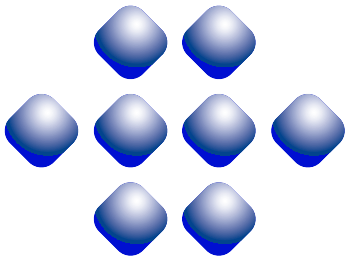
Security Builder® FIPS Module

Version 2.4

FIPS 140-2 Non-Proprietary
Security Policy

Certicom Corp.

September 22, 2008



certicom™

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 2.4** (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*, December 18, 2007.

[3] NIST *Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2*, June 14, 2007.

[4] NIST *Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2*, October 18, 2007.

[5] NIST *Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2*, January 16, 2008.

[6] NIST *Derived Test Requirements for FIPS 140-2*, Draft, March 24, 2004.

[7] NIST *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*, May 22, 2008.

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

1.4 Change Notes

The following are placed here by RCS upon check-in.

```
$Log: FIPSModule24SecurityPolicy.tex,v $
Revision 1.8.10.11  2008/09/22 13:56:42  ayamada
Added Nokia's multimedia computer.

Revision 1.8.10.10  2007/11/26 14:40:59  ayamada
Correction on key sizes and security strength.

Revision 1.8.10.9  2007/11/23 18:41:09  ayamada
Added key size and security strength in Table 4.

Revision 1.8.10.8  2007/11/20 15:16:04  ayamada
Added a table on keys and CSPs.

Revision 1.8.10.7  2007/09/19 19:26:44  ayamada
Added clarification of key establishment technique bit sizes.

Revision 1.8.10.6  2007/06/19 12:56:34  ayamada
Removed wrong statement regarding ECIES.
Reference update.

Revision 1.8.10.5  2007/05/18 18:22:51  ayamada
Correction on the list of KATs.

Revision 1.8.10.4  2007/05/07 12:10:45  ayamada
Added algorithm certificate numbers.

Revision 1.8.10.3  2007/05/02 12:39:53  ayamada
1. Updated the dates of references.
2. Added specific hardware as representative.
3. ECIES is no longer FIPS allowed.

Revision 1.8.10.2  2007/04/23 18:54:05  ayamada
Correction on the supported algorithms.

Revision 1.8.10.1  2007/04/20 17:48:44  sthambir
Synching with the trunk

Revision 1.10  2007/04/18 18:51:28  ayamada
Fixed the date.

Revision 1.9  2007/04/18 18:47:55  ayamada
Correction on KAT description.

Revision 1.8  2007/03/23 13:21:23  ayamada
Editorial errors are fixed.

Revision 1.7  2007/03/22 13:15:59  ayamada
A typo fix.

Revision 1.6  2007/03/21 19:35:20  ayamada
Further editorial corrections.

Revision 1.5  2007/03/21 19:33:31  ayamada
An editorial fix.

Revision 1.4  2007/03/21 17:02:30  ayamada
Further clarified the AES modes of operation.

Revision 1.3  2007/03/21 13:36:35  ayamada
1. Completed the algorithm list.
2. Editorial improvements.

Revision 1.2  2007/03/15 17:46:05  ayamada
```

```
Editorial corrections.

Revision 1.1  2007/03/14 19:54:14  ayamada
Initial revision.
Based on SB FIPS Module 2.0, and applied some fixes from ADS 1.2 and Palm.
Also, the algorithm sets are fixed.
```

2 Cryptographic Module Specification

SB FIPS Module is a multiple-chip standalone software cryptographic module that operates with the following components:

- A commercially available general-purpose computer hardware.
- A 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
5. Keyboard Interface
6. Mouse Interface
7. Network Interface
8. Serial Port
9. Parallel Port
10. Power Supply

The configuration of this component is illustrated in Figure 1.

