Samsung OpenSSL Cryptographic Module

FIPS 140-2 Security Policy

Version 1.4

Last Update: 2013-04-10

1. Introduction

This document is a non-proprietary FIPS 140-2 Security Policy for the Samsung OpenSSL Cryptographic Module. It contains a specification of the rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 multi-chip standalone software module.

1.1. Purpose of the Security Policy

There are three major reasons that a security policy is required:

- it is required for FIPS 140-2 validation,
- it allows individuals and organizations to determine whether the cryptographic module, as implemented, satisfies the stated security policy, and
- it describes the capabilities, protection, and access rights provided by the cryptographic module, allowing individuals and organizations to determine whether it will meet their security requirements.

1.2. Target Audience

This document is intended to be part of the package of documents that are submitted for FIPS validation. It is intended for the following people:

- Developers working on the release
- FIPS 140-2 testing lab
- Crypto Module Validation Program (CMVP)
- Consumers

2. Cryptographic Module Specification

This document is the non-proprietary security policy for the Samsung OpenSSL Cryptographic Module, and was prepared as part of the requirements for conformance to Federal Information Processing Standard (FIPS) 140-2, Level 1.

The following section describes the module and how it complies with the FIPS 140-2 standard in each of the required areas.

2.1. Description of Module

The Samsung OpenSSL Cryptographic Module is a software only security level 1 cryptographic module that provides general-purpose cryptographic services to the applications. The crypto module runs on an ARM processor.

The following table shows the overview of the security level for each of the eleven sections of the validation.

Security Component	Security 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
EMI/EMC	3
Self Tests	1
Design Assurance	3
Mitigation of Other Attacks	N/A

Table 1: Security Levels

The module has been tested on the following platform:

Module/Implementation	Device	O/S & Ver.
Samsung OpenSSL Cryptographic Module (SFOpenSSL1.0.0e-1.1)	Galaxy S3	Android Ice-cream Sandwich 4.0

Table 2: Tested Platform

2.2. Description of FIPS Approved and Non-FIPS Approved Mode

By default, upon initialization, the module performs self-tests and enters the “Non-FIPS” mode. Whenever the external application requires “FIPS-Approved” mode, it needs to invoke all the self-tests by calling FIPS_mode_set(FIPS_Mode). Please note that the self tests invoked before entering FIPS or Non-FIPS mode are described in Section 10.1.

The module can be switched between FIPS-Approved and Non-FIPS mode by invoking the API FIPS_mode_set() using the following parameters:

FIPS_MODE = 1

NOT_IN_FIPS_MODE = 2

In FIPS-Approved mode, the module will be initialized with symmetric algorithms, digest algorithms, HMAC, and random generators.

In the Approved mode the module provides the following approved functions:

- AES (CBC, ECB, CFB with 8 bits, CFB with 128 bits, OFB)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512
- RNG (ANSI X9.31)
- Triple-DES (CBC, ECB, CFB with 8 bits, CFB with 64 bits, OFB)
- HMAC (with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)
- RSA
- DSA

CAVEATS:

1) The strength of AES encryption keys is between 128 and 192 bits
2) This module generates cryptographic keys whose strength is modified by available entropy – 192 bits

Please see Table 5, “Services” in Section 4.2 for the CAVP certificate numbers.

The module implements the following Non-Approved algorithms, which shall not be used in the FIPS 140-2 approved mode of operation. When invoking one of the non-approved ciphers which are still technically callable as listed above, the module implicitly transitions into non-FIPS mode.

- Blowfish
- Triple-DES-CTR (non compliant)
- AES-CTR (non compliant)
- MD5
- IDEA
- RC2
- RC4
- Diffie-Hellman
- md_rand.c (Non Approved RNG)

In the Non-FIPS mode, all the above listed approved and non-approved algorithms except the approved RNG ANSI X9.31 are available for use, although MD5 is allowed for use in TLS only. The non-approved RNG ms_rand.c is available only in the Non-FIPS mode for usage.

