Security Builder®

Certicom Corp.
Security Builder® Government Solutions Edition

Palm

FIPS 140-2 Non-Proprietary Security Policy

June 25, 2003

Contents

1. Introduction

1.1 Purpose

This is a non-proprietary Federal Information Processing Standard (FIPS) 140-2 Security Policy for Certicom's **Security Builder® Government Solutions Edition Version 1.0** (C Module), or GSE™ (**SBGSE**). The **SBGSE** is based upon **Security Builder 3**. This Security Policy outlines **Security Builder's SBGSE** conformance to FIPS 140-2 Level 1 Security Requirements for Cryptographic Modules, and describes how to configure and operate the **Security Builder®** cryptographic module in order to comply with this standard.

1.2 References

For more information on the FIPS 140 standards and the validation program, please visit the NIST Web site at http://csrc.nist.gov/cryptval.

For more information on Certicom, **Security Builder® Government Solutions Edition**, and other Certicom products deploying the FIPS validated cryptographic module, please visit Certicom's web site at http://www.certicom.com.

1.3 Certicom's Security Builder® Products

Certicom is the industry leader in implementing public key technologies, providing the smallest and fastest solutions for strong security. The **Security Builder®** line of products provides a complete suite of cryptographic algorithms for developers to easily integrate encryption, digital signatures, and other security mechanisms into applications. It is unique in its ability to bring strong security to a full range of computing platforms, from servers and desktops to the new class of handheld and wireless information appliances. Powered by highly efficient Elliptic Curve Cryptography (ECC) and designed for the demanding requirements of constrained devices as well as offering RSA and many other ciphers, **Security Builder®** gives developers the security edge needed in today's wired and wireless world.

The uniform code-base **SB** C Product (C++ compatible) is available in optimized form for many popular embedded and server platforms, including:

AIX-32	Solaris (SPARC) 32-bit
AIX-64	Solaris (UltraSPARC II) 64-bit
ARMv4T ADS 1.2 Compiler	Symbian 6.1 (ARM4)
ARMv4T ADS 1.0.1 Compiler	Symbian 6.1 (ARMi)
ARMv4T SDT 2.5 Compiler	Symbian 6.1 (emu)
FreeBSD 3.2 (x86)	Tru64
HPUX-32	Win32 (x86)
HPUX-64	Win32 (x86) – Borland SDK 5.0
Linux (x86)	Win32 (x86) – Borland SDK 6.0
Linux (SA1110 iPAQ)	WinCE 3.0 PocketPC (SA1110)
Mac OS 9 (Motorola 680x0)	WinCE 3.0 PocketPC SH3
Mac OS 9 (PowerPC)	WinCE 3.0 PocketPC MIPS
Mac OS 10 (PowerPC)	WinCE 3.0 PocketPC (x86 emu)
MCORE 340 (Diab 4.3f)	WinCE 3.0 PocketPC 2002 (SA1110)
MCORE 330 (Diab 4.4a)	WinCE 3.0 PocketPC 2002 (x86 emu)

Netware x86 – Watcom	VxWorks 5.4 (386)
Netware x86 – Watcom	VxWorks 5.4 (386)
PalmOS (68K)	VxWorks 5.4 (486)
RimOS (386)	VxWorks 5.4 (Pentium)
Solaris (x86)	

The **Security Builder®** line of products is available in both Java and C versions to allow easy and rapid deployment of security services into any application. All **Security Builder®** products have a concise and intuitive API, combined with a consistent code base for desktop and wireless applications, dramatically reducing time-to-market and providing interoperable end-to-end security.

The **Security Builder®** line of products provides the cryptography for a variety of Certicom products, including movianCrypt™, movianVPN™, SSL Plus™, WTLS Plus™, and Trustpoint™ PKI products.

2. Security Builder® Government Solutions Edition 1.0 (C Module)

2.1 Cryptographic Module Specification

This Security Policy is for the **SBGSE** C Module product. The **SBGSE** C Module is a single product that is suitable for both embedded and desktop applications and implements DES, TDES, SHA-1, HMAC SHA-1, and AES FIPS approved algorithms and DESX, MD5, HMAC MD5 in non-FIPS mode.

The **SBGSE** C Module is designed to run on the Palm OS embedded platform running Palm OS 4 (or a compatible version of the operating system**;** in accordance with G.5). The **SBGSE** C Module is validated against FIPS 140-2 Level 1 on a Palm Vx handheld running Palm OS 4.1. The **SBGSE** C Module is supplied in the form of a Palm OS DLL. The **SBGSE** C Module is a multi-chip stand-alone module as classified by FIPS 140-2.

