Samsung Kernel Cryptographic Module

FIPS 140-2 Security Policy

version 1.5

Last Update: 2013-02-18

1. Introduction

This document is a non-proprietary FIPS 140-2 Security Policy for the Samsung Kernel 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 Kernel 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 Kernel Cryptographic Module is a software only security level 1 cryptographic module that provides general-purpose cryptographic services to the remainder of the Linux kernel. The crypto module runs on an ARM 7 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

Security Component	Security Level
Design Assurance	3
Mitigation of Other Attacks	N/A

Table 1: Security Levels

The module has been tested on the following platforms:

Module/Implementation	Processor	Device	O/S & Ver.
Samsung Kernel Cryptographic Module (SKC1.4.1)	ARM 7	Galaxy S3	Android Ice-cream Sandwich 4.0
Samsung Kernel Cryptographic Module (SKC1.4.1)	ARM 7	Galaxy S3	Android Ice-cream Sandwich 4.0
Samsung Kernel Cryptographic Module (SKC1.4.1.1)	ARM 7	Note II	Android Jelly Bean 4.1

Table 2: Tested Platforms

2.2. Description of Approved Mode

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

- AES (CBC, ECB, Counter Mode)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512
- RNG (ANSI X9.31)
- Triple-DES (CBC, ECB)
- HMAC (with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)

Kernel Crypto API implements the following Non-Approved algorithms, which shall not be used in the FIPS 140-2 approved mode of operation:

- DES
- AE-CTR (non compliant)
- Triple-DES-CTR (non compliant)
- Twofish
- AEAD
- MD5
- ansi_cprng
- ARC4
- GHASH (GCM hash)

Warning: The user of AES and Triple-DES counter mode should be aware that the counter size is 32 bit. The counter will roll over after 2^32 blocks of encrypted data. It is estimated to take seven days for AES and 33 days for Triple-DES to finish 2^32 blocks of data on an embedded device with ARM 7 as the CPU. It is the

responsibility of the calling application to refresh the key before the rolling over of the counter takes place.

In view of CTR RFC3686, user must be careful, as a combination of key and counter value is needed for each data block.

2.3. Cryptographic Module Boundary

2.3.1. Software Block Diagram

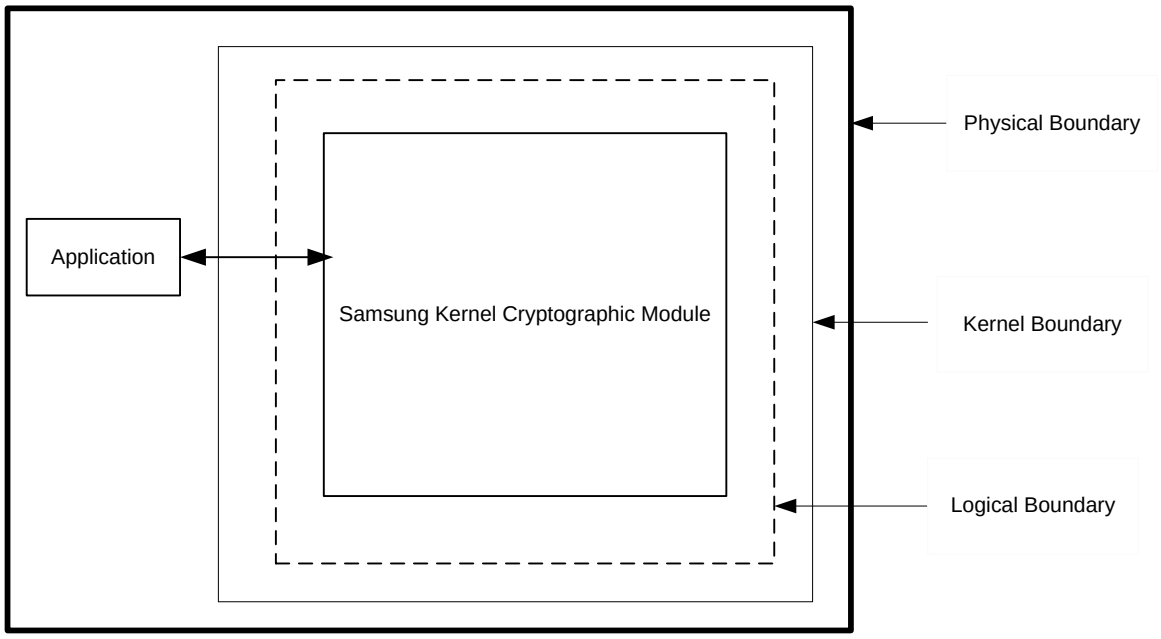

Figure 1: Software Block Diagram

The binary image that contains the Crypto API module for the appropriate platform is as follows:

- zImage (version SKC1.4.1)
- boot.img (version SKC1.4.1, SKC1.4.1.1)

Related documentation:

- S/W Detailed Level Design (FIPS_Crypto_Func_Design.docx) v1.13
- Samsung Kernel Cryptographic Module (SamsungCryptoAPI_SPv1.5.doc)

Note: The master component list is provided in Section 2.10 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.