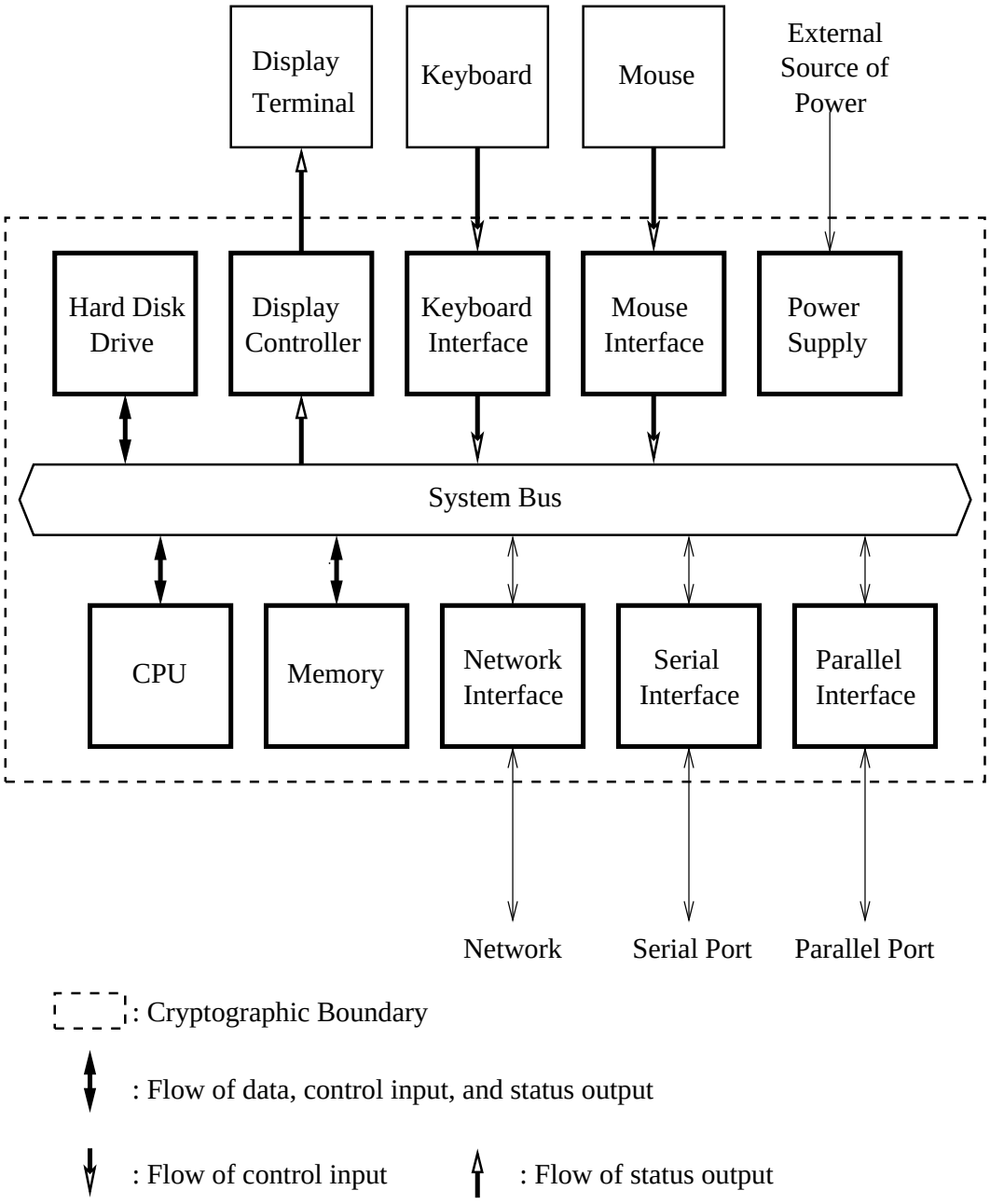


Figure 1: Cryptographic Module Hardware Block Diagram

2.2 Computer Hardware and OS

The combination of computer hardware and OS include the following representative platform:

1. Antenna Products' MMR running Yellow Dog Linux 2.6 on PowerPC
2. Nokia's multimedia computer running Maemo Linux 5 on ARMv7

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 Linux for PowerPC processors.

