Security Builder® FIPS Module

Version 6.0, 6.0.2 and 6.0.3

FIPS 140-2 Non-Proprietary
Security Policy

Certicom Corp.

May 18, 2016

Contents

1 Introduction

1.1 Overview

This is a non-proprietary Federal Information Processing Standard (FIPS) 140-2 Security Policy for Certicom's **Security Builder® FIPS Module Version 6.0, 6.0.2 and 6.0.3** (SB FIPS Module). SB FIPS Module is a cryptographic toolkit for C language users, providing services of various cryptographic algorithms such as hash algorithms, encryption schemes, message authentication, and public key cryptography. This Security Policy specifies the rules under which SB FIPS Module must operate. These security rules are derived from the requirements of FIPS 140-2 [1], and related documents [6, 7, 8].

1.2 Purpose

This Security Policy is created for the following purposes:

1. It is required for FIPS 140-2 validation.
2. To outline SB FIPS Module's conformance to FIPS 140-2 Level 1 Security Requirements.
3. To provide users with how to configure and operate the cryptographic module in order to comply with FIPS 140-2.

1.3 References

References

[1] NIST *Security Requirements For Cryptographic Modules, FIPS PUB 140-2,* December 3, 2002.

[2] NIST *Security Requirements For Cryptographic Modules, Annex A: Approved Security Functions for FIPS PUB 140-2*, Draft, July 26, 2011.

[3] NIST *Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2*, Draft, August 12, 2011.

[4] NIST *Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2*, Draft, July 26, 2011.

[5] NIST *Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2*, Draft, July 26, 2011.

[6] NIST *Derived Test Requirements for FIPS 140-2*, Draft, January 4, 2011.

[7] NIST *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program,* July 15, 2011.

[8] NIST *Frequently Asked Questions for the Cryptographic Module Validation Program*, December 4, 2007.

1.4 Change History

Change history is recorded in Table 1.

Table 1: Change History

Revision	Date	Author	Description
0.1	2011/07/13	A.Y.	Initial revision. Created based on the Security Policy for SB FIPS Module 5.6.
0.2	2011/08/17	A.Y.	Updated platform list. Also applied some editorial fixes.
0.3	2011/08/17	A.Y.	Further editorial correction.
0.4	2011/09/08	A.Y.	Update on the algorithm standard listing. Also made editorial correction on the platform listing.
0.5	2011/09/29	A.Y.	Editorial corrections for consistency.
0.6	2011/09/29	A.Y.	Added algorithm certificate numbers. Also editorial corrections.
0.7	2012/02/10	K.O.	Editorial Corrections.
0.8	2012/04/03	K.O.	Editorial Corrections.
0.9	2014/09/16	R.T.	Changes for version 6.0.2.
0.10	2014/09/17	R.T.	Changed key sizes/strengths due to NIST recommendation.
0.11	2014/10/03	R.T.	Updated platform list.
0.12	2014/10/16	R.T.	Incorporated comments from CGI and added new algorithm certificate numbers.
0.13	2014/10/17	R.T.	Corrected certificate number for ECDSA.
0.14	2014/10/20	R.T.	Corrected minor mistake in platform list.
0.15	2014/11/27	R.T.	Updated version number.
0.16	2015/04/10	R.T.	Additional platforms.
0.17	2015/09/30	H.W.	Additional platforms.
0.18	2015/11/27	H.W.	Correct iOS version 8.1, ARMv8 to iOS version 8.0, ARMv8
0.19	2016/03/01	R.T.	Updated for for NIST SP 800-131A transition.
0.20	2016/05/04	R.T.	Updated for version 6.0.3.
0.21	2016/05/18	R.T.	Addressed comments from CMVP.

2 Cryptographic Module Specification

SB FIPS Module is a multiple-chip standalone software cryptographic module in the form of an object that operates with the following components:

- Commercially available general-purpose computer hardware.
- Commercially available Operating System (OS) that runs on the computer hardware.

2.1 Physical Specifications

The general-computer hardware component consists of the following devices:

1. CPU (Microprocessor)
2. Memory
    (a) Working memory is located on the RAM containing the following spaces:
        i. Input/output buffer
        ii. Plaintext/ciphertext buffer
        iii. Control buffer

    Key storage is not deployed in this module.

