Samsung Electronics Co., Ltd.

Samsung Kernel Cryptographic Module

Software Version: 2.3

FIPS 140-3 Non-Proprietary Security Policy

Document Version 1.1

Last Update: July 25, 2024

Copyrights and Trademarks

Contents

1. General

This document is a non-proprietary FIPS 140-3 Security Policy for the Samsung Kernel Cryptographic Module. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS 140-3 (Federal Information Processing Standards Publication 140-3) for an overall Security Level 1 module.

ISO/IEC 24759 Section 6. [Number Below]	FIPS 140-3 Section Title	Security Level
1	General	1
2	Cryptographic module specification	1
3	Cryptographic module interfaces	1
4	Roles, services, and authentication	1
5	Software/Firmware security	1
6	Operational environment	1
7	Physical security	N/A
8	Non-invasive security	N/A
9	Sensitive security parameter management	1
10	Self-tests	1
11	Life-cycle assurance	1
12	Mitigation of other attacks	N/A

Table 1. Security Levels.

Purpose of the Security Policy

There are three major reasons for which a security policy is needed:

- It is required for a FIPS 140-3 validation,
- To provide a specification of the cryptographic security that will allow individuals and organizations to determine whether the cryptographic module, as implemented, satisfies the stated security policy.
- To describe to individuals and organizations the capabilities, protection, and access rights provided by the cryptographic module, thereby allowing an assessment of whether the module will adequately serve the individual or organizational security requirements.

Target Audience

This document is a part of the package of documents that is submitted for FIPS 140-3 conformance validation of the module. It is intended for the following people:

- Developers.
- FIPS 140-3 testing laboratory.
- The Cryptographic Module Validation Program (CMVP).
- Administrators of the cryptographic module.
- Users of the cryptographic module.

2. Cryptographic module specification

The following section describes the cryptographic module and how it conforms to the FIPS 140-3 specification in each of the required areas.

Module overview

The Samsung Kernel Cryptographic Module (hereafter referred to as “the module” or SKC) is a software module running on a multi-chip standalone general-purpose computing platform. The module’s software version number is 2.3. The module provides cryptographic services to Kernel through an application program interface (API). The binary image that contains the Samsung Kernel Cryptographic Module for the appropriate platform is **boot.img**.

The module has been tested on the following platforms.

#	Operating System	Hardware Platform	Processor	PAA/Acceleration
1	Android 13 (Linux Kernel 5.15)	Samsung Galaxy S23	Qualcomm Snapdragon 8 Gen 2	With PAA
2	Android 13 (Linux Kernel 5.15)	Samsung Galaxy S23	Qualcomm Snapdragon 8 Gen 2	Without PAA

Table 2. Tested Operational Environments.

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

Modes of operation

When the module starts up, the Self-tests are executed automatically and the module enters the operational state if the self-tests pass. A failure of the self-test invokes a panic and the only way to recover the state is to reboot.

Table 3 lists all Approved security functions of the module, including specific key size(s) employed for approved services, and implemented modes of operation. The module is in the Approved mode of operation when the module utilizes the services that use the security functions listed in the Table 3. The Approved mode of operation is configured in the system by default and can only be transitioned into the non-Approved mode by calling one of the non-Approved services listed in Table 9.

Algorithm Cert	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use/Function
A3242	AES [FIPS 197, SP800-38A]	CBC	128, 192, 256 bits	Data Encryption / Decryption
A3242	AES, [FIPS 197, SP800-38A]	ECB	128, 192, 256 bits	Data Encryption / Decryption
A3242	HMAC, [FIPS 198-1]	HMAC-SHA-1	at least 112 bits	Message Authentication
A3242	HMAC, [FIPS 198-1]	HMAC-SHA2-224	at least 112 bits	Message Authentication
A3242	HMAC, [FIPS 198-1]	HMAC-SHA2-256	at least 112 bits	Message Authentication
A3242	HMAC, [FIPS 198-1]	HMAC-SHA2-384 Note: HMAC-SHA2-384 is not implemented for “with PAA mode”.	at least 112 bits	Message Authentication

Algorithm Cert	Algorithm and Standard	Mode/Method	Description / Key Size(s) / Key Strength(s)	Use/Function
A3242	HMAC, [FIPS 198-1]	HMAC-SHA2-512 Note: HMAC-SHA2-512 is not implemented for “with PAA mode”.	at least 112 bits	Message Authentication
A3242	SHS, [FIPS 180-4]	SHA-1	N/A	Message Digest Note: SHA-1 is not used for digital signature generation
A3242	SHS, [FIPS 180-4]	SHA2-224	N/A	Message Digest
A3242	SHS, [FIPS 180-4]	SHA2-256	N/A	Message Digest
A3242	SHS, [FIPS 180-4]	SHA2-384 Note: SHA2-384 is not implemented for “with PAA mode”.	N/A	Message Digest
A3242	SHS, [FIPS 180-4]	SHA2-512 Note: SHA2-512 is not implemented for “with PAA mode”.	N/A	Message Digest

Table 3. Approved Algorithms.