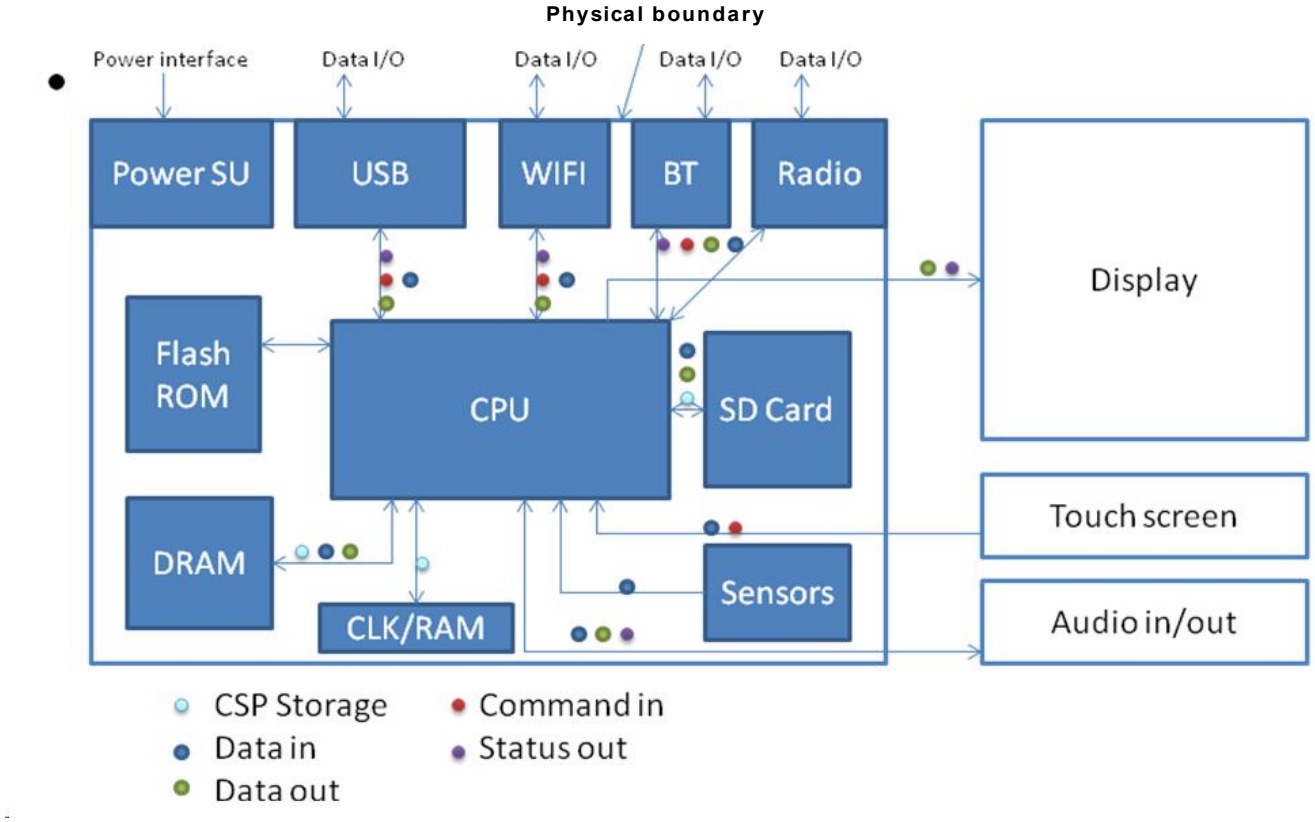

Figure 2: Hardware Block Diagram

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 return codes; kernel log file, /proc/sys/crypto/fips_status, the status of the module is also provided at user interface
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
Crypto Officer	Configuration, Encryption, Decryption, Random Numbers, Initialization of Module, Digest Creation

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

See section 1.2 for the complete list of approved and non-approved cryptographic services. This table only covers the approved services.