    (b) Program memory is also located on RAM.
3. Hard Disk (or disks)
4. Display Controller, including Touch Screen Controller
5. Keyboard Interface
6. Mouse Interface, including Trackball Interface
7. Audio Controller
8. Network Interface
9. Serial Interface
10. Parallel Interface
11. USB Interface
12. Power Supply

The configuration of this component is illustrated in Figure 1.

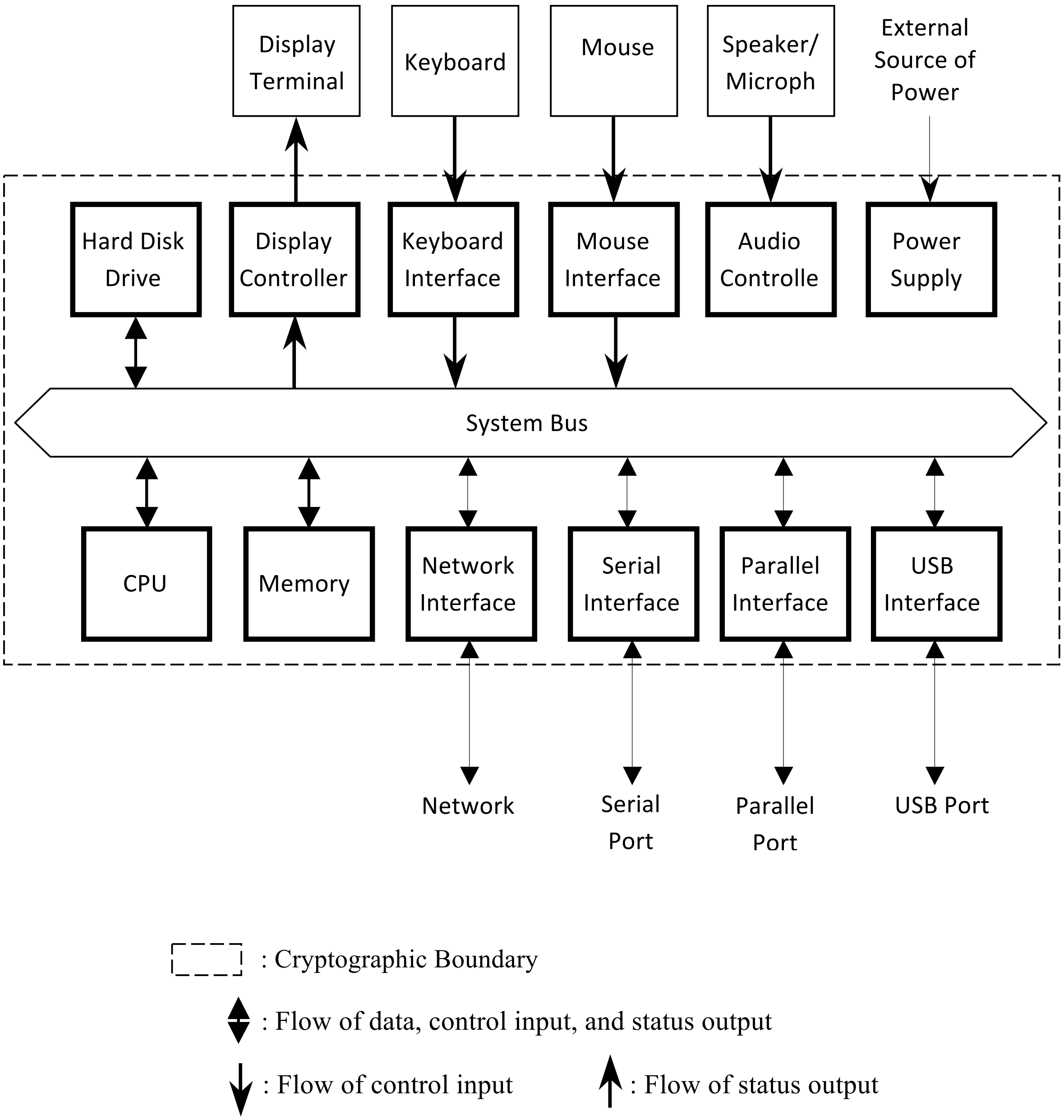

Figure 1: Cryptographic Module Hardware Block Diagram

2.2 Computer Hardware and OS

The combinations of computer hardware and OS include the following representative platforms.