Algorithm/Function	Use/Function
ESSIV-CBC-AES	Disk encryption/decryption with using AES (CBC) and SHA2-256.
CMAC	Message authentication.
AES-CTR	Symmetric Encryption and Decryption.
AES-XTS	Disk encryption/decryption.

Table 4. Non-approved Algorithms Not Allowed in Approved Mode of Operation.

In addition, the module does not support the following algorithms.

- Non-Approved Algorithms Allowed in Approved Mode of Operation
- Non-Approved Algorithms Allowed in Approved Mode of Operation with No Security Claimed

Cryptographic boundary

The module is defined as a multi-chip standalone software module, with the boundary of the Tested Operational Environment’s Physical Perimeter (TOEPP) being defined as the physical perimeter of the tested platform enclosure around which everything runs. Figure 1 below illustrates a block diagram of a typical GPC and the module’s physical perimeter. The module’s cryptographic boundary consists of all functionalities contained within the module’s compiled source code and comprises the following object files components. The object files are generated from the sources through the kernel build process. The module is intended only for single-process execution.

#	Object file name
1	fips140_integrity.o
2	fips140_post.o
3	fips140_test.o
4	fips140_out.o
5	fips140_3_services.o
6	api.o
7	cipher.o
8	algapi.o
9	scatterwalk.o
10	skcipher.o
11	ahash.o
12	shash.o
13	hmac.o
14	sha1_generic.o
15	sha256_generic.o
16	sha512_generic.o
17	ecb.o
18	cbc.o
19	aes_generic.o
20	aes-ce-core.o
21	aes-ce-glue.o
22	aes-ce.o
23	aes-glue-ce.o
24	sha256-core.o
25	sha256-glue.o
26	sha2-ce-core.o
27	sha2-ce-glue.o
28	sha1-ce-core.o
29	sha1-ce-glue.o

Table 5. The module objects list.

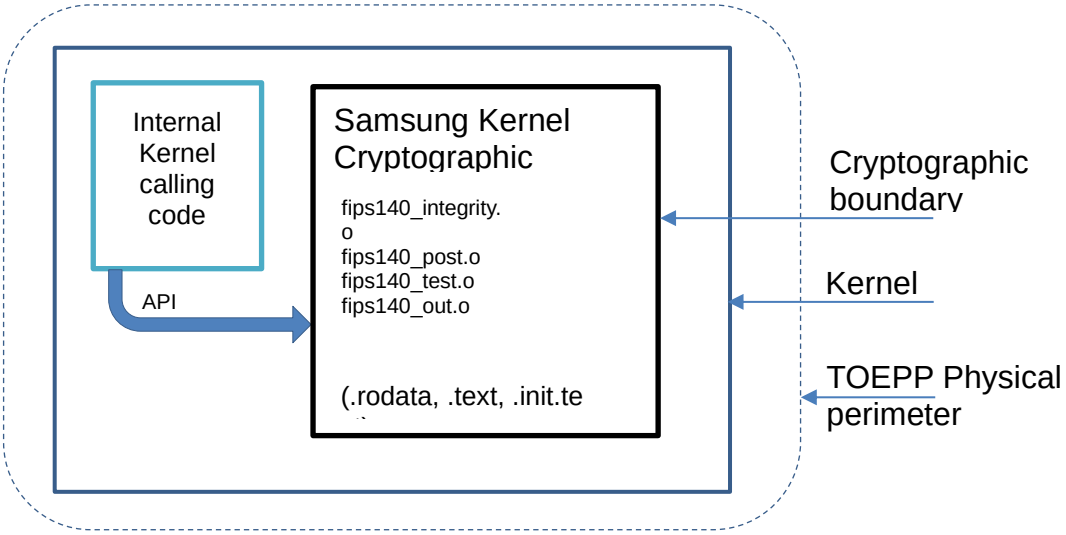

Figure 1. Module's Cryptographic Boundary

3. Cryptographic module interfaces

As a software-only module, the module does not have physical ports. For the purpose of the FIPS 140-3 validation, the physical ports are interpreted to be the physical ports of the hardware platform on which it runs.