2.3. Cryptographic Module Boundary

2.3.1. Software Block Diagram

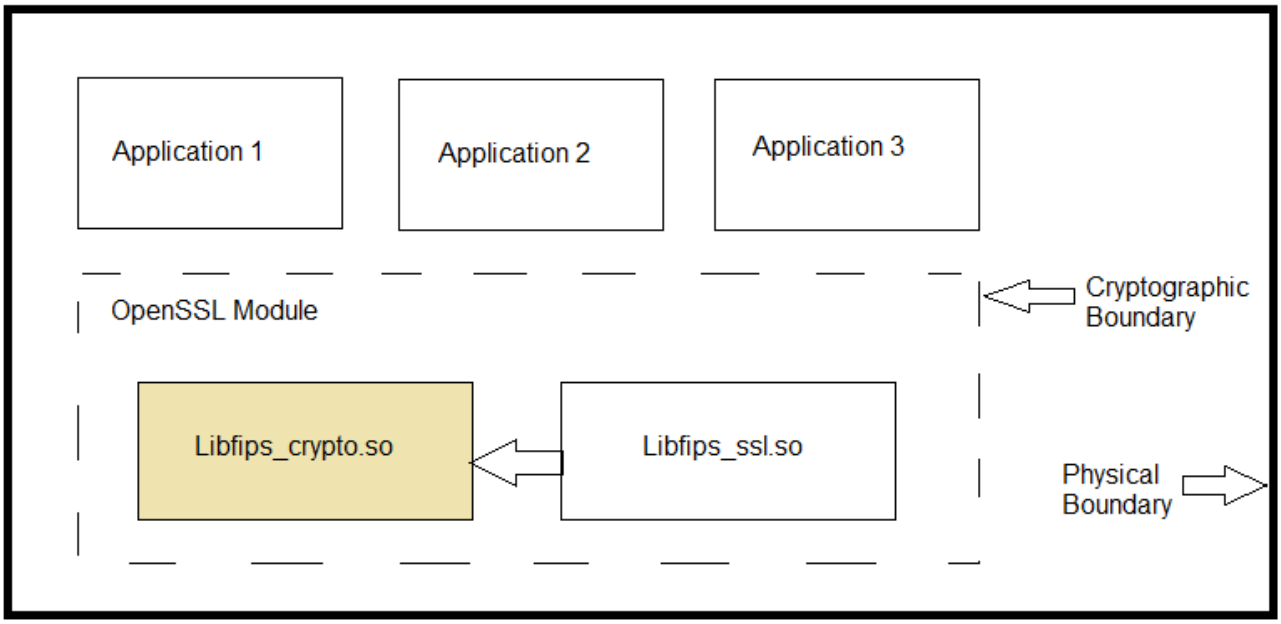

Figure 1: Software Block Diagram

Related documentation:

- S/W Detailed Level Design (FIPS_OpenSSL_Func_Design.docx) v.13
- Samsung OpenSSL Cryptographic Module (Samsung_OpenSSLv1.4.doc)

Note: The master component list is provided in Section 6.3 of S/W Detailed Level Design document.

2.3.2. Hardware Block Diagram

This figure illustrates the various data, status and control paths through the cryptographic module. Inside, the physical boundary of the module, the mobile device consists of standard integrated circuits, including processors and memory. These do not include any security-relevant, semi- or custom integrated circuits or other active electronic circuit elements. The physical boundary includes power inputs and outputs, and internal power supplies. The logical boundary of the cryptographic module contains only the security-relevant software elements that comprise the module.