For version 6.0:

1. QNX Neutrino 6.6, ARMv7 (Binary compatible to QNX Neutrino 6.5)
2. QNX Neutrino 6.5 x86
3. Red Hat Linux AS 5.6 32-bit x86 (Binary compatible to AS 4.x/5.0-5.5)
4. Red Hat Linux AS 5.6 64-bit x86 (Binary compatible to AS 4.x/5.0-5.5)

For version 6.0.2:

1. Android 4.4.2, ARMv7
2. Android 4.0.4, x86
3. iOS version 6.1.4, ARMv7
4. Windows Phone 8.0, ARMv7
5. Windows 7 Enterprise, 64-bit x86
6. iOS version 6.1.4, ARMv7s
7. Android 5.0.1, ARMv8
8. iOS version 8.0, ARMv8
9. Windows 7 Enterprise, 32-bit x86
10. Centos Linux Release 7.1, 64-bit x86
11. MacOS X Yosemite 10.10.4, 64-bit x86

For version 6.0.3:

1. MacOS X El Capitan 10.11.4, 64-bit x86

SB FIPS Module is also suitable for any platforms of any manufactures with compatible processors and equivalent or larger system configurations, and compatible OS versions. For example, an identical SB FIPS Module can be used on any compatible Linux or Windows for x86 processors, or iOS for ARMv7 processors. SB FIPS Module will run on such platforms and OS versions while maintaining its compliance to the FIPS 140-2 Level 1 requirements.

2.3 Software Specifications

SB FIPS Module is manufactured by Certicom Corp., providing services to the C computer language users in object format.

The interface into SB FIPS Module is via Application Programmer's Interface (API) function calls. These function calls provide the interface to the cryptographic services, for which the parameters and return codes provide the control input and status output (see Figure 2).

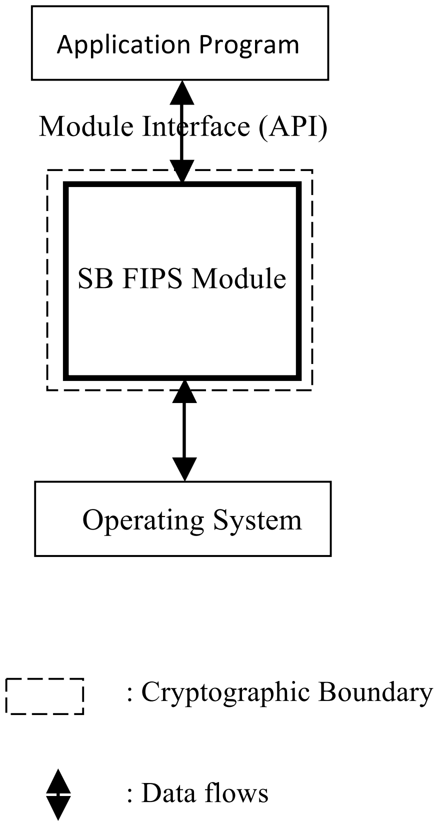

Figure 2: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The physical and logical interfaces are summarized in Table 2.

Table 2: Logical and Physical Interfaces

I/O	Logical Interface	Physical Interface
Data Input	API	Ethernet port
Data Output	API	Ethernet port
Control Input	API	Keyboard and Mouse
Status Output	Return Code	Display
Power Input	Initialization Function	The Power Supply is the power interface.
Maintenance	Not supported	Not supported

4 Roles, Services, and Authentication

4.1 Roles

SB FIPS Module supports Crypto Officer and User Roles (see Table 3). These roles are enforced by this Security Policy.

Table 3: Roles and Services

Service	Crypto Officer	User
Initialization, etc.		
Initialization	X	X
Deinitialization	X	X
Self-tests	X	X
Show status	X	X
Symmetric Ciphers (AES and TDES)		
Key generation	X	X
Encrypt	X	X
Decrypt	X	X
Key zeroization	X	X
Hash Algorithms and Message Authentication (SHA, HMAC)		
Hashing	X	X
Message Authentication	X	X
Random Number Generation (pRNG)		
Instantiation	X	X
Seeding	X	X
Request	X	X
CSP/Key zeroization	X	X
Digital Signature (DSA, ECDSA, RSA)		
Key pair generation	X	X
Sign	X	X
Verify	X	X
Key zeroization	X	X
Key Establishment (DH, ECDH, ECMQV, RSA)		
Key pair generation	X	X
Shared secret generation	X	X
Wrap	X	X
Unwrap	X	X
Key zeroization	X	X

In order to operate the module securely, it is the Crypto Officer and User's responsibility to confine calls to those methods that have been FIPS 140-2 Approved. Thus, in the approved mode of operation, all Roles shall confine themselves to calling FIPS Approved algorithms, as marked in Table 4.