The physical cryptographic boundary for the Palm OS platform is the case of a Palm OS compatible PDA running Palm OS 4 (or a compatible version of the operating system**;** in accordance with G.5), and the **SBGSE** C Module. The logical boundary of the module contains the Palm OS DLL.

2.2 Cryptographic Module Ports and Interfaces

2.2.1 Processor Interfaces

Palm OS 4 (and compatible versions of the operating system; in accordance with G.5)

The processor performs all major services (see Table 1). The software for these services resides on the internal flash/RAM memory (on the main board). This same memory contains the data.

The processor interfaces with:

1) the user keys and point device
2) the serial port

2.2.2 Module Interfaces

The **SBGSE** C Module's physical ports and logical interfaces, as a multi-chip stand-alone module, consists of the following devices:

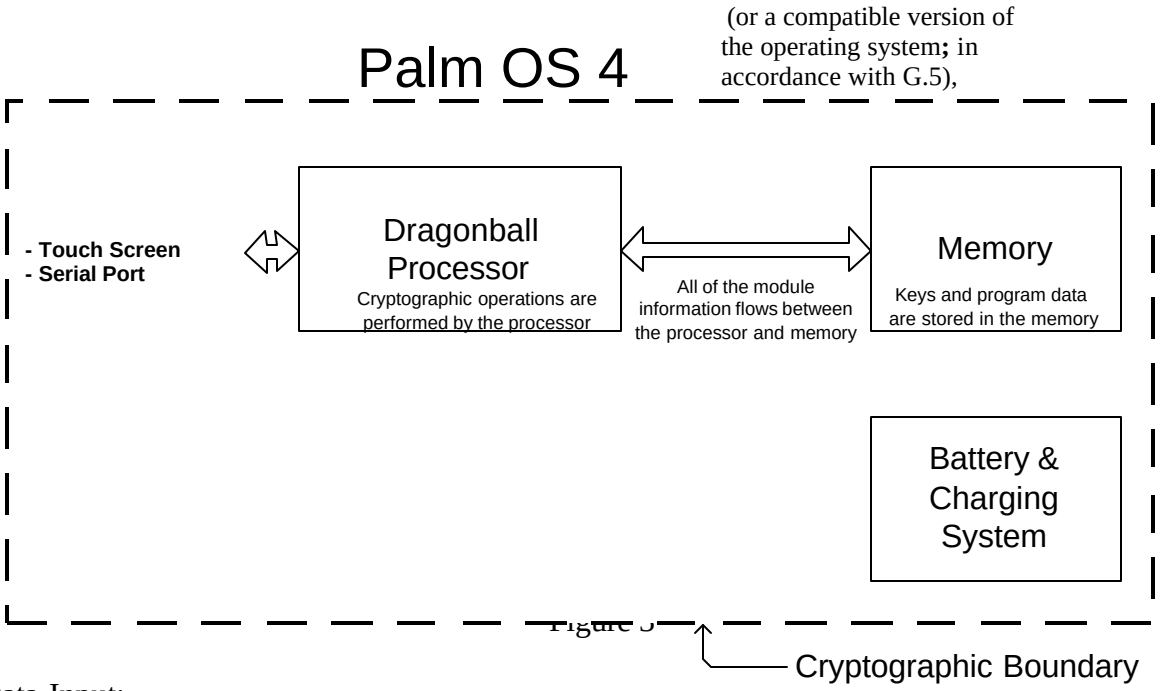

Figure 3

Data Input:

The Serial Port is used to load the **SBGSE** C Module onto the Palm device.

The Data Input Interface is the parameters of the API function calls defined as input.

Data Output:

The Serial Port is used for Data Output from the Palm device.

The Data Output Interface is the parameters of the API function calls defined as output.

Power Input:

The Battery and Charging System is the power port.

Control Input and Status Output:

The Touch Screen is used as a Control Input port in the installation of the **SBGSE** C Module. The Touch Screen is also used to display the status of the **SBGSE** C Module.

The Control Interface is the function calls of the SBGSE API.

The Status Output Interface is the return values of the function calls of the SBGSE API.

To install the **SBGSE** C Module on the Palm device, the Hotsync facility is utilized (see the Crypto Officer Manual). Once the installation files are on the device, it is to be disconnected from the host PC and the remainder of the installation completed.