The module does not implement a trusted channel.

The logical interfaces are the application program interface (API) through which applications request services. The following table summarizes the logical interfaces.

Physical port	Logical interface	Data that passes over port/interface
N/A	Data input interface	API input parameters
N/A	Data output interface	API output parameters
N/A	Control input interface	API functions calls
N/A	Control output interface	Not applicable
N/A	Status output interface	API return codes, Kernel logs

Table 6. Ports and Interfaces

4. Roles, services and authentication

The module supports Crypto Officer (CO) role. The cryptographic module does not provide any authentication methods. The module does not allow concurrent operators. The Crypto Officer is implicitly assumed based on the service requested. The module does not implement a bypass capability. The module does not implement a self-initiated cryptographic output capability. The module does not support Software/Firmware loading

The module provides the following services to the Crypto Officer.

Role	Service	Input	Output
CO	Symmetric encryption	Key, Plaintext, Mode, Direction, Initial Vector	Cyphertext
CO	Symmetric decryption	Key, Cyphertext, Mode, Direction, Initial Vector	Plaintext
CO	Message digest generation	Message	Message digest
CO	MAC generation	Key, message	Message authentication code
CO	Show status	None	Module status
CO	Show version	None	Module name/ID and versioning information
CO	Module initialization	None	None
CO	Perform self-tests	None	None
CO	Perform zeroization	SSPs	None

Table 7. Roles, Service Commands, Input and Output

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Symmetric encryption	Encrypt a plain text	AES-ECB; AES-CBC;	AES Key	CO	W, E	return value 1
Symmetric decryption	Decrypt a cipher text	AES-ECB; AES-CBC;	AES Key	CO	W, E	return value 1
Message digest generation	Generate message digest	SHA-1; SHA2-224; SHA2-256; SHA2-384; SHA2-512; (Note: SHA2-384 and SHA2-512 are not implemented for “with PAA" mode)	N/A	CO	N/A	return value 1
MAC generation	Generate message authentication code	HMAC-SHA-1; HMAC-SHA2-224; HMAC-SHA2-256; HMAC-SHA2-384; HMAC-SHA2-512; (Note: HMAC-SHA2-384 and HMAC-SHA2-512 are not implemented for “with PAA" mode)	HMAC Key	CO	W, E	return value 1
Show status	Provide Module’s current status (status message)	N/A	N/A	CO	N/A	N/A
Show version	Provide Module’s name and version information	N/A	N/A	CO	N/A	N/A
Module initialization	Initialize the module	N/A	N/A	CO	N/A	N/A
Perform self-tests	Perform self-tests (Pre-operational self-tests and Conditional Self-Tests)	N/A	N/A	CO	Software Integrity Key (non-SSP)	N/A
Perform zeroization	Perform zeroization	N/A	All SSPs	CO	Z	N/A

Table 8. Approved Services

G = Generate: The module generates or derives the SSP.
R = Read: The SSP is read from the module (e.g. the SSP is output).
W = Write: The SSP is updated, imported, or written to the module.
E = Execute: The module uses the SSP in performing a cryptographic operation.
Z = Zeroise: The module zeroises the SSP.

Service	Description	Algorithm Accessed	Role	Indicator
Message authentication Code generation.	Non-approved MAC algorithm. Input: message and key Output: message authentication code.	CMAC	CO	return value 0
Symmetric encryption/decryption.	Non-approved encryption/decryption algorithm. Encrypt input: plaintext, key. Encrypt output: cyphertext. Decrypt input: cyphertext, key. Decrypt output: plaintext.	AES-CTR, ESSIV-CBC-AES	CO	return value 0
Symmetric encryption/decryption with block stealing.	Non-approved symmetric, block stealing, encryption/decryption algorithm. Encrypt input: plaintext, key. Encrypt output: cyphertext. Decrypt input: cyphertext, key. Decrypt output: plaintext.	AES-XTS, AES-CBC-CTS	CO	return value 0

Table 9. Non-Approved Services.

