ICU Medical, Inc.

ICU Medical CE3.0 OpenSSL Cryptographic Module

FIPS 140-2 Non-Proprietary Security Policy

Document Version 6.2

Last Update: 2022-05-05

Table of Contents

1 INTRODUCTION

This document is the non-proprietary Security Policy for the ICU Medical CE3.0 OpenSSL Cryptographic Module SW version: 2.0.9.1. It describes how the requirements, specified in Federal Information Processing Standards Publication 140-2 (FIPS PUB 140-2) for a Security Level 1, are being met.

1.1 PURPOSE OF THE SECURITY POLICY

There are three major reasons why a security policy is requested:

- 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, protections, and access rights provided by the cryptographic module that will allow individuals and organizations to determine whether it meets their security requirements.

1.2 TARGET AUDIENCE

This document will be one of many that are submitted as a package for FIPS validation; it is intended for the following people:

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

2 CRYPTOGRAPHIC MODULE SPECIFICATION

This document is the non-proprietary security policy for the ICU Medical CE3.0 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 FIPS140-2 standard in each of the required areas.

2.1 DESCRIPTION OF MODULE

The ICU Medical CE3.0 OpenSSL Cryptographic Module is a software only (multi-chip standalone) Security Level 1 cryptographic module that provides general purpose cryptographic services to the applications running in the user space of the underlying operating system through an application program interface. The logical cryptographic boundary of the Module is the fipscanister object module, a single object module file named fipscanister.o.

The following table shows the Security Level for each of the eleven sections of the validation:

Security Component	FIPS 140-2 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	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A
Overall Security Level	**1**

Table 1 – Security Level of the Module

The module has been tested using the following ICU Medical Communication Engine (CE3.0) platforms:

Platform	Processor	OS and Version
ICU Medical Plum 360 Infusion System	i.MX53 Arm Cortex-A8	Android 2.3.7

Table 2 – Tested Platforms

Android 2.3.7, which is the base for the CE3.x OS, was heavily customized and optimized to meet the requirements of the ICU Medical Communication Engine HW and SW specifications.

2.2 DESCRIPTION OF FIPS APPROVED AND NON-APPROVED MODE

By default, the initialization is executed via DEP (Default Entry Point) as specified in FIPS 140-2 Implementation Guidance 9.10. As part of initialization, the module performs self-tests and enters the FIPS Mode. Whenever, the external application requires “Non-FIPS Approved” mode, it does it by calling FIPS_mode_set(FIPS_MODE_OFF).

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

Function	Algorithm	Options	Cert #
Encryption, Decryption and Message Authentication	[FIPS 197] AES [SP 800-38A] [SP 800-38C] CCM [SP 800-38B] CMAC [SP 800-38D] GCM[1]	128/192/256 ECB, CBC, OFB, CFB 1, CFB 8, CFB 128, CTR, CCM, GCM, CMAC generate and verify	A1878
Symmetric key generation[2]	CKG (vendor affirmed) Direct symmetric key generation using unmodified DRBG output (vendor affirmed)		

[1] The IV is randomly generated internally using the Approved DRBG (Cert. #A1878) using scenario 2.

[2] Keys/CSPs generated in FIPS mode cannot be used in non-FIPS mode, and vice versa. When switching between modes of operation, it is the responsibility of the calling application to reinstantiate the module into memory in order to ensure CSPs are not shared between modes.