The interface into the **SBGSE** C Module is via API function calls. These function calls provide the interface to the cryptographic services for which the parameters and return codes, provide the

input/output data and status conditions. The API function calls can be used to provide **SBGSE** customers with FIPS 140-2 Level 1 validated cryptographic services.

2.3 Roles and Services

SBGSE C Module provides Crypto Officer and User Roles, meeting FIPS 140-2 Level 1 requirements. These roles are enforced by the module itself and not by an external policy.

2.3.1 The Crypto Officer Role

The Crypto Officer has the responsibility for installing the **SBGSE** C Module.

The Crypto Officer is responsible for handling key generation and RNG seeding (see Tables 2 and 3).

The Crypto Officer's role calls Crypto Officer API functions putting the module into Crypto Officer State to perform these operations.

2.3.2 The User Role

The User of the cryptographic module may call all **SBGSE** C Module API functions which provide cryptographic functions (see Tables 2 and 3). In order to operate the module securely, it is the User's responsibility to confine calls to those methods that have been FIPS 140-2 validated (see Table 2). The modules enter the Crypto User State to operate the Crypto User functions.

2.3.3 Approved Mode of Operation

In the approved mode of operation, all Roles shall confine themselves to calling FIPS Approved algorithms, as marked in Table 1.

2.3.4 Finite State Model

A finite state model is provided as a separate document for FIPS validation.

In summary, the Finite State model contains the following states:

Initial
Installed
Self-Test
Module Active
Crypto Officer
Crypto User
Soft Error
Hard Error
Disabled

The Initial State does not contain the crypto module. The Officer installs the module, transiting to Installed. The Crypto Officer must call the sbg_Initialize() function to power up the module. On power up, the module enters Self-Test (automatically). On success the module enters Module Active; on failure the module enters Disabled.

From the Module Active State (which is only entered if self-tests have succeeded), the module can transit to the Crypto User State when a user level API function is called and to the Crypto Officer State when a Crypto Officer level API function is called (See Table 4). Currently (for all Level 1 certification) no authentication of roles is implemented.

When the Crypto function has completed successfully, the state transits back to Module Active.

If the RNG continuous test fails, the State transits to Hard Error and then to Disabled.

If there is an error with the user parameters, an error return code is returned (the Soft Error state) and the module then transits back to Module Active without completing any operation.

The User has the opportunity of correcting his input, based upon the error return code (Soft Error) and may call the function again.

2.3.5 Physical Security
The multi-chip stand-alone module embodies a commercial production quality computing system with commercial production quality cases.

2.3.6 Operational Environment
The module is restricted to single operator mode operation by the operating system.

2.4 Cryptographic Key Management
The **SBGSE** C 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's responsibility to select FIPS 140-2 validated algorithms (see Table 1). In order to securely operate this cryptographic module, a Crypto Officer must ensure that keying material is handled in accordance with FIPS 140-2. They must also ensure that the memory used to store these keys is properly zeroized after use.

2.4.1 Key Generation
The **SBGSE** C Module provides FIPS 140-2 compliant key generation. The underlying random number generation uses a FIPS Approved method, the X9.62 RNG.

2.4.2 Key Storage
The **SBGSE** C Module is a low-level cryptographic toolkit, and as such does not provide long-term key storage. The module also does not have any provision for input/output of keys or seeds outside the crypto boundary. The module does provide FIPS Approved algorithms which can be used to encrypt/decrypt keys for storage or input/output in the application employing the cryptographic module. It is the Crypto Officer's responsibility to store, zeroize, and otherwise handle keying information in accordance with FIPS 140-2. The module contains, in the code space, a hard-coded HMAC-SHA-1 key value, which is used to calculate the HMAC-SHA-1 tag used for the integrity testing. This hard-coded key has been produced with the ANSI X9.62 RNG in FIPS mode. The HMAC-SHA-1 key can be destroyed by performing a hard reset of the Palm device.

2.4.3 Zeroization of Keys

The **SBGSE** C Module functions zeroize all intermediate key-derived material. Actual key values are contained in memory inside the crypto boundary. The test functions in the **SBGSE** C Module zeroize all key material after user (employing the native key Destroy functions), as should the Crypto Officer employing the **SBGSE** C Module.

2.4.4 Protection of Keys

The test functions provided with the **SBGSE** C Module zeroize all keying material after use. This, combined with the zeroization of intermediate key-derived material inside the module, protects the keys when the OS is in single user mode (as is required by this validation). In order to comply with FIPS 140-2 guidelines, the user of this module must also take care to safeguard access to the keying information and zeroize it after use.