Role	Algorithms and Services	CSP	Modes	FIPS Approved (Cert #)	Access (Read, Write, Execute)
User, Crypto Officer	AES encryption and decryption	128, 192, 256 bit keys	ECB, CBC, Counter Mode	(Cert # 2056, 2097, 2141, 2144, 2256)	R, W, EX
Crypto Officer	HMAC (with SHA-1, SHA-224, SHA-256, SHA 384, SHA 512)	HMAC Key	N/A	(Cert # 1251, 1272, 1308, 1311, 1383)	R, W, EX
User, Crypto Officer	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512	N/A	N/A	(Cert #1799, 1821, 1863, 1866, 1943)	R, W, EX
User, Crypto Officer	Triple-DES encryption and decryption	2 Key & 3 Key	CBC, ECB	(Cert #1325, 1334, 1361, 1362, 1411)	R, W, EX
User, Crypto Officer	RNG ANSI X9.31	Seed Key	AES-128	(Cert #1075, 1079, 1096, 1099, 1126)	R, W, EX
Crypto Officer	Initialization	N/A	N/A	N/A	N/A
User, Crypto Officer (self test is execute 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

Table 5: Approved Services

The FIPS 140-2 module covers the static kernel binary and therefore provides a number of non-security services to callers within kernel space as well as user space. The following documents provide a description of these services:

- Unique Android kernel features and resources that list the modifications made to the Linux kernel by the Google Android developers are provided in http://elinux.org/Android_Kernel_Features
- Linux system call API man pages provided in chapter 2 of the Linux man pages obtainable from git://github.com/mkerrisk/man-pages.git

- Linux kernel internals including interfaces between kernel components documented in the book ISBN-13: 978-0470343432

- Linux kernel driver development documentation covering the kernel interfaces available for device drivers: ISBN-13: 978-0596005900

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 keys. Note: the RNG seed is the tuple {V key DT}, where those values are defined in ANSI X9.31 Appendix A.2.4.

The calling user provides the seed and seed key, usually by obtaining bits via get_random_bytes() from the Linux-provided /dev/random pseudo-device. The caller must ensure that the seed and seed key for the DRNG is inserted into the DRNG consistent with FIPS 140-2 requirements, i.e. that they are not identical and that they are not the same values used in the previous call. Failure to comply with this requirement will cause the module to go into an Error state.

Note: Please note that in the current implementation of the Linux kernel the approved RNG is not used for key generation. However, the module is a crypto library and offers the FIPS approved RNG service as one of the cryptographic services to the external applications.

8.2. Key Generation

The module does not provide any key generation service or perform key generation for any of its approved algorithms. Keys are passed in from clients via algorithm APIs. Seeds for key generation inputs to crypto module.

8.3. 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.

8.4. Key Storage

Keys are not stored inside crypto module. A pointer to plaintext key is passed through. Intermediate key storages are immediately assigned to Zero.

8.5. Zeroization Procedure

The zeroization mechanism includes deallocation of CSPs. For more details on zeroization 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

Self test uses the existing Crypto API tcrypt module to perform known-answer self test of algorithms. The module is configured as a built-in kernel module instead of a loadable module as is the case of Linux Crypto API. Tests of all FIPS-approved algorithms are executed. The self tests are run during early-kernel startup when built-in kernel modules are initialized. Self tests can also be invoked by the user by restarting the device. When self tests are done successfully, an indication will be shown in the settings menu. A binary integrity test will then be performed in call from tcrypt. If self test or integrity test fail, an error flag (static variable) is set, the module enters in an error state, and Crypto APIs that return cryptographic information is blocked.

A kernel proc file is set to indicate if device is in FIPS 140-2 approved mode or in error state. The error state flag is used for the value of the process file /proc/sys/crypto/fips_status. Users can check the module status in two ways:

- From the main screen, start the Settings application. Under the Settings menu, go to “About Phone.” Status is displayed in the listings.
- When device encryption is enabled, the Settings application will not be available because a password is required to show the main screen. In such cases, the error status is shown on the password screen itself.

10.1. Power-Up Tests

At module start-up, Known Answer tests are performed. These tests are automatic and do not need operator intervention. If the value calculated and the known answer does not match, the module immediately enters into FIPS_ERR state. Once the module is in FIPS_ERR state, the module becomes unusable via any interface.

Cryptographic algorithm tests (Known Answer Tests):

- AES encryption/decryption

- Triple-DES encryption/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
- Random Number Generator

10.2. Integrity Check

- Build Time
    - SHA-256- HMAC calculated on binary image (compressed kernel) for the appropriate platform.
- Run Time
    - Compressed kernel image copies itself to a different ram location
    - When algorithm self tests are completed, integrity test routine is called
        - Perform Crypto API HMAC-SHA-256 on the appropriate binary depending on the platform in ram
        - Read stored HMAC located after the binary image
        - If calculated and stored values do not match, set error state, FIPS_ERR

10.3. Conditional Tests

A continuous random number generator test is performed during each use of the approved RNG. If values of two consecutive random numbers match, then crypto module goes into error state. A CRNG test is also implemented for the Linux provided /dev/random RNG which is usually used by calling user for seeding the approved RNG.

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.

This crypto module is built-in along with the Linux Kernel. Product that does not need FIPS 140-2 certified cryptographic module may decide to change the build flag CONFIG_CRYPTO_FIPS in Kernel config. 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
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