2.3 Software Specifications

SB FIPS Module is manufactured by Certicom Corp., providing services to the C computer language users in a shared object format. A single source code base is used for all identified computer hardware and OS.

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).

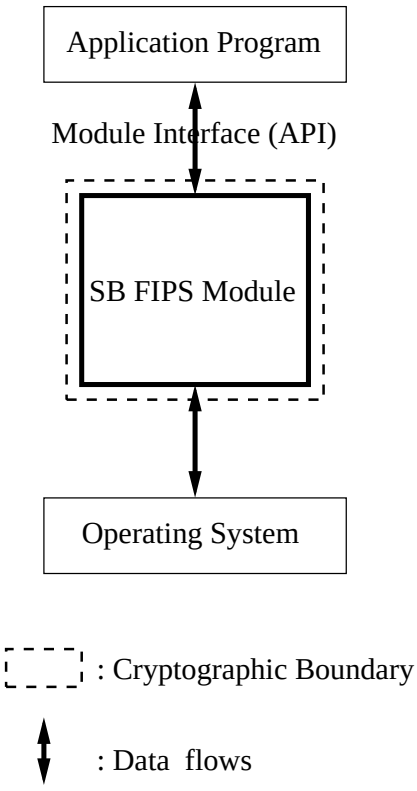


Figure 2: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The physical and logical interfaces are summarized in Table 1.

Table 1: 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. These roles are enforced by this Security Policy. The Crypto Officer has the responsibility for installing SB FIPS Module (see Table 2).

Table 2: Roles and Services

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

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 3.

4.2 Services

SB FIPS Module supports many cryptographic algorithms. The set of cryptographic algorithms supported by SB FIPS Module is given in Table 3.

Table 3: Supported Algorithms and Standards

	Algorithm	**FIPS Approved or allowed**	**Cert Number**
Block Ciphers	TDES (ECB, CBC, CFB64, OFB64) [FIPS 46-3]	×	#545
	AES (ECB, CBC, CFB128, OFB128, CTR, CCM, CMAC) [FIPS 197]	×	#549
	DES (ECB, CBC, CFB64, OFB64)		
	DESX (ECB, CBC, CFB64, OFB64)		
	AES (CCM*, GCM)		
	ARC2 (ECB, CBC, CFB64, OFB64) [RFC 2268]		
Stream Cipher	ARC4		
Hash Functions	SHA-1 [FIPS 180-2]	×	#614
	SHA-224 [FIPS 180-2]	×	#614
	SHA-256 [FIPS 180-2]	×	#614
	SHA-384 [FIPS 180-2]	×	#614
	SHA-512 [FIPS 180-2]	×	#614
	MD5 [RFC 1321]		
	MD4 [RFC 1320]		
	MD2 [RFC 1115]		
Message Authentication	HMAC-SHA-1 [FIPS 198]	×	#290
	HMAC-SHA-224 [FIPS 198]	×	#290
	HMAC-SHA-256 [FIPS 198]	×	#290
	HMAC-SHA-384 [FIPS 198]	×	#290
	HMAC-SHA-512 [FIPS 198]	×	#290
	HMAC-MD5 [RFC 2104]		
RNG	ANSI X9.62 RNG [ANSI X9.62]	×	#317
Digital Signature	DSS [FIPS 186-2]	×	#223
	ECDSA [FIPS 186-2, ANSI X9.62]	×	#57
	RSA PKCS1 v1.5 [PKCS #1 v2.1]	×	#246
	RSA PSS [PKCS #1 v2.1]	×	#246
	ECNR [IEEE 1363]		
	ECQV		
Key Agreement	DH [ANSI X9.42]	×	
	ECDH [ANSI X9.63]	×	
	ECMQV [ANSI X9.63]	×	
Key Wrapping	RSA PKCS1 v1.5 [PKCS #1 v2.1]	×	
	RSA OAEP [PKCS #1 v2.1]	×	
	ECIES [ANSI X9.63]		

The TDES, AES (ECB, CBC, CFB128, OFB128, CTR, CCM, and CMAC 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), RNG, DSA, RSA PKCS #1, and ECDSA algorithms have been validated to comply with FIPS. SB FIPS Module also supports FIPS allowed key establishment techniques (key agreement and key wrapping), DH, ECDH, ECMQV, and RSA PKCS #1. In order to operate the module in compliance with FIPS, only these FIPS Approved or allowed algorithms should be used.