2.4.5 Cryptographic Algorithms

The **SBGSE** C Module supports many cryptographic algorithms. The RNG, SHA-1, DES (DES is provided for legacy use), TDES, AES, and HMAC SHA-1 algorithms are FIPS Approved. DESX, MD5, and HMAC MD5 are supported as non FIPS Approved algorithms. The set of cryptographic algorithms supported by the **SBGSE** C Module are given in Table 1.

2.4.6 Secure Operation

In order to operate the module in compliance with FIPS, only the FIPS approved and tested algorithms or FIPS allowed methods should be used. These algorithms are RNG, SHA-1, DES (DES is provided for legacy use), TDES, and AES (see Table 1). These critical functions are all subjected to power-up tests specified in the next section.

2.5 Self Tests

2.5.1 Startup Self Tests

Self-tests are initiated automatically by the module at start-up. Start-up tests include software integrity tests and known answer tests for the tested algorithms. Self-tests (KATs) are performed on RNG, SHA-1, DES (DES is provided for legacy use), TDES, AES, and HMAC SHA-1. Failure of the self-test places the cryptographic module in the Hard Error State, wherein no cryptographic operations can be performed. (The module is disabled.) If any self-test fails, the cryptographic module will output an error output and transitions to a disabled state.

2.5.2 On-Demand Self Tests

On-demand Self Tests may be invoked by running a function available to the Cryptographic Officer, as described in the Crypto Officer & User Manual.

2.5.3 Continuous Self Tests

The RNG performs continuous self-tests of 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.

3. Design Assurance

A configuration management system for the cryptographic module is employed and has been described in a document to the certifying lab. It uses the concurrent versioning system (CVS) to track the configurations.

3.1 Delivery and Operation

Please refer to the Crypto Officers Manual, Section 2.2.1 to review the steps necessary for the secure installation, initialization and start-up of the cryptographic module.

3.2 Development

Detailed design information and procedures have been described in documentation submitted to the certifying lab.

Development for the crypto module is carried out in a multi-platform environment. We are 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 30 platforms (servers to embedded devices and PDA's) 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.

3.3 Guidance Documents

The Crypto Officers Manual, User Guide and the Programmer's Reference Manual (both made available to the Crypto Officer) outline the functions, security events and security parameters.

3.4 Mitigation of Other Attacks

This Crypto Module is not designed for the mitigation of any other attacks.

Table 1

Type	Algorithm	FIPS Approved Algorithms
Hash Functions:	SHA-1 [FIPS 180-1, http://www.itl.nist.gov/fipspubs/fip180-1.htm]	X
	MD5 [RFC: 1321, http://www.ietf.org/rfc/rfc1321.txt]	
Message Authentication:	HMAC-SHA-1(Message Authentication Codes) [FIPS 198, IEEE P1363, http://grouper.ieee.org/groups/1363/]	X
	HMAC-MD5 [IEEE P1363, http://grouper.ieee.org/groups/1363/]	
RNG (Random Number Generation):	ANSI X9.62 Random Number Generation [ANSI X9.62]	X
Block Ciphers:	DES [FIPS 46-2, http://www.itl.nist.gov/fipspubs/fip46-2.htm]	X
	(for legacy systems)	
	TDES [FIPS 46-3, http://www.itl.nist.gov/fipspubs/fip46-3.htm]	X
	DESX	
	AES [FIPS 197, http://csrc.nist.gov/encryption/aes/]	X

Table 2
Roles and Functionality

Services Available to:	Crypto Officer	User
ANSI X9.62		
Seeding	X	
Use	X	X
DES (for legacy)/TDES		
Key generation	X	
encrypt/decrypt	X	X
AES		
Key generation	X	
encrypt/decrypt	X	X
SHA-1 Hashing	X	X
MD5 Hashing	X	X
HMAC SHA-1	X	X
HMAC MD5	X	X