TLS KDF	CVL	TLSv1.2, SHA2-256, SHA2-384	A1878
Random Number Generation	[SP 800-90A] DRBG[3] Prediction resistance supported for all variations	Hash DRBG, HMAC DRBG, CTR DRBG (AES)	A1878
Digital Signature and Asymmetric Key Generation	[FIPS 186-4] ECDSA	PKG: CURVES (P-224 P-256 P-384 P-521 K-233 K-283 K-409 K-572 B-233 B-283 B-409 B-571 ExtraRandomBits TestingCandidates) PKV: CURVES (ALL-P ALL-K ALL-B) SigGen: CURVES P-224: (SHA-224, 256, 384, 512) P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) K-233: (SHA-224, 256, 384, 512) K-283: (SHA-224, 256, 384, 512) K-409: (SHA- 224, 256, 384, 512) K-571: (SHA-224, 256, 384, 512) B-233: (SHA-224, 256, 384, 512) B-283: (SHA-224, 256, 384, 512) B-409: (SHA-224, 256, 384, 512) B-571: (SHA-224, 256, 384, 512)) SigVer: CURVES P-192: (SHA-1, 224, 256, 384, 512) P-224: (SHA-1, 224, 256, 384, 512) P-256: (SHA-1, 224, 256, 384, 512) P-384: (SHA-1, 224, 256, 384, 512) P-521: (SHA-1, 224, 256, 384, 512) K-163: (SHA-1, 224, 256, 384, 512) K-233: (SHA-1, 224, 256, 384, 512) K-283: (SHA-1, 224, 256, 384, 512) K-409: (SHA-1, 224, 256, 384, 512) K-571: (SHA-1, 224, 256, 384, 512) B-163: (SHA-1, 224, 256, 384, 512) B-233: (SHA-1, 224, 256, 384, 512) B-283: (SHA-1, 224, 256, 384, 512) B-409: (SHA-1, 224, 256, 384, 512) B-571: (SHA-1, 224, 256, 384, 512))	A1878
Keyed Hash	[FIPS 198-1] HMAC[4]	SHA1, SHA2 (224, 256, 384, 512)	A1878
Digital Signature	[FIPS 186-4] RSA	SigGen 9.31 (2048/3072 with SHA-256/SHA-384/SHA-512), SigGen PKCS1.5 (2048/3072 with all SHA-2) SigGen PSS (2048/3072 with all SHA-2) SigVer 9.31 (1024/2048/3072/4096 with SHA-1/ SHA-256/SHA-384/SHA-512), SigVer PKCS1.5 (1024/2048/3072/4096 with SHA-1/SHA-224/SHA-256/SHA-384/SHA-512),	A1878

[3] For all DRBGs the "supported security strengths" is just the highest supported security strength per [SP800-90A] and [SP800-57], which is between 128 and 256 bits

[4] HMAC Key sizes must be >= 112 bits in FIPS mode

		SigVer PSS (1024/2048/3072/4096 with SHA-1/SHA-224/SHA-256/SHA-384/SHA-512)	
Message Digests	[FIPS 180-4] SHS	SHA1, SHA2 (224, 256, 384, 512)	A1878
KAS-SSC	SP 800-56A Rev3	Scheme: Ephemeral unified P-224: (SHA-224, 256, 384, 512) P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) K-233: (SHA-224, 256, 384, 512) K-283: (SHA-224, 256, 384, 512) K-409: (SHA-224, 256, 384, 512) K-571: (SHA-224, 256, 384, 512) B-233: (SHA-224, 256, 384, 512) B-283: (SHA-224, 256, 384, 512) B-409: (SHA-224, 256, 384, 512) B-571: (SHA-224, 256, 384, 512)	A1878
KAS	SP 800-56A Rev3	PKG: CURVES (P-224 P-256 P-384 P-521 K-233 K-283 K-409 K-572 B-233 B-283 B-409 B-571 ExtraRandomBits TestingCandidates) Scheme: Ephemeral unified P-224: (SHA-224, 256, 384, 512) P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) K-233: (SHA-224, 256, 384, 512) K-283: (SHA-224, 256, 384, 512) K-409: (SHA-224, 256, 384, 512) K-571: (SHA-224, 256, 384, 512) B-233: (SHA-224, 256, 384, 512) B-283: (SHA-224, 256, 384, 512) B-409: (SHA-224, 256, 384, 512) B-571: (SHA-224, 256, 384, 512) KDF: TLSv1.2, SHA2-256, SHA2-384	A1878

Table 3a – FIPS Approved Cryptographic Functions

The Module implements the following services which are Non-Approved per the SP 800-131A transition:

