FIPS 140-2 Security Policy for:

Huawei Device (Dongguan) Co. Ltd. EDK Management Module

Version 1.6

Table of Contents

1 Overview

This document is a non-proprietary FIPS 140-2 Security Policy for the Huawei EDK Management Module v1.0 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

- 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
- 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 140-2 validation. It is intended for the following people:

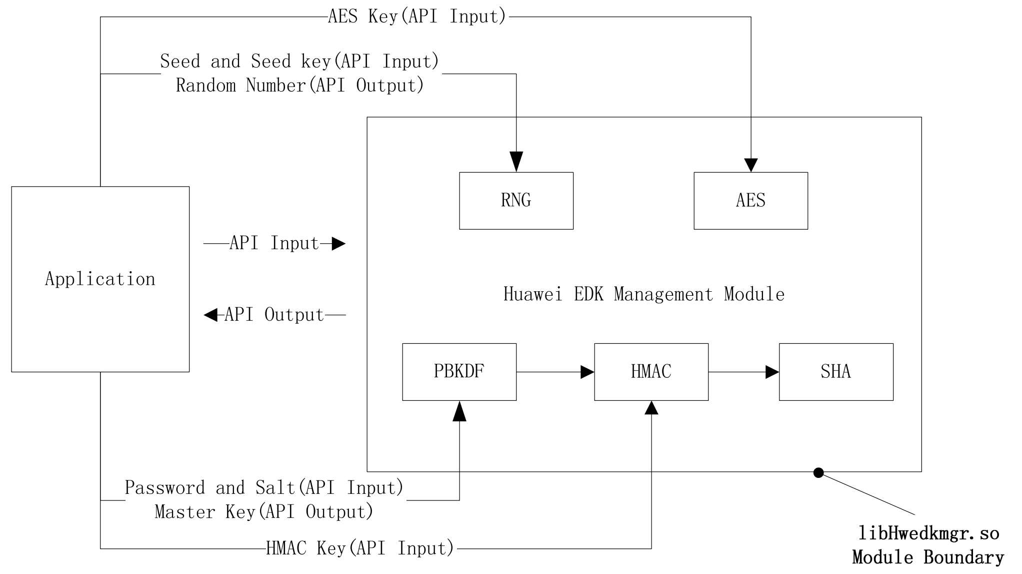
- Developers working on the release
- FIPS 140-2 testing lab
- Crypto Module Validation Program (CMVP)
- Consumers

The following table shows the overview of the security level for each section.

Section	Level
1. Cryptographic Module Specification	1
2. Cryptographic Module Ports and Interfaces	1
3. Roles, Services, and Authentication	1
4. Finite State Model	1
5. Physical Security	N/A
6. Operational Environment	1
7. Cryptographic Key Management	1
8. EMI/EMC	1
9. Self-Tests	1
10. Design Assurance	1
11. Mitigation of Other Attacks	N/A
Overall Level	1

