Security Builder® Linux Kernel Cryptographic Module

Version 1.0

FIPS 140-2 Non-Proprietary
Security Policy

Certicom Corp.

January 14, 2016

Contents

1 Introduction

1.1 Overview

This is a non-proprietary Federal Information Processing Standard (FIPS) 140-2 Security Policy for Certicom's **Security Builder® Linux Kernel Cryptographic Module** v1.0. This module is a software-only external Linux kernel module that provides general-purpose cryptographic services to the remainder of the kernel. This Security Policy specifies the rules under which the Module must operate. These security rules are derived from the requirements of FIPS 140-2 [1], and related documents [6, 7, 8].

1.2 Purpose

This Security Policy is created for the following purposes:

1. It is required for FIPS 140-2 validation.
2. To outline the Security Builder® Kernel Cryptographic Module's conformance to FIPS 140-2 Level 1 Security Requirements.
3. To provide users with how to configure and operate the cryptographic module in order to comply with FIPS 140-2.

1.3 References

References

[1] NIST *Security Requirements For Cryptographic Modules, FIPS PUB 140-2,* December 3, 2002.

[2] NIST *Security Requirements For Cryptographic Modules, Annex A: Approved Security Functions for FIPS PUB 140-2*, Draft, October 8, 2014.

[3] NIST *Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2*, Draft, August 12, 2011.

[4] NIST *Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2*, Draft, February 16, 2012.

[5] NIST *Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2*, Draft, October 8, 2014.

[6] NIST *Derived Test Requirements for FIPS 140-2*, Draft, January 4, 2011.

[7] NIST *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program,* August 7, 2015.

[8] NIST *Frequently Asked Questions for the Cryptographic Module Validation Program*, June 5, 2014.

1.4 Change History

Change history is recorded in Table 1.

Table 1: Change History

Revision	Date	Author	Description
0.1	2015/01/27	H.W.	Initial revision.
0.2	2015/04/29	H.W.	Add XTS, CCM, GCM, DRBG related stuffs.
0.3	2015/05/11	R.T.	Modifications after initial review.
0.4	2015/06/02	H.W.	Add description for module logs.
0.5	2015/06/12	H.W.	Change OS from Ubuntu 14.10 to CentOS 7
0.6	2015/07/08	H.W.	Move some details into Design doc
0.7	2015/07/09	H.W.	Add CAVP validation numbers
1.0	2015/07/10	R.T.	Modified after discussion with lab.
1.0.1	2015/08/13	H.W.	Drop GCM and some minor editorial changes after Functional Test
1.0.2	2016/01/14	H.W.	Address CMVP comments

2 Cryptographic Module Specification

The Security Builder® Linux Kernel Cryptographic Module is a multiple-chip standalone software cryptographic module in the form of an object that operates with the following components:

- Commercially available general-purpose computer and smartphone hardware.
- Commercially available Operating System (OS) that runs on the computer and smartphone hardware.

