption and decryption. Key size: 128-bits)
	AES-GCM KAT (encryption and decryption. Key size: 128-bits)
	Triple-DES KAT (encryption and decryption. Key size: 168-bits)
	ECDSA KAT (signature generation/signature verification. Curve: P-256)
	HMAC KAT (HMAC-SHA-1, HMAC-SHA-512)
	SP 800-90A CTR_DRBG KAT (Key size: 256-bits)
	RSA KAT (signature generation/signature verification and encryption/decryption. Key size: 2048-bit)
	SHA KAT (SHA-1, SHA-256, SHA-512)

Table 12 – Power-On Self-Tests

Each module performs all power-on self-tests automatically when the module is initialized. All power-on self-tests must be passed before a User/Crypto Officer can perform services. The power-on self-tests can be run on demand by power-cycling the host platform.

2.6.2 Conditional Self-Tests

Conditional self-tests are run during operation of the module. If any of these tests fail, the module will enter an error state, where no services can be accessed by the operators. The module can be re-initialized to clear the error and resume FIPS mode of operation. Each module performs the following conditional self-tests:

Type	Test
Pair-wise Consistency Test	ECDSA Key Pair generation RSA Key Pair generation
CRNGT	Performed on NDRNG per IG 9.8
DRBG Health Tests	Performed on DRBG, per SP 800-90A Section 11.3. Required per IG C.1.

Table 13 – Conditional Self-Tests

Pairwise consistency tests are performed for both possible modes of use, e.g. Sign/Verify and Encrypt/Decrypt.

2.7 Mitigation of Other Attacks

The module is not designed to mitigate against attacks that are outside of the scope of FIPS 140-2.

3. Guidance and Secure Operation

3.1 Installation Instructions

The module is provided directly to solution developers and is not available for direct download to the general public. The module and its host application are to be installed on an operating system specified in this document or one where portability is maintained.

3.2 Secure Operation

The module is not distributed as a standalone library and is only used in conjunction with a host application.

3.2.1 Initialization

The cryptographic module is initialized by loading the module before any cryptographic functionality is available. In User Space the operating system is responsible for the initialization process and loading of the library. The module is designed with a default entry point (DEP) that ensures that the power-on tests are initiated automatically when the module is loaded.

3.2.2 Usage of AES OFB, CFB and CFB8

In approved mode, users of the module must not utilize AES OFB, CFB and CFB8.

3.2.3 Usage of AES-GCM

In approved mode, users of the module must not utilize GCM with an externally generated IV unless the source of the IV is also FIPS approved for GCM IV generation.

The module's implementation of AES-GCM is used together with an application that executes outside of the module's cryptographic boundary. The application negotiates the protocol session's keys and the 32-bit nonce value of the IV. The nonce field and counter portions conform to the requirements in Provision 3 of IG A.5.

Per IG A.5, in the event module power is lost and restored the consuming application must ensure that any of its AES-GCM keys used for encryption or decryption are re-distributed.

3.2.4 TLS Operations

The module does not implement the TLS protocol. It implements the cryptographic operations that can be used to implement the TLS protocol.

3.2.5 Usage of Triple-DES

It is the calling application's responsibility to make sure that the three keys (k1, k2, and k3) are independent. Two-key triple-DES usage will bring the module into the non-Approved mode of operation implicitly.

Per IG A.13, the module shall have a limit of either 2^32 or 2^28 encryptions with the same Triple-DES key. The calling application is responsible for ensuring the module's compliance with this requirement.

3.2.6 RSA and ECDSA Keys

The module allows the use of 1024 bits RSA keys for legacy purposes including signature generation, which is disallowed to be used in FIPS Approved mode as per NIST SP800-131A. Therefore, the cryptographic operations with the non-approved key sizes will result in the module operating in non-Approved mode implicitly.

Approved algorithms shall not use the keys generated by the module's non-Approved key generation methods.

4. References and Acronyms

4.1 References

The following Standards are referred to in this Security Policy:

Abbreviation	Full Specification Name
FIPS 140-2	Security Requirements for Cryptographic modules, May 25 2001
FIPS 180-4	Secure Hash Standard (SHS)
FIPS 186-4	Digital Signature Standard (DSS)
FIPS 197	Advanced Encryption Standard
FIPS 198-1	The Keyed-Hash Message Authentication Code (HMAC)
FIPS 202	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions
IG	Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, January 11, 2016.
SP 800-20	Modes of Operation Validation System for Triple Data Encryption Algorithm (TMOVS)
SP 800-38A	Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode
SP 800-38D	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC
SP 800-38F	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping
SP 800-90A	Recommendation for Random Number Generation Using Deterministic Random Bit Generators

Table 14 – References and Standards

4.2 Acronyms

The following table defines acronyms found in this document:

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CBC	Cipher-Block Chaining
CMAC	Cipher-based Message Authentication Code
CMVP	Crypto Module Validation Program
CO	Cryptographic Officer
CPU	Central Processing Unit
CSP	Critical Security Parameter
CTR	Counter-mode
CVL	Component Validation List
DES	Data Encryption Standard
DRAM	Dynamic Random Access Memory
DRBG	Deterministic Random Bit Generator
EC	Elliptic Curve
ECB	Electronic Code Book
ECDSA	Elliptic Curve Digital Signature Authority
FIPS	Federal Information Processing Standards
GCM	Galois/Counter Mode
GPC	General Purpose Computer
HMAC	Key-Hashed Message Authentication Code
IG	Implementation Guidance
IV	Initialization Vector
KAT	Known Answer Test
MAC	Message Authentication Code
MD5	Message Digest algorithm MD5
N/A	Non Applicable
NDRNG	Non Deterministic Random Number Generator
OS	Operating System
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
TDES	Triple Data Encryption Standard

Table 15 – Acronyms and Definitions