Table 1 - Security Level Detail

Software

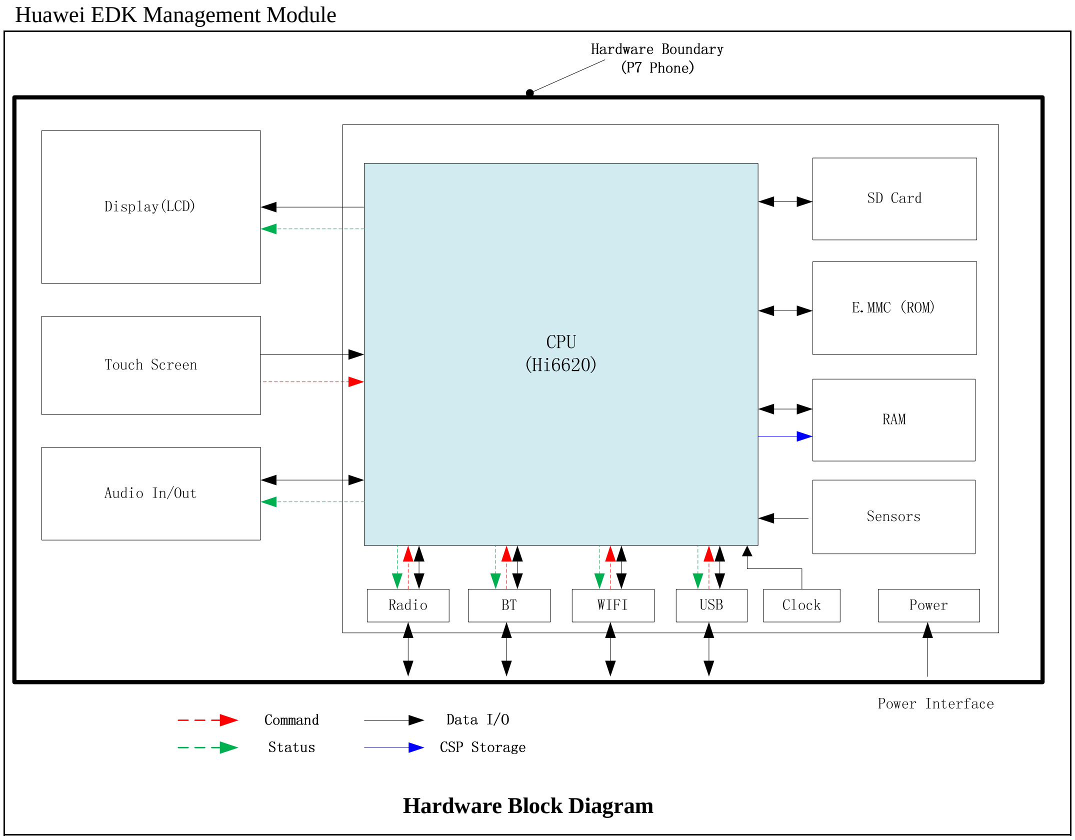


Software Block Diagram

Hardware

Front view of P7

Back view of P7



Hardware Block Diagram

The module has been tested on the following platforms.

	Module Name	Hardware Version	Firmware Version	Software Version	OE	
1	Huawei EDK Management Module	SOPHIA_ULG_VD		P7-L00V100R001C17B210	Emotion UI 2.3	Android 4.4.2

Table 2 - Module Information

2 Module Specification

The EDK management module has only FIPS 140-2 approved mode.

In approved mode the EDK management module will support the following approved functions:

- ✓ AES Certificate #2967 and #3178 (128/192/256 ECB, CBC, OFB, CFB 1, CFB 8, CFB 128, CTR, XTS, CCM, GCM)
- ✓ SHS Certificate #2495 (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)
- ✓ HMAC Certificate #1881 (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)

- ✓ RNG Certificate #1299 (ANSI X9.31)
- ✓ Password-Based Key Derivation Function (PBKDF) Vendor Affirmed (NIST 800-132)

User Guide

The FIPS mode initialization is performed when the application invokes the FIPS_module_mode_set() call. Prior to this invocation the Module is uninitialized with the internal global flag ‘fips_mode’ set to FALSE indicating non-FIPS mode by default.

The FIPS_module_mode_set () function verifies the integrity of the runtime executable using a HMAC-SHA-1 digest computed at build time. If this computed HMAC-SHA-1 digest matches the stored known digest then the power-up self-test, consisting of the algorithm specific Known Answer tests, is performed. If any component of the power-up self-test fails the internal global error flag ‘fips_selftest_fail’ is set to prevent subsequent invocation of any cryptographic function calls. If all components of the power-up self-test are successful then FIPS_module_mode_set () sets the ‘fips_mode’ flag to TRUE and the module is in FIPS mode. And in the FIPS operational mode, if the conditional test of RNG failed, it will go to Error state, and in all other cases, it will go to power off state directly.

3 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;
Power Input	Physical power connector

Table 3 - Ports and Interfaces

When the Module is performing self-tests or is in an error state, all output on the logical data output interface is inhibited. As a software module, it cannot control the physical ports.

4 Roles Services and Authentication

The module does not provide identification or authentication mechanisms that would distinguish between the two supported roles. These roles are implicitly assumed by the services that are accessed, and can be differentiated by assigning module installation and configuration services to the Crypto Officer.