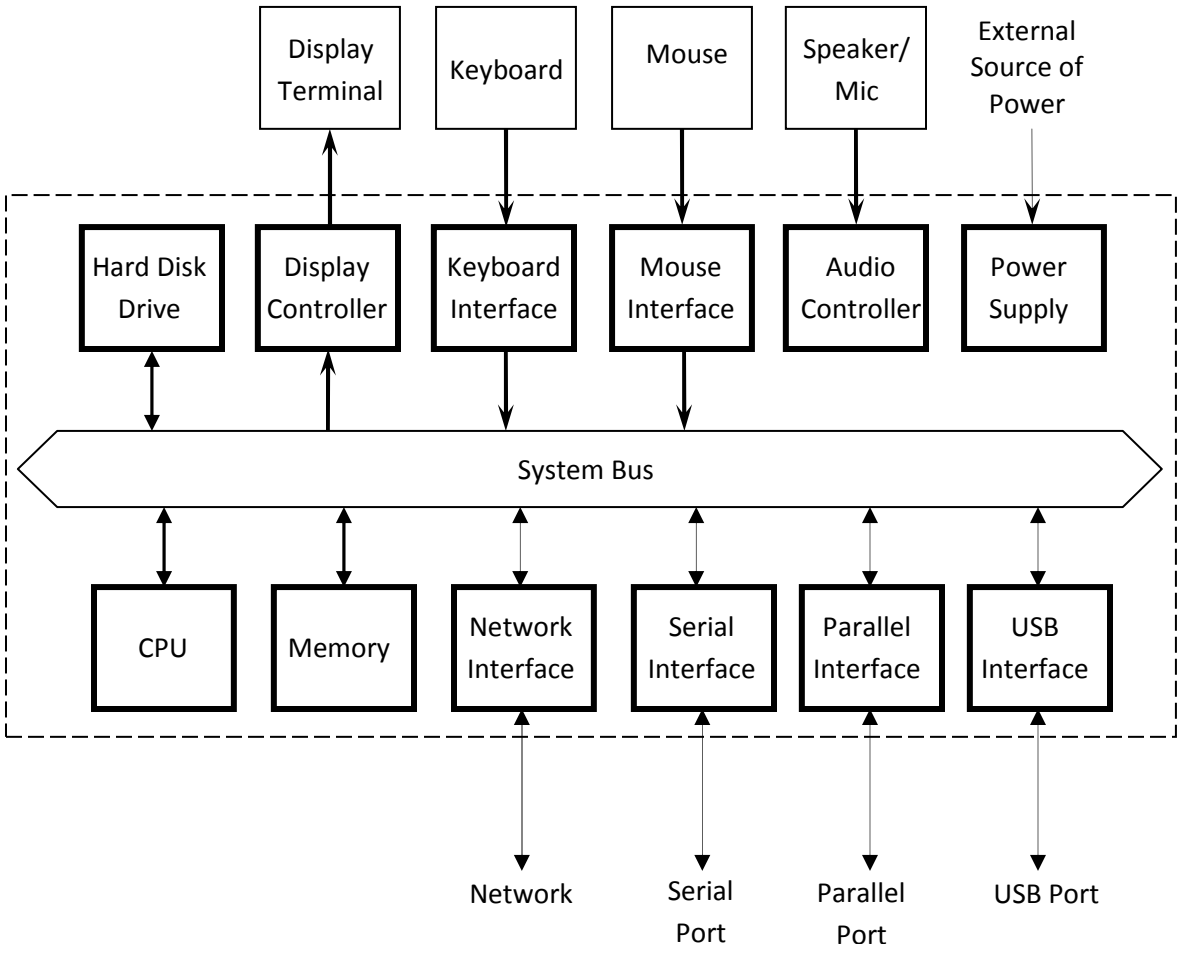
2.1 Physical Specifications

The general-computer hardware component consists of the following devices:

1. CPU (Microprocessor)
2. Memory
    (a) Working memory is located on the RAM containing the following spaces:
        i. Input/output buffer
        ii. Plaintext/ciphertext buffer
        iii. Control buffer

        Key storage is not deployed in this module.
    (b) Program memory is also located on RAM.
3. Hard Disk (or disks)
4. Display Controller, including Touch Screen Controller
5. Keyboard Interface
6. Mouse Interface, including Trackball Interface
7. Audio Controller
8. Network Interface
9. Serial Interface
10. Parallel Interface
11. USB Interface
12. Power Supply

The configuration of this component is illustrated in Figure 1.



: Cryptographic Boundary

: Flow of data, control input, and status output

: Flow of control input

: Flow of status output

Figure 1: Cryptographic Module Hardware Block Diagram

2.2 Hardware and OS

The combinations of computer hardware and OS include the following representative platforms.

1. Android 5.1 64-bit running on a Qualcomm Snapdragon MSM8992 development device
2. Centos 7 64-bit running on a Kontron NSN2U IP Network Server

2.3 Software Specifications

The Security Builder® Kernel Cryptographic Module is manufactured by Certicom Corp. It is a software only security level 1 cryptographic module that provides general-purpose cryptographic services to the remainder of the Linux kernel. The binary of the module is in the Linux kernel's external module format.