Function	Algorithm	Options
Encryption, Decryption	[SP 800-38E] AES XTS	
Random Number Generation; Symmetric key Generation	[SP 800-90A] DRBG	Dual EC DRBG
Digital Signature and Asymmetric Key Generation	[FIPS 186-2] DSA	PQG Gen, Key Pair Gen, Sig Gen (1024 with all SHA sizes, 2048/3072 with SHA1)
	[FIPS 186-4] DSA	PQG Gen (2048/3072 with all SHA-2 sizes) PQG Ver (1024/2048/3072 with SHA-1 and all SHA-2 sizes) Key Pair Gen (2048/3072) Sig Gen (2048/3072 with all SHA-2 sizes) Sig Ver (1024/2048/3072 with SHA-1 and all SHA-2 sizes)

	[FIPS 186-2] ECDSA	PKG: CURVES (P192 K163 B163) SIG(gen): CURVES (All curves)
	[FIPS 186-4] ECDSA	PKG: CURVES (P192 K163 B163) SigGen: CURVES (P192: (SHA1, 224, 256, 384, 512) P224:(SHA1) P256:(SHA1) P384: (SHA1) P521:(SHA1) K163: (SHA1, 224, 256, 384, 512) K233:(SHA1) K283:(SHA1) K409:(SHA1) K571:(SHA1) B163: (SHA1, 224, 256, 384, 512) B233:(SHA1) B283: (SHA1) B409:(SHA1) B571:(SHA1)
	[FIPS 186-2] RSA	SigGen and SigVer9.31: (1024/1536/2048/3072/4096 with SHA-1/SHA-256/SHA-384/SHA-512), SigGen and SigVerPKCS1.5 (1024/1536/2048/3072/4096 with all hash sizes), SigGen and SigVerPSS (1024/1536/2048/3072/4096 with all hash sizes)
Key Agreement	EC Diffie-Hellman	Non-compliant (untested) Diffie-Hellman scheme using elliptic curve, supporting all NIST defined B, K and P curves (except 163 and 192). Key agreement is a service provided for calling process use but is not used to establish keys into the Module.
Random Number Generation; Symmetric key generation	[ANS X9.31] RNG	AES 128/192/256
Key Encryption, Decryption	RSA	Using 1024- and 1536-bit keys
Key Encryption, Decryption	RSA	The RSA algorithm (with key sizes 2048 – 16384) may be used by the calling application for encryption o r decryption of keys. No claim is made for SP 800-56B compliance, and no CSPs are established into or exported out of the module using these services.
ECC CDH (CVL)	[SP 800-56A] (§5.7.1.2)	All NIST Recommended B, K and P curves sizes 163 and 192
Encryption, Decryption and CMAC	[SP 800-67]	3-Key Triple-DES TECB, TCBC, TCFB, TOFB; CMAC generate and verify

Table 3b – FIPS Non-Approved Cryptographic Functions

These algorithms shall not be used when operating in the FIPS Approved mode of operation. EC Diffie-Hellman Key Agreement provides a maximum of 256 bits of security strength. RSA Key Wrapping provides a maximum of 270 bits of security strength.

The module must be loaded into memory and the FIPS_module_mode_set() function called in order for an operator to successfully authenticate. Once in an operational state, the module can switch service by service between an Approved mode of operation and a non-Approved mode of operation. The module will transition to the non-Approved mode of operation when one of the above non-Approved security functions is utilized in lieu of an Approved one. The module can transition back to the Approved mode of operation by utilizing an Approved security function.

2.3 CRITICAL SECURITY PARAMETERS AND PUBLIC KEYS

All CSPs used by the Module in the Approved mode of operation are described in this section. All access to these CSPs by Module services are described in Section 4. The CSP names are generic, corresponding to API parameter data structures.