Role		Type of Auth	Authentication	Auth Strength	Mult Attempt Str
Crypto Officer		N/A	N/A	N/A	N/A
User		N/A	N/A	N/A	N/A

Table 4 - Identification and Authentication Policy

The services provided by the Module are listed in the following table.

	Service	Modes	Role	Keys & CSPs	Alg Cert	RWE	API Functions
1	AES encryption/ decryption	ECB, CBC, CFB1, CFB8, CFB128, OFB, CTR, XTS, GCM, CCM	User	128-bits key 192-bits key 256-bits key	#2967 #3178	R,W,E	FIPS_cipherinit FIPS_cipher FIPS_cipher_ctx_new FIPS_cipher_ctx_init FIPS_cipher_ctx_ctrl FIPS_cipher_ctx_copy FIPS_cipher_ctx_set_key_length FIPS_cipher_ctx_free FIPS_cipher_ctx_cleanupCRYPTO_ccm128_tag CRYPTO_ccm128_setiv CRYPTO_ccm128_aad CRYPTO_ccm128_encrypt_ccm64 CRYPTO_ccm128_encrypt CRYPTO_ccm128_decrypt_ccm64 CRYPTO_ccm128_decrypt AES_set_encrypt_key CRYPTO_ccm128_init AES_encrypt CRYPTO_xts128_encrypt AES_set_decrypt_key AES_decrypt

							CRYPTO_gcm128_setiv CRYPTO_gcm128_aad CRYPTO_gcm128_encrypt_ctr32 CRYPTO_gcm128_tag CRYPTO_gcm128_encrypt CRYPTO_gcm128_decrypt_ctr32 CRYPTO_gcm128_decrypt CRYPTO_gcm128_finish CRYPTO_gcm128_init CRYPTO_ctr128_encrypt_ctr32 CRYPTO_ctr128_encrypt CRYPTO_cfb128_8_encrypt CRYPTO_cfb128_1_encrypt CRYPTO_cfb128_encrypt CRYPTO_ofb128_encrypt CRYPTO_cbc128_encrypt AES_cbc_encrypt EVP_aes_128_cbc EVP_aes_128_ecb EVP_aes_128_ofb EVP_aes_128_cfb128 EVP_aes_128_cfb1 EVP_aes_128_cfb8 EVP_aes_128_ctr EVP_aes_192_cbc EVP_aes_192_ecb EVP_aes_192_ofb EVP_aes_192_cfb128 EVP_aes_192_cfb1 EVP_aes_192_cfb8 EVP_aes_192_ctr EVP_aes_256_cbc EVP_aes_256_ecb EVP_aes_256_ofb EVP_aes_256_cfb128 EVP_aes_256_cfb1 EVP_aes_256_cfb8 EVP_aes_256_ctr EVP_aes_128_gcm EVP_aes_192_gcm EVP_aes_256_gcm

							EVP_aes_128_xts EVP_aes_256_xts EVP_aes_128_ccm EVP_aes_192_ccm EVP_aes_256_ccm CRYPTO_cbc128_decrypt CRYPTO_gcm128_new CRYPTO_gcm128_release FIPS_get_cipherbynid
2	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512	N/A	User	N/A	#2495	R,W,E	FIPS_digest FIPS_digestinit FIPS_digestupdate FIPS_digestfinal FIPS_md_ctx_init FIPS_md_ctx_create FIPS_md_ctx_copy FIPS_md_ctx_destroy FIPS_md_ctx_cleanup SHA512_Final SHA512_Update SHA512_Init SHA384_Init SHA256_Final SHA256_Update SHA256_Init SHA224_Init SHA1_Final SHA1_Update SHA1_Init EVP_sha1 EVP_sha224 EVP_sha256 EVP_sha384 EVP_sha512 SHA224_Update SHA224_Final SHA256 SHA224 SHA384_Final SHA384_Update SHA384 SHA512 FIPS_md_ctx_destroy
3	HMAC-SHA-1	N/A	User	HMAC Key	#1881	R,W,E	FIPS_hmac FIPS_hmac_init