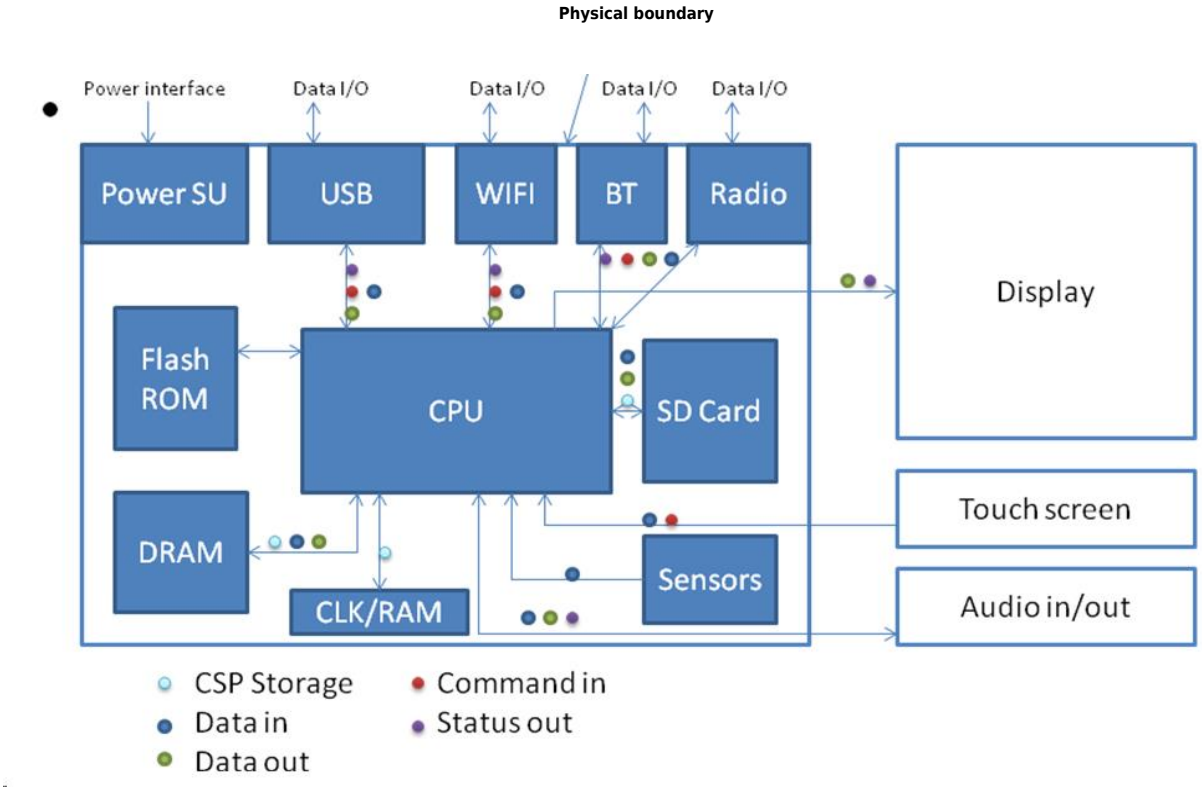

Figure 2: Hardware Block Diagram

3. Cryptographic Module Ports and Interfaces

FIPS Interface	Ports
Data Input	API input parameters
Data Output	API output parameters
Control Input	API function calls
Status Output	API function calls, or configuration files on filesystem
Power Input	Physical power connector

Table 3: Ports and Interfaces

4. Roles, Services and Authentication

4.1. Roles

Role	Services (see list below)
User	Encryption, Decryption, Random Numbers, Digest Creation, Key Generation, Signature Generation, Signature Verification
Crypto Officer	Configuration, Encryption, Decryption, Random Numbers, Initialization of Module, Digest Creation, Key Generation, Signature Generation, Signature Verification

Table 4: Roles

The module meets all FIPS 140-2 level 1 requirements for Roles and Services, implementing both User and Crypto Officer roles. 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. No further authentication is required. The Crypto Officer can initialize the Module.

4.2. Services

Role	Service	CSP	Modes	FIPS Approved (Cert #)	Access (Read, Write, Execute)
User, Crypto Officer	AES encryption and decryption	128, 192, 256 bit keys	ECB, CBC, OFB, CFB,	Cert #2108	R, W, EX
Crypto Officer, User	HMAC (with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)	HMAC Key	N/A	Cert #1282	R, W, EX
User, Crypto Officer	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512	N/A	N/A	Cert #1831	R, W, EX
User, Crypto Officer	Triple-DES	2 Key & 3 Key	CBC, ECB, OFB, CFB	Cert #1343	R, W, EX
User, Crypto Officer	RSA	1024, 2048 bit keys	N/A	Cert #1082	R, W. EX
User, Crypto Officer	DSA	1024, 2048 bit keys	N/A	Cert #658	R, W, EX
User, Crypto Officer	RNG ANSI X9.31	Seed Key	AES-128, AES-192, AES-256	Cert #1083	R, W, EX
Crypto Officer	Initialization	N/A	N/A	N/A	N/A

Role	Service	CSP	Modes	FIPS Approved (Cert #)	Access (Read, Write, Execute)
User, Crypto Officer (self test is executed automatically when device is booted or restarted)	Self Test	N/A	N/A	N/A	N/A
User, Crypto Officer	Check Status/Get State	N/A	N/A	N/A	R
Crypto Officer	Configuration	N/A	N/A	N/A	R, W, EX
User, Crypto Officer	Key Generation	N/A	N/A	N/A	R, W, EX
User, Crypto Officer	Zeroization of Asymmetric Keys	RSA/DSA Keys	N/A	N/A	R, W, EX

Table 5: Services

4.3. Operator Authentication

There is no operator authentication; assumption of role is implicit by action.