5. Software/Firmware security

Integrity techniques

The module is provided in the form of binary executable code. To ensure the software security, the module is protected by HMAC-SHA2-256 (HMAC Certs. #A3242) algorithm. The software integrity test key (non-SSP) was preloaded to the module's binary in the factory and used for software integrity test only at the pre-operational self-test. At Module's initialization, the integrity of the runtime executable is verified using a HMAC-SHA2-256 digest which is compared to a value computed at build time. If at load time the MAC does not match the stored, known MAC value, the module would enter to an Error state with all crypto functionality inhibited.

Integrity test on-demand

Integrity test is performed as part of the Pre-Operational Self-Tests. It is automatically executed at power-on. The operator can power-cycle or reboot the tested platform to initiate the software integrity test on-demand.

6. Operational environment

The module operates in a modifiable operational environment per FIPS 140-3 level 1 specifications. The module runs within a commercially available kernel of the general-purpose operating system. The module is executing on the hardware specified in the Table 2.

The operating is restricted to a single operator. Only a single instance of the module is allowed in the Operational environment. The operating environment is non-configurable for the operator. The operational environment provides the capability to separate the module during operation from other functions in the operational environment. Those functions do not obtain information from the module related to the CSPs and do not modify CSPs, PSPs, or the execution flow of the module other than via the interfaces provided by the module itself.

The module does not spawn any processes.

7. Physical security

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

8. Non-invasive security

The module does not implement non-invasive attack mitigation techniques to protect the module's unprotected SSPs from non-invasive attacks referenced in Annex F of FIPS 140-3.

9. Sensitive security parameters management

The following table summarizes the keys and Sensitive Security Parameters (SSPs) that are used by the cryptographic services implemented in the module:

Key/SSP /Name /Type	Strength	Security Function and Cert. Number	Gener-ation	Import /Export	Establish-ment	Storage	Zeroization	Use & related keys
AES Key	128 to 256 bits	AES-CBC; AES-ECB; Cert. #A3242	N/A	Import from caller application within TOEPP; No Export	N/A	N/A: The module does not provide persistent keys/SSPs storage	Automatic zeroization when the crypto structure is deallocated or when the system is powered down	Symmetric Encryption / Decryption
HMAC Key	At least 112 bits	HMAC-SHA-1; HMAC-SHA2-224; HMAC-SHA2-256; HMAC-SHA2-384; HMAC-SHA2-512; Cert. #A3242 (Note: HMAC-SHA2-384 and HMAC-SHA2-512 are not implemented for "with PAA" mode)	N/A	Import From caller application within TOEPP; No Export	N/A	N/A: The module does not provide persistent keys/SSPs storage	Automatic zeroization when the crypto structure is deallocated or when the system is powered down	Keyed Hash

Table 10. SSPs

Notes:

- The module protects all keys/SSPs through the memory separation and protection mechanisms provided by the Kernel.
- The module doesn't operate with any PSP.

SSP entry and output

SSPs enter the module's cryptographic boundary as cryptographic algorithm API parameters in plaintext. They are associated with memory locations and do not persist across power cycles. The Approved and Non-Approved services listed in Table 3. and Table 4. Non-approved Algorithms Not Allowed in Approved Mode of Operation.are working with separate data blocks, so sharing SSP at switching between approved/non-approved is impossible.

SSP storage

The module does not provide persistent storage for keys or SSPs. The module uses pointers to plaintext keys/SSPs that are passed in by the calling application. The module does not store any SSP beyond the lifetime of an API call. Allocated memory in RAM for SSP is managed by the module.

SSP zeroization

The zeroization mechanism for all of the CSPs is to replace 0s in the memory which originally store the CSPs. Zeroization of the sensitive data within the module and is performed automatically in the frame of release of early allocated crypto handler, by calling function crypto_free_<transformation type>() e.g. crypto_free_skcipher() for symmetric cyphers or crypto_free_shash() for hashes. In addition, powering down the tested platform also has all SSPs zeroized.

10. Self-tests

All the pre-operational self-test and conditional self-tests are performed automatically in the frame of module initialization routine at the kernel boot up. The module performs Cryptographic Algorithm Self-Tests (CASTs) first, and then subsequently conducts the software integrity test.

The Self-tests do not require any operator intervention. If any of the Self-tests fail, the module enters the Error state. In the Error state, all output interfaces are inhibited and no cryptographic operations are allowed.

The module can be recovered from the Error state by module reboot only.