	HMAC-SHA-224 HMAC-SHA-256 HMAC-SHA-384 HMAC-SHA-512						FIPS_hmac_init_ex FIPS_hmac_update FIPS_hmac_ctx_copy FIPS_hmac_ctx_init FIPS_hmac_ctx_set_flags FIPS_hmac_final FIPS_hmac_ctx_cleanup FIPS_get_digestbynid
4	RNG	AES-128 AES-192 AES-256	User	Seed and Seed Key	#1299	R,W,E	FIPS_x931_bytes FIPS_x931_reset FIPS_x931_seed FIPS_x931_set_dt FIPS_x931_set_key FIPS_x931_status FIPS_x931_test_mode FIPS_x931_stick FIPS_get_timevec FIPS_x931_method
5	PBKDF	HMAC-SHA-1 HMAC-SHA-224 HMAC-SHA-256 HMAC-SHA-384 HMAC-SHA-512	User	Password and Salt	PKCS#5	R,W,E	PKCS5_PBKDF2_HMAC
6	Initialization	N/A	Crypto Officer	N/A	N/A	E	FIPS_module_mode_set
7	Get Status	N/A	Crypto Officer	N/A	N/A	R,E	FIPS_module_mode FIPS_selftest_failed FIPS_module_version FIPS_module_version_text
8	Self Test	N/A	Crypto Officer	N/A	N/A	E	FIPS_selftest fips_set_selftest_fail FIPS_selftest_sha1 FIPS_selftest_hmac FIPS_selftest_aes FIPS_selftest_aes_ccm FIPS_selftest_aes_gcm FIPS_selftest_aes_xts FIPS_selftest_x931 fips_pkey_signature_test fips_cipher_test
9	Integrity Validation	N/A	Crypto Officer	N/A	N/A	E	FIPS_text_start FIPS_rodata_start FIPS_incore_fingerprint FIPS_text_end

							FIPS_rodata_end FIPS_check_incore_fingerprint
10	Post	N/A	User	N/A	N/A	E	fips_post_started fips_post_success fips_post_failed FIPS_post_set_callback fips_post_cb fips_post_begin fips_post_end fips_post_status
11	Exception	N/A	User	N/A	N/A	E	FIPS_die

Table 5 – FIPS Approved services

1. The AES algorithm provides encryption and decryption services with key size of 128,192 and 256 bits, and modes of ECB,CBC,CFB1,CFB8, CFB128, OFB,CTR, XTS,GCM,CCM;

2. The SHA algorithm provides the cryptographic hash functions to produce a message digest. The module provides SHA 1, SHA 224, SHA 256, SHA 384 and SHA 512.

3. The HMAC algorithm provides the functions to calculate a message authentication code involving a cryptographic hash function, which can be SHA 1, SHA 224, SHA 256, SHA 384 and SHA 512.

4. The RNG algorithm validated for use with the module allows for the generation of AES128, 192, and 256 bit keys.

5. The PBKDF2 algorithm provides password-based encryption functionality based on PKCS#5, with a SHA-based HMAC.

5 Physical Security

Huawei EDK Management Module is comprised of software only. Physical security is not applicable.

6 Operational Environment

This module will operate in a modifiable operational environment per the FIPS 140-2 definition. The phone is a single user device. 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.

7 Key Management

Key/CSP	Service	Length	Strength	Type	Zeroize Method	Establishment	Output	Persistence/ Storage
128-bits key	1.AES	128	128	Symmetric	FIPS_cipher_ctx_cleanup	PBKDF	EDK, AES to encrypt DEK	Plain Text, Store in RAM for the lifetime of API call
192-bits key		192	192					
256-bits key		256	256					
HMAC key	3.HMAC	160	160	HMAC key	FIPS_hmac_ctx_cleanup	Password	No	Plain Text, Store in RAM for the lifetime of API call
		224	224					
		256	256					
		384	384					
		512	512					
RNG CSPS	4. RNG	128	128	Seed and Seed key	FIPS_x931_reset	dev/random	DEK	Plain Text, Store in RAM for the lifetime of API call
		192	192					
		256	256					
Password	5. PBKDF	N/A	N/A	Password	FIPS_hmac_ctx_cleanup	User Input	No	Plain Text, Store in RAM for the lifetime of API call
Salt		N/A	N/A	Salt		dev/random		

Table 6 - Key Management

- Random Number Generation

The module employs an ANSI X9.31 compliant random number generator for creation of keys which is externally seeded by the application which is using the module. The application may get the seed key from /dev/random utility, and pass the pointer of the seed key to the module by calling FIPS_x931_seed(const void *buf, int num) .