4.2 Services

SB FIPS Module supports many cryptographic algorithms. The set of cryptographic algorithms supported by SB FIPS Module is given in Table 4.

Table 4: Supported Algorithms and Standards

	Algorithm	FIPS Approved or Allowed	Cert # (6.0)	Cert # (6.0.2)	Cert # (6.0.3)
Block Ciphers	TDES (ECB, CBC, CFB64, OFB64 [FIPS 46-3]	X	#1159	#1773	#2164
	AES (ECB, CBC, CFB128, OFB128, CTR, CCM, GCM, CMAC, XTS) [FIPS 197]	X	#1789	#3029	#3946
	DES (ECB, CBC, CFB64, OFB64)				
	DESX (ECB, CBC, CFB64, OFB64)				
	AES (CCM*) [ZigBee 1.0.x]				
	AES EAX [ANSI C12.22]				
	ARC2 (ECB, CBC, CFB64, OFB64) [RFC 2268]				
Stream Cipher	ARC4				
Hash Functions	SHA-1 [FIPS 180-4]	X	#1571	#2530	#3256
	SHA-224 [FIPS 180-4]	X	#1571	#2530	#3256
	SHA-256 [FIPS 180-4]	X	#1571	#2530	#3256
	SHA-384 [FIPS 180-4]	X	#1571	#2530	#3256
	SHA-512 [FIPS 180-4]	X	#1571	#2530	#3256
	MD5 [RFC 1321]				
	MD4 [RFC 1320]				
	MD2 [RFC 1115]				
	AES MMO [ZigBee 1.0.x]				
Message Authentication	HMAC-SHA-1 [FIPS 198]	X	#1054	#1914	#2571
	HMAC-SHA-224 [FIPS 198]	X	#1054	#1914	#2571
	HMAC-SHA-256 [FIPS 198]	X	#1054	#1914	#2571
	HMAC-SHA-384 [FIPS 198]	X	#1054	#1914	#2571
	HMAC-SHA-512 [FIPS 198]	X	#1054	#1914	#2571
	HMAC-MD5 [RFC 2104]				
	AES-XCBC-MAC [RFC 3566]				
pRNG	DRBG [NIST SP 800-90]	X	#127	#579	#1151
	ANSI X9.62 RNG [ANSI X9.62]				
	ANSI X9.31 RNG [ANSI X9.31]				
Digital Signature	DSS [FIPS 186-4]	X	#563	#891	#1076
	ECDSA [FIPS 186-4, ANSI X9.62]	X	#242	#553	#866
	RSA PKCS1 v1.5 [FIPS 186-4, PKCS #1 v2.1]	X	#894	#1574	#2017
	RSA PSS [FIPS 186-4, PKCS #1 v2.1]	X	#894	#1574	#2017
	ECNR [IEEE 1363]				
	ECQV				
Key Agreement	DH [NIST SP 800-56A] Security strength >= 112 bits	X	#25	#50	#79
	DH [NIST SP 800-56A] Security strength < 112 bits				
	ECDH [NIST SP 800-56A] Component (ECC CDH)	X	#25 #7	#50 #67	#789
	ECMQV [NIST SP 800-56A]	X	#25	#50	#79
	ECPVS [ANSI X9.92]				
	ECSPEKE [IEEE 1363.2]				

	Algorithm	FIPS Approved or Allowed	Cert # (6.0)	Cert # (6.0.2)	Cert # (6.0.3)
Key Wrapping	RSA PKCS1 v1.5 [PKCS #1 v2.1]	X			
	RSA OAEP [NIST SP 800-56B]				
	RSA KEM [ANSI X9.44]				
	ECIES [ANSI X9.63]				