The interface into the Security Builder® Kernel Cryptographic Module is via Application Programmer's Interface (API) function calls. These function calls provide the interface to the cryptographic services, for which the parameters and return codes provide the control input and status output (see Figure 2).

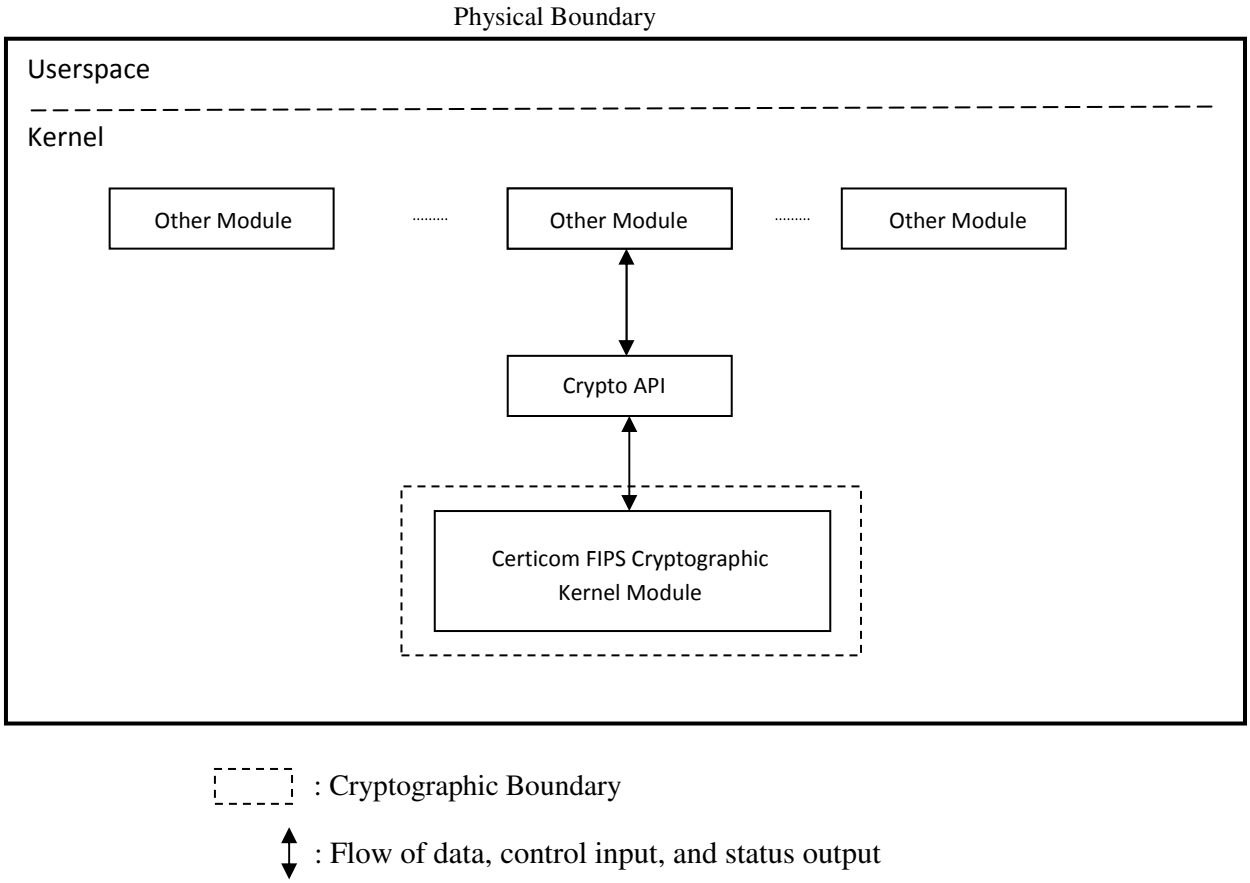


Figure 2: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The ports and interfaces are summarized in Table 2.