4.4. Mechanism and Strength of Authentication

No authentication is required at security level 1; authentication is implicit by assumption of the role.

5. Finite State Machine

For information pertaining to the Finite State Model, please refer to the Functional Design document.

6. Physical Security

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

7. Operational Environment

This module will operate in a modifiable operational environment per the FIPS 140-2 definition.

7.1. Policy

The operating system shall be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded).

The external application that makes calls to the cryptographic module is the single user of the cryptographic module, even when the application is serving multiple clients.

8. Cryptographic Key Management

8.1. Random Number Generation

The Module employs an ANSI X9.31 compliant random number generator for creation of cryptographic keys and CSPs. For more details on the RNG please refer to the Functional Design document.

Caveat: The encryption strength of AES keys are modified by available entropy of seeds that are provided to the RNG. The RNG uses seed source from dev/random which provides entropy of 192 bits. Therefore, the maximum encryption strength of AES keys is 192 bits.

8.2. Key Entry and Output

The module does not support manual key entry or key output. Keys or other CSPs can only be exchanged between the module and the calling application using appropriate API calls.

8.3. Key Storage

Keys are not stored inside crypto module. A pointer to plaintext key is passed through. Intermediate/temporary key storages are immediately zeroized.

8.4. Zeroization Procedure

In order to zeroize, keys and other CSPs free() is called which in turn calls another API function that overwrites the memory with an algorithm that depends on the pointer to the value, but not the value itself.

In regards to key generation, the external application must call the respective zeroization functions. Intermediate key storages are immediately assigned to Zero. For more details on zeroization and related APIs, please refer to the Functional Design document.

9. Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

Lab Name: PC Engineering Laboratory, Inc
FCC Registration: #90864

For information related to FCC ID of the devices, please refer to the Functional Design document.

10. Self Tests

The module performs an integrity test, as well as known answer tests upon initialization of the module and before the module becomes usable. If self tests fail, the module is in error state. In error state, no cryptographic operation is allowed, except the invocation of on demand self test.

Self test consists of the following tests:

10.1. Power-Up Tests

The module performs all power-up self tests when entering Non-FIPS mode. Upon initialization and successful completion of power on self tests, the module then enters Non-FIPS mode. In order to switch to FIPS-Approved mode, the module performs the integrity test and all known answer tests (KATs) by invoking the API, FIPS_mode_set(FIPS_MODE). Upon successful execution of KATs of SHA-1, HMAC, PRNG, AES, TDES, RSA, and DSA, the module will be in FIPS-Approved mode. Please note that a DSA pair-wise consistency test is implemented instead of a KAT as a part of the power on self tests.

The module performs the integrity test, as well as all the KATs in both FIPS-Approved and Non-FIPS mode. The module enters error state if the conditional tests fail in either mode.

10.1.1. Cryptographic algorithm tests (Known Answer Tests)

Cryptographic algorithm test using a known answer will be conducted for all cryptographic functions (e.g., encryption, decryption, and random number generation) of each Approved cryptographic algorithm implemented by the module in FIPS-Approved mode.

Algorithm	**Test**
AES (encryption/decryption)	KAT
Triple-DES (encryption/decryption)	KAT
RSA (signature generation/verification)	KAT
DSA (signature generation/verification)	Pair-wise consistency test
PRNG	KAT
HMAC-SHA-1	KAT
HMAC-SHA-224	KAT
HMAC-SHA-256	KAT
HMAC-SHA-384	KAT
HMAC-SHA-512	KAT
SHA-1	KAT
SHA-224	Tested as part of HMAC-SHA-224
SHA-256	KAT
SHA-384	Tested as part of HMAC-SHA-384
SHA-512	KAT

Table 6: Power-Up Tests

The module implements a separate Known Answer Test (KAT) for each of the following operations separately: AES encryption, AES decryption, Triple-DES encryption, Triple-DES decryption, RSA signature generation, and RSA signature verification.

10.1.2. Integrity test

The module's integrity test is performed using HMAC-SHA-256.