Pre-operational self-test

The module performs Pre-operational Self-tests automatically when the module is loaded into memory (i.e. at power on). The Pre-operational Self-tests contain pre-operational software integrity test to ensure that the module is not corrupted. The integrity test is performed on the runtime image of the module using HMAC-SHA2-256. Prior to software integrity test, a CAST for HMAC-SHA2-256 is performed. If the CAST on the HMAC-SHA-256 is successful, the HMAC value of the runtime image is recalculated and compared with the stored HMAC value pre-computed at compilation time (for details, see also Section 5). While the module is performing the Pre-operational Self-tests no other functions are available and all output is inhibited. Once Pre-operational Self-tests are completed successfully, the module enters operational mode and cryptographic services are available.

Conditional self-tests

Cryptographic algorithm self-tests

The module performs Cryptographic Algorithm Self-Tests at module initialization to ensure that the algorithms work as expected, before any security function or process is invoked via module interface, as detailed below.

Algorithm	Test	Condition
AES	AES-ECB 128 and 256 bits encryption KAT (with PAA and without PAA) AES-ECB 128 and 256 bits decryption KAT (with PAA and without PAA) AES-CBC 128 and 256 bits encryption KAT (with PAA and without PAA) AES-CBC 128 and 256 bits decryption KAT (with PAA and without PAA)	Start-up, or on-demand
SHA	SHA-1 KAT (with PAA and without PAA) SHA2-224 KAT (with PAA and without PAA) SHA2-256 KAT (with PAA and without PAA) SHA2-384 KAT (without PAA) SHA2-512 KAT (without PAA)	Start-up, or on-demand
HMAC	HMAC-SHA-1 KAT (with PAA and without PAA) HMAC-SHA2-224 KAT (with PAA and without PAA) HMAC-SHA2-256 KAT (with PAA and without PAA) HMAC-SHA2-384 KAT (without PAA) HMAC-SHA2-512 KAT (without PAA)	Start-up, on-demand

Table 11. Cryptographic Algorithm Self-Tests.

Periodic/Self-tests on-demand

The module performs on-demand self-tests initiated by the operator (via calling ***fips140_post()*** service), by power-cycling, or rebooting the tested platform. The pre-operational software integrity test and the full suite of self-tests listed in Table 11 are executed. The same procedure may be employed by the operator to perform periodic self-tests. If any of the tests fail, the module will enter error state.

Error state and status indicators

The module has a variable (***skc_fips_enabled***) indicating the status of the Self-test. It contains ***0*** if the Self-test was failed and it contains ***1*** if the Self-test was successful.

The kernel logs contains message:

```
FIPS : POST - integrity test failed
```

in case of integrity test is failed,

```
FIPS : <alg name>, test failed, err=<test number>
```

11. Life-cycle assurance

Configuration management

Perforce is used as the repository for both source code and documents. All source code and documents are maintained in an internal server. Release is based on the Changelist number, which is automatically generated. Every check-in process creates a new Changelist number.

Versions of controlled items include information about each version. For documentation, document version number

inside the document provides the current version of the document. Version control maintains all the previous versions 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. The source code of the module available in the Samsung internal Perforce repository, as listed in Functional Design document, is used to build target binary.

Secure initialization and startup

This cryptographic module is built-in along with the Linux Kernel. The module is initialized during the Kernel boot-up before any cryptographic functionality is available. The Kernel is responsible for the initialization and loading processes of the module. The module is designed with module init entry point which ensures that the Pre-operational self-tests and CASTs are initiated automatically when the module is loaded.

Delivery and operation

The module is provided directly to solution developers and is not available for direct download to the general public. 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.

Samsung vets all service providers and establishes secure communication with them for delivery of tools and software updates. If the binary is modified by an 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

The module does not implement security mechanisms to mitigate other attacks.

Glossary and Abbreviations

AES	Advanced Encryption Specification
CAST	Cryptographic Algorithm Self-Test
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DRBG	Deterministic Random Bytes Generator
FIPS	Federal Information Processing Standards Publication
HMAC	Hash Message Authentication Code
KAT	Known-answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
POST	Pre-Operational Self-Test
RNG	Random Number Generator
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard

References

FIPS180-4 **Secure Hash Standard (SHS)**
August 2015
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

FIPS197 **Advanced Encryption Standard**
November 2001
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

FIPS198-1 **The Keyed Hash Message Authentication Code (HMAC)**
July 2008
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf

IG **Implementation Guidance for FIPS 140-3 and the Cryptographic Module Validation Program**
October, 2022
https://csrc.nist.gov/CSRC/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf

SP800-38A **NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques**
December 2001
https://csrc.nist.gov/publications/detail/sp/800-38a/final