Table 2: Ports and Interfaces

Interface	Port
Data Input	API input parameters
Data Output	API output parameters
Control Input	API calls, module parameters
Status Output	API return code, console, Kernel log ring buffer
Power Input	API Initialization function
Maintenance	Not supported

The Security Builder® Kernel Cryptographic Module is a normal Kernel external module aggregating a collection of cryptographic algorithms. Security Builder® Kernel Cryptographic Module does not make any changes to the API of using these algorithms.

For API usage, please refer to the publicly available Kernel Cryptographic API documents or the Linux kernel document under the kernel source tree (Documentation/crypto/api-intro.txt). Also many examples are available in the regression test module (tcrypt.c) in the kernel source.

Besides Kernel Cryptographic API, the Security Builder® Kernel Cryptographic Module provided extra functions to do on-demand self-tests and to return module status, which are described in the Crypto Officer and User Guide in Appendix A.

4 Roles, Services, and Authentication

4.1 Roles

The Security Builder® Kernel Cryptographic Module supports Crypto Officer and User Roles (see Table 3). These roles are enforced by this Security Policy.

Table 3: Roles and Services

Service	Crypto Officer	User
Initialization, etc.		
Initialization	X	
Deinitialization	X	
Self-tests	X	X
Show status	X	X
Key and CSP zeroization	X	X
Symmetric Ciphers (AES and Triple-DES)		
Encrypt	X	X
Decrypt	X	X
Hash Algorithms and Message Authentication (SHA, HMAC)		
Hashing	X	X
Message Authentication	X	X
Random Number Generation (DRBG)		
Instantiation	X	
Seeding	X	X
Request	X	X

In order to operate the module securely, it is the Crypto Officer and User's responsibility to confine calls to those methods that have been FIPS 140-2 Approved. Thus, in the approved mode of operation, all Roles shall confine themselves to calling FIPS Approved algorithms, as marked in Table 4.

4.2 Services

The set of cryptographic algorithms supported by Certicom FIPS kernel Module is given in Table 4.

Table 4: Supported Algorithms and Standards

	Algorithm	FIPS Approved or Allowed	Cert Number
Block Ciphers	DES		
	Triple-DES (ECB, CBC) [NIST SP 800-67]	X	#1953
	AES (ECB, CBC, CTR, CCM, XTS) [FIPS 197]	X	#3464
	AES(GCM)		#3464
	AES(LRW)		
Hash Functions	SHA-1 [FIPS 180-4]	X	#2859
	SHA-224 [FIPS 180-4]	X	#2859
	SHA-256 [FIPS 180-4]	X	#2859
	SHA-384 [FIPS 180-4]	X	#2859
	SHA-512 [FIPS 180-4]	X	#2859
Message Authentication	HMAC-SHA-1 [FIPS 198-1]	X	#2209
	HMAC-SHA-224 [FIPS 198-1]	X	#2209
	HMAC-SHA-256 [FIPS 198-1]	X	#2209
	HMAC-SHA-384 [FIPS 198-1]	X	#2209
	HMAC-SHA-512 [FIPS 198-1]	X	#2209
Random Number Generation	DRBG [NIST SP 800-90]	X	#850
	ANSI X9.31 [ANSI X9.31]		#1383

On hardware platform Centos 7 64-bit running on a Kontron NSN2U IP Network Server, AES (ECB, CTR, CCM, and XTS modes) is tested with and without AES-NI instructions. AES (LRW) is only available with AES-NI instructions.

The 3-key Triple-DES, AES (ECB, CTR, CCM, and XTS modes), SHS (SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512), HMAC-SHS (HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA256, HMAC-SHA-384, and HMAC-SHA-512), DRBG (NIST SP 800-90), have been validated to comply with FIPS. In order to operate the module in compliance with FIPS, only these FIPS Approved or allowed algorithms should be used.

The AES (LRW, GCM), ANSI X9.31 Random Number Generation and DES are supported as non-FIPS Approved algorithms. In order to operate the module in compliance with FIPS, these algorithms should not be used.

Table 5 summarizes the keys and CSPs used in the FIPS mode.

