Security Builder® FIPS Module

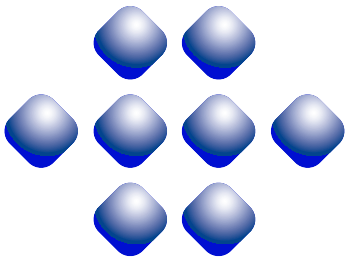
Version 3.0

for ARM ADS 1.2

FIPS 140-2 Non-Proprietary Security Policy

Certicom Corp.

February 16, 2007



certicom™

Contents

1 Introduction **5**
- 1.1 Overview . . . 5
- 1.2 Purpose . . . 5
- 1.3 References . . . 5
- 1.4 Change Notes . . . 7

2 Cryptographic Module Specification **8**
- 2.1 Physical Specifications . . . 8
- 2.2 Computer Hardware and OS . . . 10
- 2.3 Software Specifications . . . 10

3 Cryptographic Module Ports and Interfaces **12**

4 Roles, Services, and Authentication **13**
- 4.1 Roles . . . 13
- 4.2 Services . . . 13
- 4.3 Operator Authentication . . . 13

5 Finite State Model **16**

6 Physical Security **18**

7 Operational Environment **19**

8 Cryptographic Key Management **20**
- 8.1 Key Generation . . . 20
- 8.2 Key Establishment . . . 20
- 8.3 Key Entry and Output . . . 21
- 8.4 Key Storage . . . 21
- 8.5 Zeroization of Keys . . . 21

9 Self-Tests **22**
- 9.1 Power-up Tests . . . 22
  - 9.1.1 Tests upon Power-up . . . 22
  - 9.1.2 On-Demand Self-Tests . . . 22
- 9.2 Conditional Tests . . . 22
- 9.3 Failure of Self-Tests . . . 22

10 Design Assurance **23**
10.1 Configuration Management 23
10.2 Delivery and Operation 23
10.3 Development 23
10.4 Guidance Documents 24

11 Mitigation of Other Attacks **25**
11.1 Timing Attack on RSA 25
11.2 Attack on Biased Private Key of DSA 25

A Crypto Officer And User Guide **26**
A.1 Installation 26
A.1.1 Installing 26
A.1.2 Uninstalling 26
A.2 Commands 26
A.2.1 Initialization 26
A.2.2 De-initialization 26
A.2.3 Self-Tests 27
A.2.4 Show Status 27
A.3 When Module is Disabled 27

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 3.0** (SB FIPS Module) for ARM ADS 1.2. 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*, December 3, 2002.

[2] NIST *Security Requirements For Cryptographic Modules, Annex A: Approved Security Functions for FIPS PUB 140-2*, April 3, 2006.

[3] NIST *Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2*, November 4, 2004.

[4] NIST *Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2*, January 31, 2005.

[5] NIST *Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2*, September 12, 2005.

[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 5, 2006.

[8] NIST *Frequently Asked Questions for the Cryptographic Module Validation Program*, February 6, 2006.

1.4 Change Notes

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

```
$Log: ADS1_2ModuleSecurityPolicy.tex,v $
Revision 1.4.4.1  2006/08/30 12:46:25  ayamada
1. Added algorithm certificate numbers.
2. Typo fix.

Revision 1.4  2006/07/06 18:38:17  ayamada
Editorial correction.

Revision 1.3  2006/06/29 14:30:40  ayamada
More editorial improvements.

Revision 1.2  2006/06/29 12:22:43  ayamada
Editorial improvement on the title.

Revision 1.1  2006/06/28 12:59:43  ayamada
Initial revision.
```

2 Cryptographic Module Specification

SB FIPS Module is a multiple-chip standalone cryptographic module, consisting of the following components:

- A commercially available general-purpose handheld computing and communication device.
- A commercially available Operating System (OS) that runs on the device.
- SB FIPS Module software that runs on the device and the OS.

2.1 Physical Specifications

The device generally consists of the following physical components:

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. Data Storage (such as flush memory or hard disk)
4. Display
5. Keys and Buttons
6. Network Interface
7. Serial and/or Parallel Port
8. Battery

The configuration of the device is illustrated in Figure 1.