CSP Name	Description
RSA SGK	RSA (2048 to 16384 bits) signature generation key
ECDSA SGK	ECDSA (P curves 224-521, B and K curves 233-571) signature generation key
EC Diffie-Hellman Private	EC Diffie-Hellman (All NIST defined B, K, and P curves except 163 and 192) private key agreement key.
AES EDK	AES (128/192/256) encrypt / decrypt key
AES CMAC	AES (128/192/256) CMAC generate / verify key
AES GCM	AES (128/192/256) encrypt / decrypt / generate / verify key
HMAC Key	Keyed hash key (160/224/256/384/512)
Hash_DRBG CSPs	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
HMAC_DRBG CSPs	V (160/224/256/384/512 bits) and Key (160/224/256/384/512 bits), entropy input (length dependent on security strength)
CTR_DRBG CSPs	V (128 bits) and Key (AES 128/192/256), entropy input (length dependent on security strength)
CO-AD-Digest	Pre-calculated HMAC-SHA-1 digest used for Crypto Officer role authentication
User-AD-Digest	Pre-calculated HMAC-SHA-1 digest used for User role authentication
ECDH Shared Secret	Used to derive the TLS encryption key
TLS encryption key	Used to encrypt user data in TLS sessions

Table 4a – Critical Security Parameters

Authentication data is loaded into the module during the module build process, performed by an authorized operator (Crypto Officer), and otherwise cannot be accessed.

The Module does not output intermediate key generation values.

CSP Name	Description
RSA SVK	RSA (1024 to 16384 bits) signature verification public key
ECDSA SVK	ECDSA (All NIST defined B, K and P curves) signature verification key
EC Diffie-Hellman Public	EC Diffie-Hellman (All NIST defined B, K and P curves except 163 and 192) public key agreement key.

Table 4b – Public Keys

For all CSPs and Public Keys:

Storage: RAM, associated to entities by memory location. The Module stores the DRBG state values for the lifetime of the DRBG instance. The Module uses CSPs passed in by the calling application on the stack. The Module does not store any CSP persistently (beyond the lifetime of an API call), with the exception of DRBG state values used for the Modules' default key generation service.
Generation: The Module implements SP 800-90A compliant DRBG services for creation of symmetric keys, and for generation of elliptic curve and RSA keys as shown in Table 3a. The calling application is responsible for storage of generated keys returned by the Module.
Entry: All CSPs enter the Module's logical boundary in plaintext as API parameters, associated by memory location. However, none cross the physical boundary.
Output: The Module does not output CSPs, other than as explicit results of key generation services. However, none cross the physical boundary.

Destruction: Zeroization of sensitive data is performed automatically by API function calls for temporarily stored CSPs. In addition, the Module provides functions to explicitly destroy CSPs related to random number generation services. The calling application is responsible for parameters passed in and out of the Module.

Private and secret keys as well as seeds and entropy input are provided to the Module by the calling application and are destroyed when released by the appropriate API function calls. Keys residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the Module defined API. The operating system protects memory and process space from unauthorized access. Only the calling application that creates or imports keys can use or export such keys. All API functions are executed by the invoking calling application in a non-overlapping sequence such that no two API functions will execute concurrently. An authorized application as user (Crypto Officer and User) has access to all key data generated during the operation of the Module.
In the event Module power is lost and restored, the calling application must ensure that any AES-GCM keys used for encryption or decryption are re-distributed.
The module operator (the calling applications) shall use entropy sources that meet the security strength required for the random number generation mechanism: 128 bits for the DRNG and DRBG mechanisms. This entropy is supplied by means of callback functions. Those functions must return an error if the minimum entropy strength cannot be met.

