Samsung Key Management Module

FIPS 140-2 Security Policy

version 1.8
Last Update: 2013-11-04
1. Introduction ................................................................................................................................. 4
  1.1. Purpose of the Security Policy ............................................................................................. 4
  1.1. Target Audience .................................................................................................................... 4
2. Cryptographic Module Specification .......................................................................................... 5
  2.1. Description of Module .......................................................................................................... 5
  2.2. Description of Approved Mode ............................................................................................ 5
  2.3. Cryptographic Module Boundary .......................................................................................... 6
  2.3.1. Software Block Diagram ................................................................................................. 6
  2.3.2. Hardware Block Diagram ............................................................................................... 7
3. Cryptographic Module Ports and Interfaces ............................................................................ 8
4. Roles, Services and Authentication ........................................................................................... 9
  4.1. Roles ....................................................................................................................................... 9
  4.2. Services .................................................................................................................................. 9
  4.3. Operator Authentication ....................................................................................................... 10
  4.4. Mechanism and Strength of Authentication ........................................................................ 10
5. Finite State Machine ................................................................................................................ 11
6. Physical Security ........................................................................................................................ 12
7. Operational Environment .......................................................................................................... 13
  7.1. Policy ..................................................................................................................................... 13
8. Cryptographic Key Management .............................................................................................. 14
  8.1. Random Number Generation ............................................................................................... 14
  8.2. Key Generation ..................................................................................................................... 14
  8.2.1. Master key ....................................................................................................................... 14
  8.2.2. Device encryption key ....................................................................................................... 14
  8.2.3. Encrypted device encryption key (EDK) and EDK payload .............................................. 14
  8.3. Key Entry and Output ........................................................................................................... 15
  8.4. Key Storage .......................................................................................................................... 15
  8.5. Zeroization Procedure .......................................................................................................... 15
9. Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC) ......................... 16
10. Self Tests .................................................................................................................................. 17
  10.1. Integrity Check ................................................................................................................... 17
  10.2. Conditional Tests .............................................................................................................. 17
11. Design Assurance ..................................................................................................................... 19
11.1. Configuration Management ................................................................. 19
11.2. Delivery and Operation................................................................. 19
12. Mitigation of Other Attacks .............................................................. 20
13. Glossary and Abbreviations .............................................................. 21
14. References.......................................................................................... 23
1. Introduction
This document is a non-proprietary FIPS 140-2 Security Policy for the Samsung Key Management 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 two major reasons that a security policy is required:

- To provide a specification of the cryptographic security that will allow individuals and organizations to determine whether a cryptographic module, as implemented, satisfies a 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.

1.1. Target Audience
This document has the following audience:

- Those specifying cryptographic modules
- Administrators of the cryptographic module(s)
- Users of the cryptographic module(s)
2. Cryptographic Module Specification

This document is the non-proprietary security policy for the Samsung Key Management 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 Key Management Module is a software-only Security Level 1 cryptographic module that provides key management services for user space applications. The Key Management 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.

<table>
<thead>
<tr>
<th>Security Component</th>
<th>Security Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cryptographic Module Specification</td>
<td>1</td>
</tr>
<tr>
<td>Cryptographic Module Ports and Interfaces</td>
<td>1</td>
</tr>
<tr>
<td>Roles, Services, and Authentication</td>
<td>1</td>
</tr>
<tr>
<td>Finite State Model</td>
<td>1</td>
</tr>
<tr>
<td>Physical Security</td>
<td>N/A</td>
</tr>
<tr>
<td>Operational Environment</td>
<td>1</td>
</tr>
<tr>
<td>Cryptographic Key Management</td>
<td>1</td>
</tr>
<tr>
<td>EMI/EMC</td>
<td>1</td>
</tr>
<tr>
<td>Self Tests</td>
<td>1</td>
</tr>
<tr>
<td>Design Assurance</td>
<td>3</td>
</tr>
<tr>
<td>Mitigation of Other Attacks</td>
<td>N/A</td>
</tr>
</tbody>
</table>

*Table 1: Security Levels*

The module has been tested on the following platforms:

<table>
<thead>
<tr>
<th>Module/Implementation</th>
<th>Device</th>
<th>O/S &amp; Ver.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samsung Key Management Module (KM1.1)</td>
<td>Galaxy S2</td>
<td>Android Ice-cream Sandwich 4.0</td>
</tr>
<tr>
<td>Samsung Key Management Module (KM1.1)</td>
<td>Galaxy S3</td>
<td>Android Ice-cream Sandwich 4.0</td>
</tr>
<tr>
<td>Samsung Key Management Module (KM1.1)</td>
<td>Galaxy Note II</td>
<td>Android Jelly Bean 4.1</td>
</tr>
<tr>
<td>Samsung Key Management Module (KM1.3)</td>
<td>Galaxy S4</td>
<td>Android Jelly Bean 4.2</td>
</tr>
</tbody>
</table>

*Table 2: Tested Platforms*

2.2. Description of Approved Mode

The crypto module has only FIPS 140-2 approved mode.