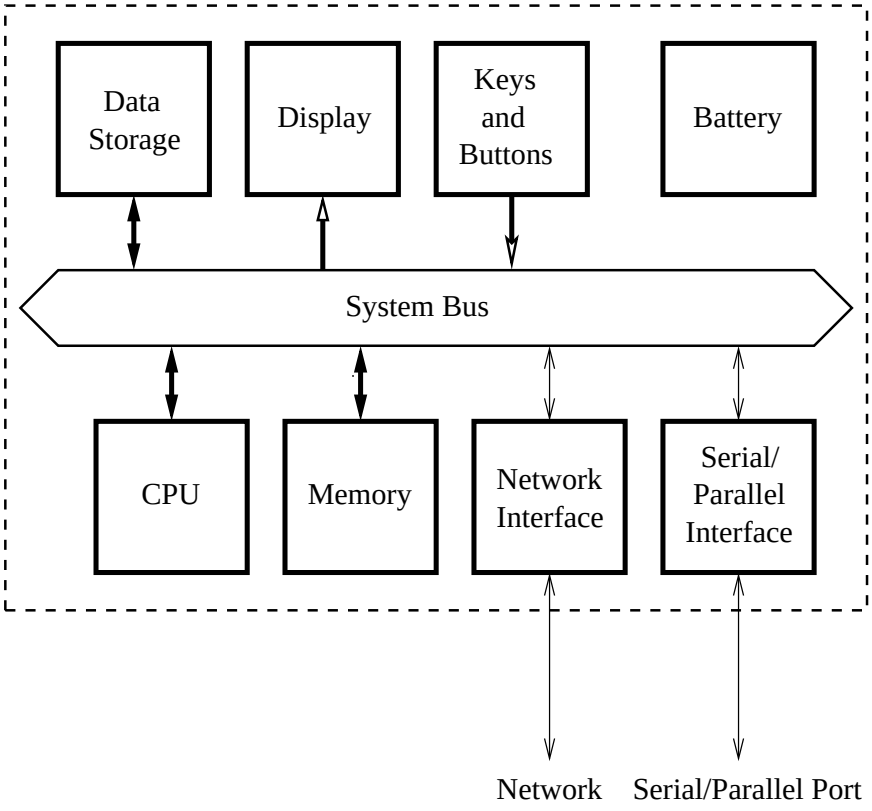


Figure 1: Cryptographic Module Hardware Block Diagram

2.2 Computer Hardware and OS

SB FIPS Module is designed for use with ARM ADS 1.2 development tools, and tested on a Samsung mobile phone with an ARM processor running the Phillips OS. SB FIPS Module is also suitable for use with development tools and devices that are compatible with the configuration as tested.

2.3 Software Specifications

The SB FIPS Module component is manufactured by Certicom Corp., providing services to the C computer language users as an object module. This object module, `sbgse3.fips.o`, must be linked with an application, which is loaded on smartphones.

The interface into the 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).

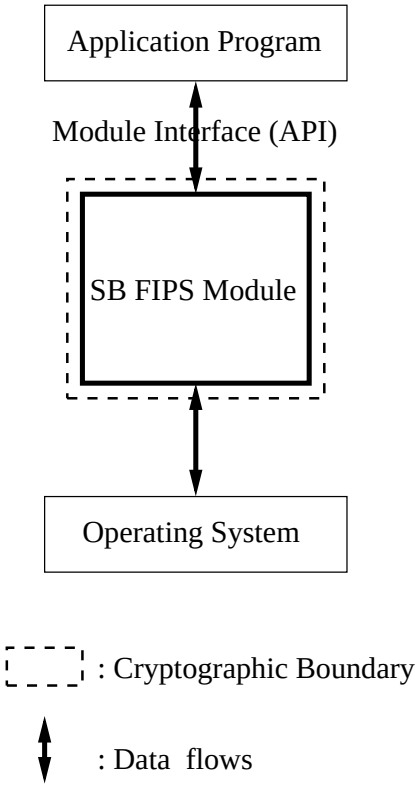


Figure 2: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The physical and logical interfaces are summarized in Table 1.

Table 1: Logical and Physical Interfaces

FIPS 140-2 Interface	**Logical Interface**	**Physical Interface**
Data Input	API	Wireless modem
Data Output	API	Wireless modem
Control Input	API	Keys and buttons
Status Output	Return Code	Display
Power Input	Initialization Function (`sbg3_FIPS140Initialize()`)	Not applicable (Battery is included)
Maintenance	Not supported	Not supported

4 Roles, Services, and Authentication

4.1 Roles

SB FIPS Module supports Crypto Officer and User Roles, meeting FIPS 140-2 Level 1 requirements. These roles are enforced by this Security Policy. The Crypto Officer has the responsibility for installing SB FIPS Module (see Table 2).

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 are given in Table 3.

The TDES, AES, 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 Approved or 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 algorithms should be used.

DES, DESX, ARC2, ARC4, MD5, MD2, and HMAC-MD5 are supported as non FIPS Approved algorithms. In order to operate the module in compliance with FIPS, these algorithms should not be used.

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.