2.4 CRYPTOGRAPHIC MODULE BOUNDARY

The logical boundary of the Module is the fipscanister object module, a single binary object module file fipscanister.o that is linked as part of libcrypto library

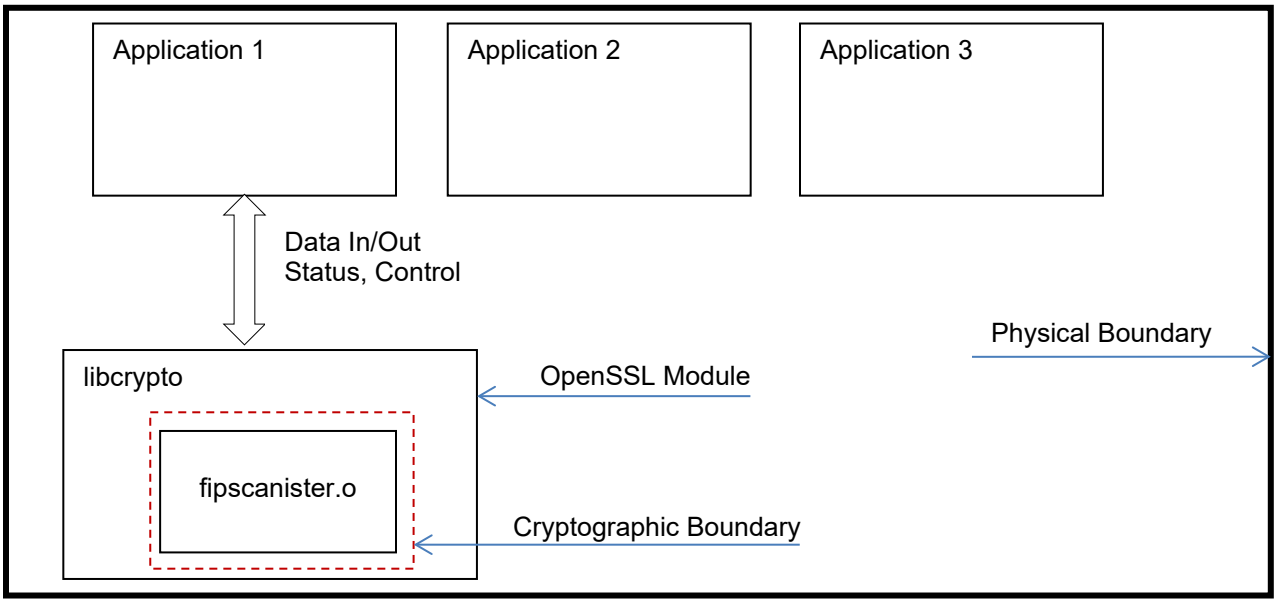

Figure 1 – Software Block Diagram

The physical boundary of the Module is the enclosure of the general-purpose computer which includes the processing unit i.MX53 Arm Cortex-A8 on which the Module is installed and executed.