Table 3
Security Relevant Data Item Access Matrix

Operator*	Role	Service	Security relevant data item	Access
An Operator	Crypto Officer	`sbg_DESKeyCreate`	DES Key	Create
An Operator	Crypto Officer	`sbg_DESKeyGet`	DES Key	Read
An Operator	Crypto Officer	`sbg_DESEncrypt`	DES Key	Use
An Operator	Crypto Officer	`sbg_DESDecrypt`	DES Key	Use
An Operator	Crypto Officer	`sbg_DESKeyDestroy`	DES Key	Delete
An Operator	User	`sbg_DESEncrypt`	DES Key	Use
An Operator	User	`sbg_DESDecrypt`	DES Key	Use
An Operator	User	`sbg_DESKeyDestroy`	DES Key	Delete
An Operator	Crypto Officer	`sbg_DESKeyCreate`	TDES Key	Create
An Operator	Crypto Officer	`sbg_DESKeyGet`	TDES Key	Read
An Operator	Crypto Officer	`sbg_DESEncrypt`	TDES Key	Use
An Operator	Crypto Officer	`sbg_DESDecrypt`	TDES Key	Use
An Operator	Crypto Officer	`sbg_DESKeyDestroy`	TDES Key	Delete
An Operator	User	`sbg_DESEncrypt`	TDES Key	Use
An Operator	User	`sbg_DESDecrypt`	TDES Key	Use
An Operator	User	`sbg_DESKeyDestroy`	TDES Key	Delete
An Operator	Crypto Officer	`sbg_AESEncryptKeyCreate`	AES Key	Create
An Operator	Crypto Officer	`sbg_AESDecryptKeyCreate`	AES Key	Create
An Operator	Crypto Officer	`sbg_AESKeyCreate`	AES Key	Create
An Operator	Crypto Officer	`sbg_AESKeyGet`	AES Key	Read
An Operator	Crypto Officer	`sbg_AESEncrypt`	AES Key	Use
An Operator	Crypto Officer	`sbg_AESDecrypt`	AES Key	Use
An Operator	Crypto Officer	`sbg_AESKeyDestroy`	AES Key	Delete
An Operator	User	`sbg_AESEncrypt`	AES Key	Use
An Operator	User	`sbg_AESDecrypt`	AES Key	Use
An Operator	User	`sbg_AESKeyDestroy`	AES Key	Delete
An Operator	Crypto Officer	`sbg_HMACSHA1Begin`	HMAC-SHA-1 Key	Use
An Operator	User	`sbg_HMACSHA1Begin`	HMAC-SHA-1 Key	Use

*The module uses implicit role based authentication

Table 4
Roles and API Calls

Services Available to:	Crypto Officer	User
ANSI X9.62 RNG		
sbg_ANSIRngCreate	X	
sbg_ANSIRngDestroy	X	X
sbg_FIPS140ANSIRngCreate	X	
sbg_FIPS140ANSIRngDestroy	X	X
sbg_RngReseed	X	
sbg_RngGetBytes	X	X
DES/TDES Symmetric Encryption and Decryption		
sbg_DESParamsCreate	X	
sbg_DESParamsGet	X	
sbg_DESParamsDestroy	X	X
sbg_DESKeyCreate	X	
sbg_DESKeyGet	X	
sbg_DESKeyDestroy	X	X
sbg_DESBegin	X	X
sbg_DESEncrypt	X	X
sbg_DESDecrypt	X	X
sbg_DESEnd	X	X
AES Symmetric Encryption and Decryption		
sbg_AESParamsCreate	X	
sbg_AESParamsGet	X	
sbg_AESParamsDestroy	X	X
sbg_AESEncryptKeyCreate	X	
sbg_AESDecryptKeyCreate	X	
sbg_AESKeyCreate	X	
sbg_AESKeyGet	X	
sbg_AESKeyDestroy	X	X
sbg_AESBegin	X	X
sbg_AESEncryptBegin	X	X
sbg_AESDecryptBegin	X	X
sbg_AESEncrypt	X	X
sbg_AESDecrypt	X	X
sbg_AESEnd	X	X
SHA-1 Hashing		
sbg_SHA1Begin	X	X
sbg_SHA1Hash	X	X
sbg_SHA1End	X	X
sbg_SHA1DigestGet	X	X
sbg_SHA1CtxReset	X	X
sbg_SHA1CtxDuplicate	X	X

MD5 Hashing		
sbg_MD5Begin	X	X
sbg_MD5Hash	X	X
sbg_MD5End	X	X
sbg_MD5CtxDuplicate	X	X
HMAC		
sbg_HMACSHA1Begin	X	X
sbg_HMACSHA1Hash	X	X
sbg_HMACSHA1End	X	X
sbg_HMACMD5Begin	X	X
sbg_HMACMD5Hash	X	X
sbg_HMACMD5End	X	X