In Approved mode the module will support the following approved functions:
The CAVP certificate numbers for each approved algorithm is provided in Table 5 in section 4.2. The Key Management module has only FIPS approved mode.

2.3. Cryptographic Module Boundary

2.3.1. Software Block Diagram

![Software Block Diagram]

*Figure 1: Software Block Diagram*

The binary image that contains the Key Management module is as follows:

- `libsec_km.so (version KM1.1)` - Galaxy S2
- `libsec_km.so (version KM1.1)` - Galaxy S3
- `libsec_km.so (version KM1.1)` - Galaxy Note II
- `libsec_km.so (version KM1.3)` - Galaxy S4

Related documentation:
S/W Detailed Level Design (FIPS_KM Func Design.docx) v2.2

- Samsung Key Management Cryptographic Module
  (SamsungKeyManagement_SPv1.8.docx)

Note: The master component list is provided in Section 2.14 of S/W Detailed Level Design document. Please contact the vendor (as specified on the FIPS 140-2 Validation list) to request access to the documentation related to the module.

2.3.2. Hardware Block Diagram

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

Figure 2: Hardware Block Diagram
3. Cryptographic Module Ports and Interfaces

<table>
<thead>
<tr>
<th>FIPS Interface</th>
<th>Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Input</td>
<td>API input parameters</td>
</tr>
<tr>
<td>Data Output</td>
<td>API output parameters</td>
</tr>
<tr>
<td>Control Input</td>
<td>API function calls</td>
</tr>
<tr>
<td>Status Output</td>
<td>API return codes; device log file; the status of the module is provided at UI via the status API</td>
</tr>
<tr>
<td>Power Input</td>
<td>Physical power connector</td>
</tr>
</tbody>
</table>

*Table 3: Ports and Interfaces*
4. Roles, Services and Authentication

4.1. Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Services (see list below)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Encryption, Decryption, Random Numbers, Digest Creation, Key Generation, Change Password, Verify Password, Check Status</td>
</tr>
<tr>
<td>Crypto Officer</td>
<td>Configuration, Encryption, Decryption, Random Numbers, Initialization of Module, Digest Creation, Key Generation, Change Password, Verify Password, Check Status</td>
</tr>
</tbody>
</table>

4.2. Services

<table>
<thead>
<tr>
<th>Role</th>
<th>Service</th>
<th>CSP</th>
<th>Modes</th>
<th>FIPS Approved (Cert #)</th>
<th>Access (Read, Write, Execute)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User, Crypto Officer</td>
<td>AES encryption and decryption</td>
<td>256 bit keys</td>
<td>ECB</td>
<td>(Cert #2048, 2098, 2142, 2143, 2257, 2393)</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>HMAC- SHA-256</td>
<td>HMAC Key</td>
<td>N/A</td>
<td>(Cert #1245, 1273, 1309, 1310, 1384, 1484)</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>SHA-256</td>
<td>N/A</td>
<td>N/A</td>
<td>(Cert #1792, 1822, 1864, 1865, 1944, 2055)</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>RNG ANSI X9.31</td>
<td>Seed Key</td>
<td>AES-128</td>
<td>(Cert #1069, 1080, 1097, 1098, 1127, 1185)</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>PBKDF SP 800-132 (Generate Key)</td>
<td>Password, DEK, EDK</td>
<td>N/A</td>
<td>Vendor affirmed</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>PBKDF SP 800-132 (Verify Password)</td>
<td>Password, EDK</td>
<td>N/A</td>
<td>Vendor affirmed</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>PBKDF SP 800-132 (Password Change)</td>
<td>Password, EDK</td>
<td>N/A</td>
<td>Vendor affirmed</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>PBKDF SP 800-132 (Get Key)</td>
<td>Password, EDK</td>
<td>N/A</td>
<td>Vendor affirmed</td>
<td>R, W, EX</td>
</tr>
<tr>
<td>Crypto Officer</td>
<td>Initialization</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>Self Test</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>User, Crypto Officer</td>
<td>Check Status/Get State</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>R</td>
</tr>
</tbody>
</table>

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.
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 S/W Detailed Level 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
Both the phone and the tablet are single user devices. The operating system shall be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded).
The external applications that make calls to the cryptographic module should belong to 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 which is externally seeded by the /dev/random utility which provides entropy of 128 bits.

The /dev/random utility is outside of the cryptographic boundary of this module.