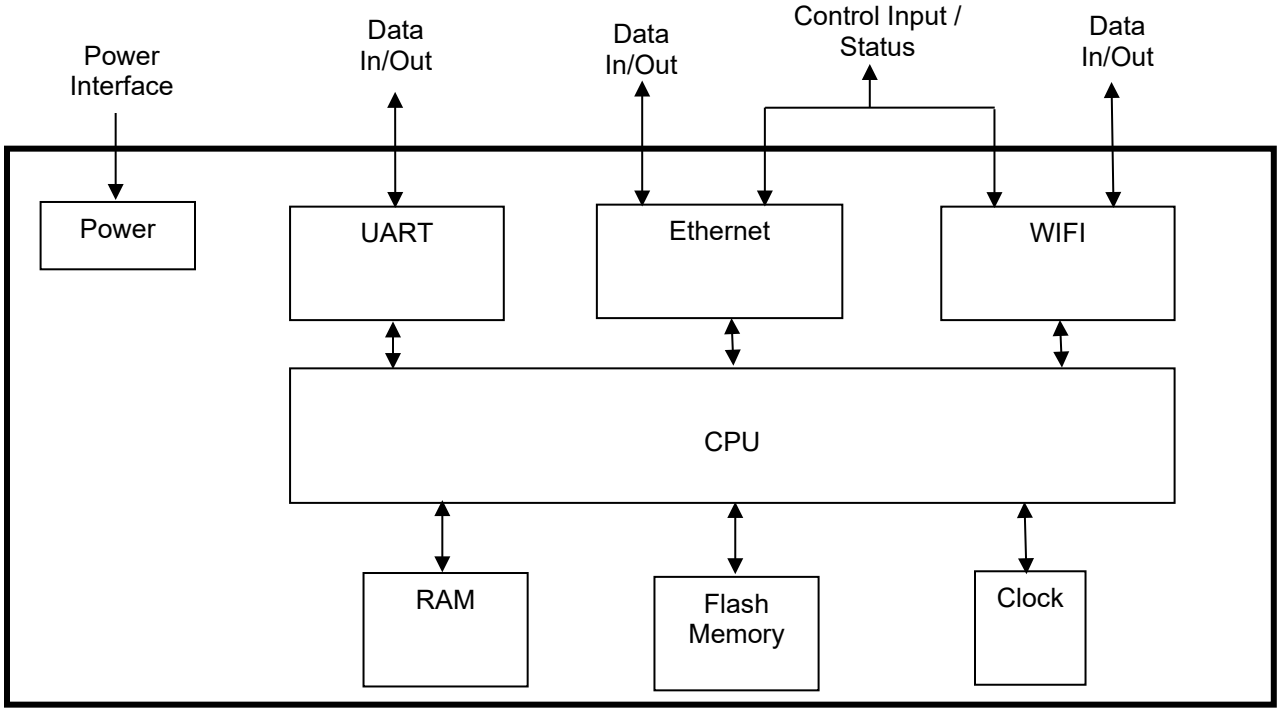

Figure 2 – Hardware Block Diagram

3 CRYPTOGRAPHIC MODULE PORTS AND INTERFACES

As a software-only module, the FIPS Cryptographic Module provides an API logical interface for invocation of FIPS140-2 approved cryptographic functions. The functions shall be called by the referencing application, which assumes the operator role during application execution. The API, through the use of input parameters, output parameters, and function return values, defines the four FIPS 140-2 logical interfaces: data input, data output, control input and status output.

The physical ports of module are the same as the device on which it is executing. The logical interface is an application program interface (API):

- Data Input - API entry point data input stack parameters
- Data Output - API entry point data output stack parameters
- Status Output - API entry point return values and status stack parameters
- Control Input - API entry point and corresponding stack parameters

As a software module, control of the physical ports is outside module scope. However, when the Module is performing self-tests, or is in an error state, all output on the logical data output interface is inhibited. The module is single-threaded and in error scenarios returns only an error value (no data output is returned).

4 ROLES, SERVICES AND AUTHENTICATION

The Module implements the required User and Crypto Officer roles and requires authentication for those roles. Only one role may be active at a time and the Module does not allow concurrent operators.

The User or Crypto Officer role is assumed by passing the appropriate password to the FIPS_module_mode_set() function.

The password values may be specified at build time and must have a minimum length of 16 characters. Any attempt to authenticate with an invalid password will result in an immediate and permanent failure condition rendering the Module unable to enter the FIPS mode of operation, even with subsequent use of a correct password.

Authentication data is loaded into the Module during the Module build process, performed by the Crypto Officer, and otherwise cannot be accessed. Since minimum password length is 16 characters, the probability of a random successful authentication attempt in one try is a maximum of $1/256^{16}$, or less than $1/10^{6}$. The Module permanently disables further authentication attempts after a single failure, so this probability is independent of time. Both roles have access to all of the services provided by the Module.

User Role (User): Loading the Module and calling any of the API functions.

Crypto Officer Role (CO): Installation of the Module on the host computer system and calling of any API functions.