The 3-key TDES, AES (ECB, CBC, CFB128, OFB128, CTR, CCM, GCM, CMAC, and XTS modes), SHS (SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512), HMAC-SHS (HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA256, HMAC-SHA-384, and HMAC-SHA-512), pRNG (NIST SP 800-90), DSA, ECDSA, RSA PKCS #1 v1.5 Signature, RSA PSS algorithms, and NIST SP 800-56A Key Establishment (key agreement) techniques, DH with strength >= 112 bits, ECDH, and ECMQV, have been validated to comply with FIPS. SB FIPS Module also supports a FIPS Allowed Key Establishment technique, RSA #1 v1.5 Encryption algorithm. In order to operate the module in compliance with FIPS, only these FIPS Approved or allowed algorithms should be used.

The DES, DESX, AES CCM* (CCM Star) and EAX modes, pRNG (ANSI X9.62 and ANSI X9.31), ARC2, ARC4, MD5, MD4, MD2, AES MMO, HMAC-MD5, AES-XCBC-MAC, ECNR, ECQV, ECIES, ECPVS, ECSPEKE, key establishment (key wrapping) techniques, RSA OAEP and RSA KEM, and DH with strength < 112 bits, are supported as non FIPS Approved algorithms. In order to operate the module in compliance with FIPS, these algorithms should not be used.

Please be advised that 2-Key Triple-DES decryption is permitted for legacy purposes. 2-Key Triple-DES encryption is now considered a non-FIPS Approved algorithm as of January 1, 2016. Please see NIST SP 800-131A for more information.

Table 5 summarizes the keys and CSPs used in the FIPS mode.

Table 5: Key and CSP, Key Size, Security Strength, and Access

Algorithm	**Key and SP**	**Key Size**	**Strength**	**Access**
AES	key	128-256 bits	128-256 bits	Create, Read, Use
TDES	key	168 bits	112 bits	Create, Read, Use
HMAC	key	160-512 bits	128-256 bits	Use
pRNG (DRBG)	seed	112-256 bits	112-256 bits	Use
DSA	key pair	2048-15360 bits	112-256 bits	Create, Read, Use
ECDSA	key pair	224-521 bits	112-256 bits	Create, Read, Use
RSA Signature	key pair	2048-15360 bits	112-256 bits	Create, Read, Use
DH	static/ephemeral key pair	2048-15360 bits	112-256 bits	Create, Read, Use
ECDH	static/ephemeral key pair	224-521 bits	112-256 bits	Create, Read, Use
ECMQV	static/ephemeral key pair	224-521 bits	112-256 bits	Create, Read, Use
RSA Key wrapping	key pair	224-15360 bits	112-256 bits	Create, Read, Use

Note:
DH (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength; non-compliant less than 112-bits of encryption strength)

ECDH (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength; non-compliant less than 112-bits of encryption strength)

ECMQV (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength; non-compliant less than 112-bits of encryption strength)

RSA (key wrapping; key establishment methodology provides between 112 and 256 bits of encryption strength; non-compliant less than 112-bits of encryption strength)

Digital signature generation that provides less than 112 bits of security (using RSA, DSA or ECDSA) is disallowed beginning January 1st, 2014.

Digital signature generation using SHA-1 as its underlying hash function is disallowed beginning January 1st, 2014.

HMAC-SHA-1 shall have a key size of at least 112 bits

4.3 Operator Authentication

SB FIPS Module does not deploy authentication mechanism. The roles of Crypto Officer and User are implicitly selected by the operator.

5 Finite State Model

The Finite State model contains the following states:

- Installed/Uninitialized
- Initialized
- Self-Test
- Idle
- Crypto Officer/User
- Error

The following is the important features of the state transition:

1. When the module is installed by the Crypto Officer, the module is in the In- stalled/Uninitialized state.

2. When the initialization command is applied to the module, i.e., the module is loaded on the memory, turning to the Initialization state. Then, it transits to the Self-Test state automatically, running the Power-up Tests. While in the Self-Test state, all data output via the data output interface is prohibited. On success the module enters Idle; on failure the module enters Error and the module is disabled. From the Error state the Crypto Officer may need to re-install to attempt correction.

3. From the Idle state (which is only entered if self-tests have succeeded), the module can transit to the Crypto Officer/User state when an API function is called.