Caveat: The encryption strength of AES keys are modified by available entropy of seeds that are provided to the RNG; there is no assurance of the minimum strength of the generated keys.

- 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.

- Key generation

- DEK is generated using approved RNG (Certificate #1299), and output to the application.
- AES symmetric key is input by application, and is generated using approved PBKDF2 (PKCS#5).
- EDK is generated using approved AES (Certificate #2967), using the AES symmetric key to encrypt DEK.

- Key storage

The module does not provide persistent key storage for keys or CSPs. The module uses pointers to plaintext keys/CSPs that are passed in by the calling application. The module does not store any CSP beyond the lifetime of an API call. And all keys and CSPs are ephemeral and are destroyed when released by the appropriate API function calls. Keys and CSPs residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the Module defined API.

8 EMI/EMC

Lab: Reliability Laboratory of Huawei Technologies Co., Ltd

Report No: SYBH(Z-EMC)007032014-1

9 Self Tests

The module performs a number of power-up and conditional self-tests to ensure proper operation of the module. Power-up tests include cryptographic algorithm known answer tests and integrity tests. The integrity tests are performed using a HMAC-SHA-1 digest calculated over the object code in the FIPS Object Module. Power-up tests are run automatically when the module is initialized. Additionally, powerup tests may be executed at any time by calling the FIPS_selftest() function and verifying it returns true. No FIPS mode cryptographic functionality will be available until after successful execution of all power-up tests. No authentication is required to perform self-tests either automatically or upon demand. The failure of any power-up self-test or continuous test causes the module to enter the Self-Test Failure state, and all cryptographic operations are disabled until the module is reinitialized with a successful FIPS_module_mode_set () call.

Power-up Tests - (Known Answer Tests):

- AES encryption/decryption 128, 192, and 256 bit keys (ECB, CBC, CFB1, CFB8, CFB128, OFB, CTR, XTS, GCM, CCM)
- 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 (from known IV)

Conditional Test

Conditional tests are performed automatically as necessary and cannot be turned off. Currently, all conditional tests relate to services available only to users. Thus, conditional and critical function tests are not performed at any time in response to Crypto Officer actions.

A continuous random number generator test is performed. If values of two consecutive random numbers match, then crypto module goes into error state. In order to recover from the error state, the module must be powered off and then powered-on and re-initialized. This RNG is externally seeded by /dev/random, which is outside the module boundary.

10 Design Assurance

- Configuration Management

All source code is maintained in internal source code server, and GIT is used as code control. Release is based on the submit id which is auto-generated. Every check-in process creates a new submit id.

Revision history inside the document provides the current version of the document. Version control maintains all the previous version, and the version number of the module is defined as "FIPS 1.0.1 validated module DD MM YYYY", if there is more than one version in the same date, version number will add one more character like "FIPS 1.0.1a validate module DD MM YYYY".

- Delivery

The module is never released as source code. The module is compiled to a binary and packaged in the UPDATE.APP, which is used to download to devices in manufacturing factory. And the development team and the manufacturing factory share a secured internal server for exchanging the UPDATE.APP. The factory is also a secure site with strict access control to the manufacturing facilities. The module binary is downloaded to the devices using direct binary image installation at the factory. The devices are then delivered to mobile service operators. Users can not install or modify the module. The developer has the capability to deliver software update using OTA(Over The Air) update or using SD card update. Only Huawei can deliver the update package as the device will verify the signature of the update package. Once the module is installed on the device, the Android loader will call FIPS_module_mode_set() upon startup of the device and this will power-up the module and execute the self-test procedure. Once the self-tests have completed successfully, the module is operational.

11 Mitigation of Other Attacks

No other attacks are mitigated.