All services implemented by the Module are listed below, along with a description of service CSP access. The access modes shown in Table 5 are defined as follows:

- Generate (G): Generates the Critical Security Parameter (CSP_ using an approved Random Bit Generator
- Read (R): Exports the CSP
- Write (W): Enter/establish and stores a CSP
- Destroy (D): Overwrites the CSP
- Execute (E): Employs the CSP

Service	Role	Description
Initialize	User, CO	Module initialization. **E:** CO-AD-Digest, User-AD-Digest
Self-test	User, CO	Perform self-tests (FIPS_selftest). Does not access CSPs.
Show status	User, CO	Functions that provide module status information: • Version (as unsigned long or const char *) • FIPS Mode (Boolean) Does not access CSPs
Zeroize	User, CO	Functions that destroy CSPs: • fips_drbg_uninstantiate: **D:** DRBG CSPs (Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs) All other services automatically overwrite CSPs stored in allocated memory. Stack cleanup is the responsibility of the calling application.
Random number generation	User, CO	Used for random number and symmetric key generation. • Determine security strength of a DRBG instance • Obtain random data Uses and updates **E:** Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs.
Asymmetric key generation	User, CO	Used to generate ECDSA keys and complies with SP800-133 Rev 2 Sect. 5.1. **G:** ECDSA SGK, ECDSA SVK
Symmetric encrypt/decrypt	User, CO	Used to encrypt or decrypt data. **E:** AES EDK, AES GCM, AES
Symmetric key generation	User, CO	Used to generate AES keys and complies with SP800-133 Rev 2 Sect. 6.1. **G:** AESCMAC, AESGCM, AES-CBC, AES-CCM, AES-CFB1, AESCFB128, AES-CFB8, AES-CTR, AES-ECB, AES-OFB
Symmetric digest	User, CO	Used to generate or verify data integrity with CMAC. **E:** AES CMAC.
Message digest	User, CO	Used to generate a SHA1 or SHA2 message digest. Does not access CSPs.
Keyed Hash	User, CO	Used to generate or verify data integrity with HMAC.

		E: HMAC Key (passed in by the calling process).
Key agreement	User, CO	Used to perform key agreement primitives on behalf of the calling process (does not establish keys into the module). **E:** EC Diffie-Hellman Private, EC Diffie-Hellman Public (passed in by the calling process).
Digital signature	User, CO	Used to generate or verify RSA or ECDSA digital signatures. **E:** RSA SGK, RSA SVK; ECDSA SGK, ECDSA SVK (passed in by the calling process).
Utility	User, CO	Miscellaneous helper functions. Does not access CSPs.

Table 5 – Services and CSP Access

[1] "Key transport" can refer to a) moving keys in and out of the module, or b) the use of keys by an external application. The latter definition is the one that applies to the OpenSSL FIPS Object Module.

The Module supports the following non-Approved Services:

- Random number generation - using ANSI X9.31, Dual EC DRBG
- Asymmetric key generation - Using FIPS 186-2 key generation mechanisms
- Symmetric encrypt/decrypt – using Triple DES and AES XTS
- Key transport using RSA 1024- and 16384-bit keys
- Key agreement - Using curve sizes 163 and 192
- Digital signature - Using RSA, DSA, ECDSA and non-Approved key sizes, curves and digest sizes
- Utility

5 PHYSICAL SECURITY

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

6 OPERATIONAL ENVIRONMENT

The tested operating systems segregate user processes into separate process spaces. Each process space is logically separated from all other processes by the operating system software and hardware. The Module functions entirely within the process space of the calling application, and implicitly satisfies the FIPS 140-2 requirement for a single user mode of operation. Please see Table 2 for the module Operational Environment.

7 SELF TEST

The Module performs the self-tests listed below on invocation of Initialize or Self-test.