4. When the API function has completed successfully, the state transits back to Idle.

5. If the Conditional Test (Continuous RNG Test or Pair-wise Consistency Test) fails, the state transits to Error and the module is disabled.

6. When On-demand Self-test is executed, the module enters the Self-Test state. On success the module enters Idle; on failure the module enters Error and the module is disabled.

7. When the de-initialization command is executed, the module goes back to the Installed/Uninitialized state.

6 Physical Security

Physical security is not applicable to this software module at Level 1 Security.

7 Operational Environment

This module is designed for commercially available general purpose computer operating systems such as Linux, Windows, Android, iOS or QNX. These operating systems provide modifiable environment.

This module is to be run in single user operational environment, where each user application runs in virtually separated independent space. Note that modern Operating Systems such as Linux, Windows, Android, iOS or QNX provide such operational environment.

8 Cryptographic Key Management

SB FIPS Module provides the underlying functions to support FIPS 140-2 Level 1 key management. The user will select FIPS Approved algorithms and will handle keys with appropriate care to build up a system that complies with FIPS 140-2. It is the Crypto Officer and User's responsibility to select FIPS 140-2 validated algorithms (see Table 4).

8.1 Key Generation

SB FIPS Module provides FIPS 140-2 compliant key generation. The underlying random number generation uses a FIPS Approved method, a DRBG (Hash, HMAC and Counter).

The module also supports Dual_EC DRBG, ANSI X9.62 and ANSI X9.31 RNGs, however, the use of Dual_EC DRBG or ANSI X9.62/ANSI X9.31 RNGs is non-approved for key generation. No keys generated using the Dual_EC DRBG or ANSI X9.62/ANSI X9.31 RNGs can be used to protect sensitive data in the Approved mode. Any random output in Approved mode using these DRBG/RNGs is equivalent to plaintext.

8.2 Key Establishment

SB FIPS Module provides the following FIPS Approved or allowed key establishment techniques [5]:

1. Diffie-Hellman (DH)
2. EC Diffie-Hellman (ECDH)
3. ECMQV
4. RSA PKCS1 v1.5

The ECDH and ECMQV key agreement technique implementations support elliptic curve sizes from 163 bits to 521 bits that provides between 80 and 256 bits of security strength, where 224 bits and above must be used to provide a minimum of 112 bits of security in the FIPS mode. The DH key agreement technique implementation supports modulus sizes from 512 bits to 15360 bits that provides between 56 and 256 bits of security strength, where 2048 bits and above must be used to provide a minimum of 112 bits of security in the FIPS mode. The RSA PKCS v1.5 key wrapping implementation supports modulus sizes from 512 bits to 15360 bits that provides between 56 and 256 bits of security, where 2048 bits and above must be used to provide minimum of 112 bits of security in the FIPS mode.

It is responsibility of the application to ensure that the appropriate key establishment techniques are applied to the appropriate keys.

8.3 Key Entry and Output

Keys must be imported or exported from the cryptographic boundary in encrypted form using a FIPS Approved algorithm.

8.4 Key Storage

SB FIPS Module is a low-level cryptographic toolkit, and as such does not provide key storage.

8.5 Zeroization of Keys

SB FIPSModule provides zeroizable interfaces which implement zeroization functions (see Table 3). Zeroization of keys and CSPs must be performed by calling the destroy functions of the objects when no longer needed; otherwise SB FIPS Module will not be functional.

9 Self-Tests

9.1 Power-up Tests

9.1.1 Tests upon Power-up

Self-tests are initiated automatically by the module at start-up. The following tests are applied:

1. **Known Answer Tests (KATs):**
   KATs are performed on TDES, AES, AES GCM, SHS (via HMAC-SHS), HMAC-SHS, DRBG, ANSI X9.62 RNG, ANSI X9.31 RNG, RSA Signature Algorithm, and KDF. For DSA and ECDSA, Pair-wise Consistency Test is used. For DH, ECDH, ECMQV, the underlying arithmetic implementations are tested via DSA and ECDSA tests.

2. **Software Integrity Test:**
   The software integrity test deploys ECDSA signature validation to verify the integrity of the module.

9.1.2 On-Demand Self-Tests

On-demand self tests may be invoked by the Cryptographic Officer or User by invoking a function, which is described in the Crypto Officer And User Guide in Appendix A.