Table 5: Key and CSP, Key Size, Security Strength, and Access

Algorithm	Key and CSP	Key Size	Strength	Access
AES	key	128, 192, 256 bits	128, 192, 256 bits	Read, Use
Triple-DES	key	168 bits	112 bits	Read, Use
HMAC	key	160-512 bits	160-512 bits	Use
DRBG	seed	192-384 bits	128-256 bits	Use

Note:
HMAC-SHA-1 shall have a key size of at least 112 bits

4.3 Operator Authentication

Certicom FIPS Kernel Module does not deploy authentication mechanism. The roles of Crypto Officer and User are implicitly selected by the operator.

5 Finite State Model

The Finite State model contains the following states:

- Unloaded/Uninitialized
- Initialized
- Self-Test
- Idle
- Crypto Officer/User
- Error

The following is the important features of the state transition:

1. Before the module is installed by the Crypto Officer, the module is in the Unloaded/Uninitialized state.

2. When the initialization command is applied to the module, i.e., the module is loaded in memory, it switches to the Initialization state. Then, it transits to the Self-Test state automatically without any input, running the Power-up Tests. On success the module enters Idle; on failure the module enters Error and the module is unloaded. From the Error state the Crypto Officer may need to re-install to attempt correction.

3. From the Idle state (which is only entered if self-tests have succeeded), the module can transit to the Crypto Officer/User state when an API function is called.

4. When the API function has completed successfully, the state transits back to Idle.

5. If the Conditional Test (Continuous Random Number Generation Test or Pair-wise Consistency Test) fails, the state transits to Error state and the module is disabled.

6. When On-demand Self-test is executed, the module enters the Self-Test state. On success the module enters Idle; on failure the module enters Error and the module is disabled.

7. When the unload/de-initialize command is executed, the module goes back to the Unloaded/Uninitialized state.

8. In the disabled state, the module can no longer be used.

6 Physical Security

Physical security is not applicable to this software module at Level 1 Security.

7 Operational Environment

This module is designed for commercially available general purpose computer or smartphone operating systems such as Linux and Android. These operating systems provide modifiable environment. This module is to be run in single user operational environment.

8 Cryptographic Key Management

The Security Builder® Kernel Cryptographic Module does not provide any key generation service. Keys can be established externally and passed into the module via API. The operating system protects unauthorized access to the keys and CSPs in the address space of the module process.

8.1 Random Number Generation

The Security Builder® Kernel Cryptographic Module provides FIPS Approved random number generator, DRBG (Hash, HMAC and CTR). The user needs to provide the seed, and must ensure that the seed is consistent with FIPS 140-2 requirement.

8.2 Key Generation

The cryptographic module does not perform key generation. The module implements a compliant NIST SP 800-90A DRBG which is exclusively used to generate random bits which are used by the calling application.

8.3 Key Entry and Output

The Security Builder® Kernel Cryptographic Module does not support manual key entry and key output. Keys and other CSPs can only be exchanged between the module and the calling application via API parameters.

8.4 Key Storage

The Security Builder® Kernel Cryptographic Module does not provide persistent key storage.

8.5 Zeroization Procedure

Kernel function kzfree() is used to de-allocate internal and intermediate generated CSPs in memory. This function guarantees zeroization occurs during de-allocatio. All keys and CSPs can be zeroized by powering off the module and performing a system reset by the operational environment.

9 Self-Tests

If any failure of self-test is detected, the module will enter the Error state and the module will be disabled. No cryptographic operations can be performed by the module. See Crypto Officer and User Guide in Appendix A.3 for recovery steps.

9.1 Power-up Tests

9.1.1 Tests upon Power-up

Self-tests are initiated automatically when the module is loaded. If the self-tests fail, the module will not be loaded and an error indicator will be output. No cryptographic operations can be performed by the module. If the self-tests pass, the module will be loaded and enabled without output.

The following tests are applied:

1. **Known Answer Tests (KATs):**
   KATs are performed on
   - AES(CBC, ECB, CTR, XTS, CCM) encryption and decryption
   - Triple-DES(CBC, ECB) encryption and decryption
   - 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
   - ANSI X9.31 RNG
   - DRBG(CTR, HASH, HMAC)

2. **Software Integrity Test:**
   The software integrity test deploys SHA-256-HMAC to verify the integrity of the module.