Algorithm	Type	Test Attributes
Software integrity	KAT	HMACSHA1
HMAC	KAT	One KAT per SHA1, SHA224, SHA256, SHA384 and SHA512 Per IG 9.3, this testing covers SHA POST requirements.
AES	KAT	Separate encrypt and decrypt, ECB mode, 128-bit key length
AES CCM	KAT	Separate encrypt and decrypt, 192 key length
AES GCM	KAT	Separate encrypt and decrypt, 256 key length
AES-XTS[5]	KAT	Separate encrypt and decrypt, 128, 256-bit key sizes to support either the 256-bit key size (for XTS AES128) or the 512-bit key size (for XTSAES256)
AES CMAC	KAT	Sign and verify CBC mode, 128, 192, 256 key lengths
Triple-DES[5]	KAT	Separate encrypt and decrypt, ECB mode, 3Key
Triple-DES CMAC[5]	KAT	CMAC generate and verify, CBC mode, 3Key
RSA	KAT	Sign and verify using 2048 bit key, SHA256
DSA[5]	PCT	Sign and verify using 2048 bit key, SHA384
DRBG	KAT	CTR_DRBG: AES, 256-bit with and without derivation function HASH_DRBG: SHA256 HMAC_DRBG: SHA256
ECDSA	PCT	Keygen, sign, verify using P224, K233 and SHA512. The K233 self-test is not performed for operational environments that support prime curve only.
ECC CDH	KAT	Shared secret calculation per SP 80056A §5.7.1.2, IG 9.6 (Primitive "Z" Computation KAT)
TLS KDF	KAT	KAT for TLSv1.2 (SHA256 and SHA384)

Table 6 – Power On Self-Tests (KAT = Known answer test; PCT = Pairwise consistency test)

Note: No parts of TLS protocol, other than the KDF, have been tested by the CAVP and CMVP.

The Module performs the following Conditional Self-Tests:

i. Continuous (RNG) Tests: ANSI X9.31 RNG[5], DRBG

ii. SP800-90A Section 11.3: DRBG health checks (instantiate, generate and reseed)

iii. Pair-wise consistency test (Signature Generation/Verification or Encryption/Decryption) on each generation of a key pair: RSA, ECDSA, DSA[5]

The FIPS_mode_set() is called from DEP (as per FIPS140-2 IG 9.10) of libcrypto library as it is being loaded by OS. The FIPS_mode_set() API calls module function FIPS_module_mode_set(), which performs all power-up self-tests listed above with no operator intervention required. It returns a "1" if all power-up self-tests succeed, and a "0" otherwise. If any component of the power-up self-test fails an internal flag is set to prevent subsequent invocation of any cryptographic function calls. The Module will only enter the FIPS Approved mode if the Module is reloaded and the call to FIPS_mode_set() succeeds. The power-up self-tests may also be performed OnDemand by calling FIPS_selftest(), which returns a "1" for success and "0" for failure. Interpretation of this return code is the responsibility of the calling application.

The Status of the FIPS_mode_set() call is indicated by the debug message available via ADB debug shell:

[5] These are non-approved algorithms as they are not ACVP tested.

"FIPS Mode Set - Successful" - in case of success

"FIPS Mode Set - Failed" - in case of failure

8 MITIGATION OF OTHER ATTACKS

The Module is not designed to mitigate against attacks which are outside of the scope of FIPS 140-2

9 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
FSM	Finite State Machine
HMAC	Hash Message Authentication Code
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
OFB	Output Feedback
OS	Operating System
OTA	Over The Air
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SOF	Strength of Function
SVT	Scenario Verification Testing
Triple-DES	Triple Data Encryption Standard
UI	User Interface

10 REFERENCES

Reference	Full Specification Name
[ANS X9.31]	Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)
[FIPS 140-2]	Security Requirements for Cryptographic modules, May 25, 2001
[FIPS 180-4]	Secure Hash Standard
[FIPS 186-4]	Digital Signature Standard
[FIPS 197]	Advanced Encryption Standard
[FIPS 198-1]	The KeyedHash Message Authentication Code (HMAC)
[SP 800-38B]	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
[SP 800-38C]	Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality
[SP 800-38D]	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC
[SP 800-56A]	Recommendation for PairWise Key Establishment Schemes Using Discrete Logarithm Cryptography
[SP 800-67R1]	Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher
[SP 800-89]	Recommendation for Obtaining Assurances for Digital Signature Applications
[SP 800-90Ar1]	Recommendation for Random Number Generation Using Deterministic Random Bit Generators
[SP 800-131A]	Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths

END OF DOCUMENT