- Build Time
  - HMAC-SHA-256 calculated on libfips_crypto.so (dynamic library) file
    - HMAC appended to libfips_crypto.so file
- Run Time
  - libfips_crypto.so is read as a file
  - When algorithm self tests are completed, integrity test routine is called
    - Perform HMAC-SHA-256 on the read libfips_crypto.so value in ram
    - Read stored HMAC located after libfips_crypto.so (last 32 bytes)
    - If calculated and stored values do not match, set error state, FIPS_R_FINGERPRINT_DOES_NOT_MATCH and the system property as "error_integrity"

10.2. Conditional Tests

Algorithm	**Test**
DSA	Key generation, Pair-wise consistency test
RSA	Key generation, Pair-wise consistency test
PRNG	Continuous test

Table 7: Conditional Tests

10.2.1. Pair-wise consistency test

A pair-wise consistency test must be conducted for every key generation.

The module implements RSA and DSA pair-wise consistency tests during key generation. If the test fails, it updates the FIPS status to error.

10.2.2. Continuous random number generator (CRNG) test

The continuous random number generator test implemented in the module is as follows: the CRNG test consists of a priming test which is implemented by using the very first RNG value to initialize the comparing value, discarding the RNG value and obtaining the next round of the RNG for output to the caller. The module performs the CRNG test every time the random number generation

service is invoked. The very first random number is stored and it is compared against the second random number. If the two random numbers are the same, then the CRNG test fails and the module enters an error state. If the two random numbers are different, the CRNG test passes.

11. Design Assurance

11.1. Configuration Management

All source code is maintained in internal source code servers and the tools, Perforce and SVN, are used as code control. Perforce is used for commercial products and SVN is used for in-development projects. Release is based on the Change List number, which is auto-generated. Every check-in process creates a new change list number.

Versions of controlled items include information about each version. For documentation, revision history inside the document provides the current version of the document. Version control maintains the all the previous version and the version control system automatically numbers revisions.

For source code, unique information is associated with each version such that source code versions can be associated with binary versions of the final product.

11.2. Delivery and Operation

The Crypto module is never released as Source code. The module sources are stored and maintained at a secure development facility with controlled access.

The development team and the manufacturing factory share a secured internal server for exchanging binary software images. The factory is also a secure site with strict access control to the manufacturing facilities. The module binary is installed on the mobile devices (phone and tablets) using direct binary image installation at the factory. The mobile devices are then delivered to mobile service operators. Users cannot install or modify the module. The developer also has the capability to deliver software updates to service operators who in turn can update end-user phones and tablets using Over-The-Air (OTA) updates. Alternatively, the users may bring their mobile devices to service stations where authorized operators may use developer-supplied tools to install software updates on the phone. The developer vets all service providers and establishes secure communication with them for delivery of tools and software updates. If the binary is modified by unauthorized entity, the device has a feature to detect the change and thus not accept the binary modified by an unauthorized entity.

12. Mitigation of Other Attacks

No other attacks are mitigated.

13. Glossary and Abbreviations

AES	Advanced Encryption Specification
CAVP	Cryptographic Algorithm Validation Program
CBC	Cypher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cypher Feedback
CC	Common Criteria
CMT	Cryptographic Module Testing
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CVT	Component Verification Testing
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
EAL	Evaluation Assurance Level
FSM	Finite State Model
HMAC	Hash Message Authentication Code
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
NVLAP	National Voluntary Laboratory Accreditation Program
OFB	Output Feedback
O/S	Operating System
PP	Protection Profile
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SLA	Service Level Agreement
SOF	Strength of Function
SSH	Secure Shell
SVT	Scenario Verification Testing
TDES	Triple DES

TOE	Target of Evaluation
UI	User Interface

14. References

[1] FIPS 140-2 Standard, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[2] FIPS 140-2 Implementation Guidance, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[3] FIPS 140-2 Derived Test Requirements, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[4] FIPS 197 Advanced Encryption Standard, http://csrc.nist.gov/publications/PubsFIPS.html

[5] FIPS 180-3 Secure Hash Standard, http://csrc.nist.gov/publications/PubsFIPS.html

[6] FIPS 198-1 The Keyed-Hash Message Authentication Code (HMAC), http://csrc.nist.gov/publications/PubsFIPS.html

[7] FIPS 186-3 Digital Signature Standard (DSS), http://csrc.nist.gov/publications/PubsFIPS.html