Table 2: Roles, Services and Access

Service	Crypto Officer	User	Keys and CSPs	Access
Installation, etc.				
Installation	×			
Uninstallation	×			
Self-tests	×	×	ECDSA public key	Use
Show status	×	×		
Symmetric Ciphers				
Key generation	×	×	AES, TDES keys	Create, Read
Encrypt	×	×	AES, TDES keys	Use
Decrypt	×	×	AES, TDES keys	Use
Hash Algorithms and Message Authentication				
Hashing	×	×		
Msg. Authentication	×	×	HMAC keys	Use
Random Number Generation				
Seeding	×	×	Seed	Use
Request	×	×		
Digital Signature				
Key pair generation	×	×	RSA, DSA, ECDSA key pairs	Create, Read
Sign	×	×	RSA, DSA, ECDSA private keys	Use
Verify	×	×	RSA, DSA, ECDSA public keys	Use
Key Agreement				
Key pair generation	×	×	ECDH, ECMQV, DH key pairs	Create, Read
Shared secret generation	×	×	ECDH, ECMQV, DH key pairs	Use
Key Wrapping				
Key pair generation	×	×	RSA key pairs	Create, Read
Wrap	×	×	RSA public keys	Use
Unwrap	×	×	RSA private keys	Use

Table 3: Supported Algorithms and Standards

	Algorithm	**FIPS Approved or allowed**	**Cert Number** r
Block Ciphers	DES (ECB, CBC, CFB64, OFB64)		
	TDES (ECB, CBC, CFB64, OFB64) [FIPS 46-3]	×	#452
	DESX (ECB, CBC, CFB64, OFB64)		
	AES (ECB, CBC, CFB128, OFB128, CTR) [FIPS 197]	×	#421
	ARC2 (ECB, CBC, CFB64, OFB64) [RFC 2268]		
Stream Cipher	ARC4		
Hash Functions	SHA-1 [FIPS 180-2]	×	#491
	SHA-224 [FIPS 180-2]	×	#491
	SHA-256 [FIPS 180-2]	×	#491
	SHA-384 [FIPS 180-2]	×	#491
	SHA-512 [FIPS 180-2]	×	#491
	MD5 [RFC 1321]		
	MD2 [RFC 1115]		
Message Authentication	HMAC-SHA-1 [FIPS 198]	×	#195
	HMAC-SHA-224 [FIPS 198]	×	#195
	HMAC-SHA-256 [FIPS 198]	×	#195
	HMAC-SHA-384 [FIPS 198]	×	#195
	HMAC-SHA-512 [FIPS 198]	×	#195
	HMAC-MD5 [RFC 2104]		
RNG	ANSI X9.62 RNG [ANSI X9.62]	×	#217
Digital Signature	DSS [FIPS 186-2]	×	#176
	ECDSA [FIPS 186-2, ANSI X9.62]	×	#31
	RSA PKCS1-v1_5 [PKCS #1 v2.1]	×	#159
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]	×	

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. After the module is loaded on the memory along with the application, the module turns to to the Initialization state when the initialization command is applied to the module. 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

SB FIPS Module is designed for mobile phones, which are single user devices, thus always 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. EC Diffie-Hellman (ECDH)
2. ECMQV
3. Diffie-Hellman (DH)
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 size up to 384 bits that provides 192 bits of security. The DH key agreement technique implementation supports modulus size up to 15360 bits that provides 256 bits of security. The RSA implementation supports modulus size up to 15360 bits that provides 256 bits of security.

It is users responsibility 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, SHS, HMAC-SHS, RNG, and RSA PKCS #1 v1.5 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 RSA, DSA, or ECDSA 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.

This toolkit is designed and developed using high level language C, for C and C++ users. The low level language assembly is used to optimize lower level operations.

Development for the cryptographic module is carried out in a multi-platform environment. Certicom is using the CVS revision control system to control revisions. Development of new versions and major features are performed on a branch of the software, and these branches merged back into the trunk after testing and review, but the branch is maintained to perform post-release maintenance.

Releases are tested first in engineering (via an automated daily procedure, the daily build). They are then committed to the product candidate repository. These releases are then sent to the QA department which must run its own tests before they move them into the product branch. Only QA can move candidates into products.

Daily the software is built automatically on over 45 platforms (servers to embedded devices and PDAs) and the regression tests run. In addition to this, daily integration builds are performed to check the use of the cryptographic library as used in the higher level protocol products.

Weekly, code test coverage and code quality tools (purify and insure) are run on the product (these are very extensive tests taking longer than a day to complete).

Weekly benchmarks are also automatically performed on a representative subset of devices.

10.4 Guidance Documents

Crypto Officer Guide and User Guide are 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.

Unpack the distribution and place the file containing the object module in an appropriate location for application development. The object module will be incorporated into your application by the linker. Installing the application the application on target device will automatically install the SB FIPS Module.

A.1.2 Uninstalling

Removing your application from the target device will automatically remove the object module the SB FIPS Module.

A.2 Commands

A.2.1 Initialization

`sbg3_FIPS140Initialize()`

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

`sbg3_FIPS140Deinitialize()`

This function de-initializes the module.

A.2.3 Self-Tests

`sbg3_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 module, 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

`sbg3_FIPS140GetState()`

This function will return the current state of the module. For more details, refer to `sbg3fsm.h`.

A.3 When Module is Disabled

When SB FIPS Module becomes disabled, attempt to bring the module back to the Installed state by calling `sbg3_FIPS140Deinitialize()`, and then to initialize the module using `sbg3_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.