9.2 Conditional Tests

The Continuous RNG Test is executed on all RNG generated data, examining the first 160 bits of each requested random generation for repetition. This ensures that the RNG is not stuck at any constant value.

Upon each generation of a DSA, ECDSA, or RSA key pair, the generated key pair is tested of their correctness by generating a signature and verifying the signature on a given message as a Pair-wise Consistency Test.

Upon generation or reception of DH, ECDH, or ECMQV key pair, the full key validation is performed upon reception, and SP 800-56A conformant computation is performed upon key generation.

9.3 Failure of Self-Tests

Failure of the Self-tests places the cryptographic module in the Error state, wherein no cryptographic operations can be performed. If any Self-test fails, the cryptographic module will output error code, and goes into the Error state.

10 Design Assurance

10.1 Configuration Management

A configuration management system for the cryptographic module is employed and has been described in a document to the testing laboratory. It uses Subversion (SVN) to track the configurations.

10.2 Delivery and Operation

Please refer to Section A.1 of Crypto Officer And User Guide in Appendix A to review the steps necessary for the secure installation and initialization of the cryptographic module.

10.3 Development

Detailed design information and procedures have been described in documentation submitted to the testing laboratory. The source code is fully annotated with comments, and is also submitted to the testing laboratory.

10.4 Guidance Documents

Crypto Officer Guide And User Guide is provided in Appendix A. This appendix outlines the operations for Crypto Officer and User to ensure the security of the module.

11 Mitigation of Other Attacks

SB FIPS Module implements mitigation of the following attacks:

1. Timing Attack on RSA
2. Attack on biased private key of DSA

11.1 Timing Attack on RSA

When employing Montgomery computations, timing effects allow an attacker to tell when the base of exponentiation is near the secret modulus. This leaks information concerning the secret modulus.

In order to mitigate this attack, the following is executed: The bases of exponentiation are randomized by a novel technique that requires no inversion to remove (unlike other blinding methods e.g. BSAFE Crypto-C User Manual v 4.2).

Note that Remote Timing Attacks are practical:
http://crypto.stanford.edu/ dabo/papers/ssl-timing.pdf

11.2 Attack on Biased Private Key of DSA

The standards for choosing ephemeral values in El-Gamal type signatures introduce a slight bias. Means to exploit these biases were presented to ANSI by D. Bleichenbacher.

In order to mitigate this attack, the following is executed: The bias in the RNG is reduced to levels which are far below the Bleichenbacher attack threshold.

Change Notice 1 of FIPS 186-2 is published to mitigate this attack:
http://csrc.nist.gov/CryptoToolkit/tkdigsigs.html

A Crypto Officer And User Guide

A.1 Installation

In order to carry out a secure installation of SB FIPS Module, the Crypto Officer must follow the procedure described in this section.

A.1.1 Installing

The Crypto Officer is responsible for the installation of SB FIPS Module. Only the Crypto Officer is allowed to install the product.

Place the object in an appropriate location on the computer hardware for your development environment.

A.1.2 Uninstalling

Remove the object from the computer hardware.

A.2 Commands

A.2.1 Initialization

sbg_FIPS140Initialize()
This function runs a series of self-tests on the module. These tests examine the integrity of the object, and the correct operation of the cryptographic algorithms. If these tests are successful, a value of *SB_SUCCESS* will be returned and the module will be enabled.

A.2.2 De-initialization

sbg_FIPS140Deinitialize()
This function de-initializes the module.

A.2.3 Self-Tests

sbg_FIPS140RunTest()
This function runs a series of self-tests, and return *SB_SUCCESS* if the tests are successful. These tests examine the integrity of the object, and the correct operation of the cryptographic algorithms. If these tests fail, the module will be disabled. Section A.3 of this document describes how to recover from the disabled state.

A.2.4 Show Status

sbg_FIPS140GetState()
This function will return the current state of the module.

A.3 When Module is Disabled

When SB FIPS Module becomes disabled, attempt to bring the module back to the Installed state by calling *sbg_FIPS140Deinitialize(),* and then to initialize the module using *sbg_FIPS140Initialize()*. If the initialization is successful, the module is recovered. If this attempt fails, uninstall the module and re-install it. If the module is initialized successfully by this re-installation, the recovery is successful. If this recovery attempt fails, it indicates a fatal error. Please contact Certicom Support immediately.