Caveat: The module generates cryptographic keys whose strengths are modified by available entropy

8.2. Key Generation
The following keys/CSPs are used in the key management module:

- Seed Key
- Password: User entered 4 to 32 characters long
- Master key: Generated using PBKDF
- Device encryption key: 256 bits long, generated from RNG
- Encrypted device encryption key: Encrypted DEK with master key using AES
- EDK Payload or Payload: consists of EDK, salt, HMAC of EDK

8.2.1. Master key
Master Key (MK) is generated using Password-based key derivation function (NIST 800-132).
Password length must be greater than 4, but cannot be greater than 32 characters.
Note: As per NIST 800-132, passwords shorter than 10 characters are usually considered to be weak.
This master key is used as an AES encryption key (256 bits) to encrypt the device encryption key (DEK). It is also used to compute HMAC with SHA-256.

8.2.2. Device encryption key
Device encryption key (DEK) is derived as 32 bytes read from RNG.

8.2.3. Encrypted device encryption key (EDK) and EDK payload
DEK is protected by encrypting it with the master key. The resulting encrypted key is called EDK.
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. The output of PBKDF is DEK, which is supplied to the calling application via API call.

8.4. Key Storage

The encryption key (DEK) for the purposes of storage, is randomly generated which is protected using master key.

8.5. Zeroization Procedure

This is a dynamic library module which provides API based services. All the temporary variables for CSP are zeroized using memset() function, which is a Linux kernel function which fills memory with a constant byte. In this case, the module performs zeroization by filling the memory with zeroes. DEK is zeroized using the zeroization API. It is the responsibility of the calling application to zeroize DEK by invoking this function.

For more information on APIs, please refer to the S/W Detailed Level 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 S/W Detailed Level Design document.
10. Self Tests
The power-on-self-test is executed automatically during the loading of the module without any operator intervention. If the self test is successful, the module enters the FIPS mode. If the self test fails, the module will be put in error state and thus disabled. There will be an indicator in User Interface via the status API. For details on status specific API, please refer to the Functional Design document.

During self test, the value calculated is verified against the known answer. If there is no match, then immediately the module enters into FIPS_ERR state. Once the module is in FIPS_ERR state, the module becomes unusable via any interface.

The module implements each of the following Known Answer Tests separately:

- AES encryption/decryption
- HMAC-SHA-256
- SHA-256
- Random Number Generator

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. Integrity Check

- Build Time
  - HMAC-SHA-256 calculated on libsec_km.so (dynamic library) file
    - HMAC appended to libsec_km.so file
- Run Time
  - libsec_km.so is read as a file
  - When Integrity test routine is called
    - Perform HMAC-SHA-256 on the read libsec_km.so value in ram
    - Read stored HMAC value located after libsec_km.so (last 32 bytes)
    - If calculated and stored values do not match, set error state, STATUS_ERROR_INTEGRITY and the system property as “error_integrity”

10.2. 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. This
RNG is externally seeded by /dev/random, which is also subjected to a CRNG test. Note that /dev/random is outside the module boundary.
11. Design Assurance

11.1. Configuration Management

All source code is maintained in internal source code servers and the tool, Perforce, is used as code control. Release is based on the Change List number maintained by Perforce, 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.

All documents are maintained in an internal document server per project. The versioning tool used is Sub version (svn). The version number is auto generated by the tool and version is controlled by a check-in and check-out mechanism.

In the development team, only authorized developers verified by login/password is allowed to access permitted documents in version control system.

11.2. Delivery and Operation

The key management module is never released as Source code. It may be released as Source for internal purposes based on Change List number generated by svn.

Official release binaries can only be made through build system, run by the Software Project Lead.

Once the device enters manufacturing phase, source code branch is locked. Once it is locked, source control system provides only read access. Thus, no one can then modify the source code in the Perforce depot.

Final binary thus built is registered with its hash value to internal system which is not connected to any other network.

Only authorized personnel through VPN can register the binary to automated manufacturing system, so that it can be downloaded to hardware without any manual intervention. Employees are not allowed to bring in any personal belongings to the manufacturing facility and entrance is controlled with employee ID based badge access and monitored using CCTV.

The binary is released only by Samsung released tool and OTA (Over the Air). Over the air mechanism is controlled by service providers. 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  Cipher Block Chaining
CCM  Counter with Cipher Block Chaining-Message Authentication Code
CFB  Cipher Feedback
CMT  Cryptographic Module Testing
CMVP  Cryptographic Module Validation Program
CSP  Critical Security Parameter
CVT  Component Verification Testing
DEK  Device Encryption Key
DES  Data Encryption Standard
DSA  Digital Signature Algorithm
EAL  Evaluation Assurance Level
EDK  Encrypted Device Encryption Key
FSM  Finite State Model
HMAC  Hash Message Authentication Code
MAC  Message Authentication Code
MK  Master Key
NIST  National Institute of Science and Technology
NVLAP  National Voluntary Laboratory Accreditation Program
OFB  Output Feedback
O/S  Operating System
PBKDF  Password Based Key Derivation Function
RNG  Random Number Generator
RNGVS  Random Number Generator Validation System
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