9.1.2 On-Demand Self-Tests

On-demand self tests may be invoked by the Cryptographic Officer or User by invoking a function, which is described in the Crypto Officer and User Guide in Appendix A.

9.2 Conditional Tests

The Continuous Test is executed on all RNG (DRBG and ANSI X9.31 RNG) generated data, examining each requested random generation for repetition. This ensures that the DRBG is not stuck at any constant value.

9.3 Critical Function Tests

For DRBG (CTR, HASH, and HMAC), the module implements following critical function tests:

- SP 800-90 DRBG Instantiate Health Test
- SP 800-90 DRBG Generate Health Test
- SP 800-90 DRBG Reseed Health Test
- SP 800-90 DRBG Uninstantiate Health Test

10 Design Assurance

10.1 Configuration Management

A configuration management system for the cryptographic module is employed and has been described in a document to the testing laboratory. It uses Subversion (SVN) to track the configurations.

10.2 Delivery and Operation

Please refer to Section A.1 of Crypto Officer and User Guide in Appendix A to review the steps necessary for the secure installation and initialization of the cryptographic module.

10.3 Development

Detailed design information and procedures have been described in documentation submitted to the testing laboratory. The source code is fully annotated with comments, and is also submitted to the testing laboratory.

10.4 Guidance Documents

Crypto Officer Guide and User Guide is provided in Appendix A. This appendix outlines the operations for Crypto Officer and User to ensure the security of the module.

11 Mitigation of Other Attacks

No other attacks are mitigated.

Appendix A Crypto Officer And User Guide

A.1 Installation

In order to carry out a secure installation of the Security Builder® Kernel Cryptographic Module, the Crypto Officer must follow the procedure described in this section.

A.1.1 Installing

The Crypto Officer is responsible for the installation of the Security Builder® Kernel Cryptographic Module. Only the Crypto Officer is allowed to install the product.

Place the object in an appropriate location on the computer hardware.

A.1.2 Uninstalling

Remove the object from the computer hardware.

A.2 Commands

A.2.1 Load/Initialization

insmod fipsm.ko
This root command loads Certicom FIPS kernel module into memory and triggers a series of initialization and self-tests on the module. These tests examine the integrity of the object, and the correct operation of the cryptographic algorithms. If these tests are successful, the module will be enabled, otherwise, the module will output an error message saying the module loading is failed and the module will be unloaded.

A.2.2 Unload/De-initialization

rmmod fipsm
This root command will de-initialize the module and unload it from memory. In some cases (platform dependent), the module may be used by other kernel modules and `rmmod` command could fail. At this time, a system reboot is required to unload/de-initialize the module.

A.2.3 Self-Tests

int fipsmod_selftest(void)
This function runs a series of self-tests, and return zero if the tests are successful. These tests examine the integrity of the object, and the correct operation of the cryptographic algorithms. If these tests fail, the function will return a non-zero error code and the module will be disabled. Meanwhile, a message "fipsm: module disabled" can be seen from kernel log dmesg. Section A.3 of this document describes how to recover from the disabled state.

A.2.4 Show Status

int fipsmod_get_state(void)
This function will return the current state of the module. The state value could be:

```
#define FIPSMOD_STATE_ENABLED      1
#define FIPSMOD_STATE_DISABLED     2
```

A.2.5 Module logs

insmod fipsm.ko v=1
Certicom FIPS kernel module can write logs into Kernel's log ring buffer. Users can use dmesg to view the logs. Certicom FIPS kernel module has a module parameter v (0-5) to control verbose level of the logs. If v is not specified or v=0, log will be turned off.

A.3 When Module is Disabled

When the Security Builder® Kernel Cryptographic Module becomes disabled, attempt to bring the module back to the Initialized state by unloading and reloading the module. If the initialization is successful, the module is recovered. If this recovery attempt fails, it indicates a fatal error. Please contact Certicom Support immediately.