DES, DESX, AES (CCM* and GCM modes), ARC2, ARC4, MD5, MD4, MD2, HMAC-MD5, ECNR, ECIES, and ECQV are supported as non FIPS Approved algorithms. In order to operate the module in compliance with FIPS, these algorithms should not be used.

Table 4 summarizes the keys and CSPs used in the FIPS mode.

Table 4: Key and CSP, Key Size, Security Strength, and Access

Algorithm	**Key and CSP**	Key Size	Strength	**Access**
AES	key	128-256 bits	128-256 bits	Create, Read, Use
TDES	key	112 bits	112 bits	Create, Read, Use
HMAC	key	160-512 bits	80-256 bits	Use
pRNG	seed key, seed	160 bits	80 bits	Use
DSA	key pair	1024-15360 bits	80-256 bits	Create, Read, Use
ECDSA	key pair	163-384 bits	80-192 bits	Create, Read, Use
RSA Signature	key pair	1024-15360 bits	80-256 bits	Create, Read, Use
DH	static/ephemeral key pair	1024-15360 bits	80-256 bits	Create, Read, Use
ECDH	static/ephemeral key pair	163-384 bits	80-192 bits	Create, Read, Use
ECMQV	static/ephemeral key pair	163-384 bits	80-192 bits	Create, Read, Use
RSA Key wrapping	key pair	1024-15360 bits	80-256 bits	Create, Read, Use

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 Installed/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 to be run in single user mode on the target operating system. For an operating system such as UNIX, Linux or Windows, it must be configured to run in single user mode.

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 3).

8.1 Key Generation

SB FIPS Module provides FIPS 140-2 compliant key generation. The underlying random number generation uses a FIPS Approved method, the ANSI X9.62 RNG [4].

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
5. RSA OAEP

The RSA key wrapping techniques above are based on the PKCS #1 v2.1 standard, and are used to transport keys.

The ECDH and ECMQV key agreement technique implementations support elliptic curve sizes from 163 bits to 384 bits that provides between 80 and 192 bits of security strength. 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 1024 bits and above must be used to provide minimum of 80 bits of security in the FIPS mode. The RSA implementation supports modulus sizes from 512 to 15360 bits that provides between 56 bits and 256 bits of security, where 1024 bits and above must be used to provide minimum of 80 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

SB FIPS Module does not import or export keys. It is responsibility of the application to import/export keys from the physical cryptographic boundary. 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 FIPS Module functions zeroize all intermediate security sensitive material. All CSPs are zeroized when they are no longer needed by calling destroy functions. Destruction of CSP is enforced in a manner such that missed destruction will make SB FIPS Module no longer 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, HMAC-SHS, RNG, and RSA Signature Algorithm. For DSA and ECDSA, Pair-wise Consistency Test is used.

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.

Also, 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.

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.

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 the Concurrent Versioning System (CVS) 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. The Crypto Officer must have administrative privileges on the computer.

Place the shared object, `libsbgse24.[so,dll]`, in an appropriate location on the computer hardware for your development environment.

A.1.2 Uninstalling

Remove the shared object, `libsbgse24.[so,dll]`, from the computer hardware.

A.2 Commands

A.2.1 Initialization

`sbg24_FIPS140Initialize()`

This function runs a series of self-tests on the module. These tests examine the integrity of the shared 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

`sbg24_FIPS140Deinitialize()`

This function de-initializes the module.

A.2.3 Self-Tests

`sbg24_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 shared 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

`sbg24_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 `sbg24_FIPS140Deinitialize()`, and then to initialize the module using `sbg24_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.