Attachmate Crypto Module

FIPS 140-2 Level 1 Non-Proprietary
Security Policy
Version 1.19

Revision Table

Revision #	Date	Author	Description
0.9	Oct 31, 2005	Attachmate	Initial work.
1.0	Nov 03, 2005	Attachmate	Initial Submission
1.1	Jan 03, 2006	Attachmate	Code and doc changes per Atlan
1.2	Jan 04, 2006	Attachmate	Updated supported platforms.
1.3	Jan 05, 2006	Attachmate	Doc changes per Atlan
1.4	Jan 23, 2006	Attachmate	Updated section 2.1 per Atlan
1.5	Apr 19, 2006	Attachmate	Ia32 requirement doc change
1.6	Jul 07, 2006	Attachmate	Logo and company name changes
1.7	Jul 07, 2006	Attachmate	Copyright changes
1.8	Jul 07, 2006	Attachmate	Doc changes per Atlan
1.9	Jul 12, 2006	Attachmate	Doc changes per Atlan
1.10	Jul 12, 2006	Attachmate	Doc changes Section 7.4 – added API's that perform key zeroization.
1.11	Jul 13, 2006	Attachmate	Doc changes per Altan.
1.12	Jul 14, 2006	Attachmate	Doc change per Atlan
1.13	Jul 17, 2006	Attachmate	CSP doc changes per Atlan
1.14	Jul 20, 2006	Attachmate	Section 2.4 changes per Atlan
1.15	Jul 21, 2006	Attachmate	Doc changes per Atlan
1.16	Oct 05, 2006	Attachmate	Fix for US-CERT Vulnerability #845620
1.17	Feb 22, 2007	Attachmate	Doc changes per Atlan
1.18	Apr 2, 2007	Attachmate	Doc changes per Atlan
1.19	Apr 30, 2007	Attachmate	Doc changes per Atlan

Copyright

Table of Contents

1. Introduction

This document specifies the non-proprietary Security Policy for the Attachmate Crypto Module cryptographic module version 1.0.170; hereafter known as the cryptographic module or simply the module. This document describes how the cryptographic module complies with FIPS 140-2 Level 1 requirements and how to run this module in FIPS 140-2 mode. This security policy has been created in accordance with the Federal Information Processing Standard (FIPS) 140-2 Level 1.
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

1.1 Documents

The Security Policy document is one of several documents that comprise the complete FIP 140-2 submission package.

- Security Policy (this document)
- Other supporting documentation as referenced within this document

2. Cryptographic Module Specification

2.1 Module Overview

The physical configuration of the module as defined in FIPS PUB 140-2 is Multi-Chip Standalone. The primary purpose for this software module is to provide secure communication over TCP/IP networks between a host computer and a personal computer (PC). The cryptographic module is a software module that runs on multiple platforms. The module is available as a dynamic link library, RSSCCM.DLL for Microsoft Windows and a shared library, libssccm.so for HP-UX, Linux, and Solaris operating systems running in single user mode.

Attachmate products utilize this module through the interfaces specified in the Functional Specification Document. The FIPS 140-2 Level 1 validated platforms are listed in Section 2.5. The module is written in C and assembler code.

For *Microsoft Operating Systems* the module runs as a dynamically linked library (DLL).

For *Unix Operating Systems* the module runs as a shared library for HP-UX, Linux, and Solaris.

See section 2.5 for a complete list of validated platforms and platforms tested for binary compatibility.

This security policy and the FIPS 140-2 validation applies to all Attachmate products that use this version of the cryptographic module. No components of the cryptographic module were excluded from the validation process.

2.2 Module Security Level Specification

The module is validated to the following FIPS 140-2 levels:

Section	Level
Overall requirements	1
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles and Services	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	3
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	1

2.3 Boundary

The *physical boundary* for the module is defined by the hardware enclosure which completely surrounds the entire cryptographic module, hardware and operating system. The *logical boundary* contains the software module that comprises the cryptographic module.

2.4 Modes of operation

The module supports both an Approved and a Non-Approved mode of operation. The module can be set to run in Approved mode on power-up. The Approved mode of operation is implicitly assumed based on the functions being executed. If the user executes a Non-Approved function, such as a MD2 hashing function, the module will not execute said function in FIPS mode.

2.5 Validated platforms

The module was tested and validated on the following platforms:

Intel Pentium 4
Window 2003 Server SP1 (x86), Red Hat Enterprise Linux 4.0 (x86), SuSE Linux Enterprise Server 9.0 (x86), and Sun Solaris 10 (x86),

Intel Pentium D
Windows 2003 Server SP1 (x64) and Red Hat Enterprise Linux 4.0 (x64)

Intel Itanium
Windows 2003 Server SP1 (IA64), Red Hat Enterprise Linux 4.0 (IA64), and HP-UX 11iv2 (IA64)

AMD Opteron
SuSE Linux Enterprise Server 9.0 (x64) and Sun Solaris 10 (x64)

Sun UltraSPARC

Solaris 8

PA-RISC
HP-UX 11iv1

In addition, the module was tested by Attachmate on the following platforms:

- Windows XP (x86)
- Windows 2000 Server (x86)
- Windows Longhorn Server (x86, x64)
- SuSE Linux Enterprise 8 (x86)
- SuSE Linux Enterprise 9(IA64, IBM LPAR)
- RHEL 3.0 (x86, IA64, x64)
- RHEL 4.0 (IBM LPAR)
- Sun Solaris 7 (SPARC)
- Sun Solaris 9 (SPARC, x86)
- Sun Solaris 10 (SPARC)
- IBM AIX 5.2 (PowerPC)
- IBM AIX 5.3 (PowerPC)
- IBM Z/OS (Unix system services)

3. Cryptographic Module Ports and Interfaces

The cryptographic module is a software module; as such the interfaces that it provides are defined in terms of the API. *Data Input Interface* is defined as those API calls that accept as their arguments data to be used or processed by the module. The *Data Output Interface* is comprised of those API calls that return by means of a return value, or arguments of the appropriate types, data generated or otherwise processed by the module. The *Control Input Interface* is comprised of the call used to initiate the module and the API calls used to control the operation of the module. *Status Output Interface* defines the API calls that provide information about the status of the module through the use of return variables.

The operating system and application layer software map these ports to the logical interfaces described in the Functional Specification and API documents.

4. Roles and Services

Roles

This module supports *User* and *Crypto Officer* roles as defined in the FIPS 140-2 standard. A *Maintenance* role is not implemented. Since the module is validated at Level 1 it does not provide authentication for any role.

Roles are defined as follows:
A **User** is any entity that can access services provided by the module.
A **Crypto Officer** is any entity that can install the module onto the computer system, configure the device to ensure correct operation of the module, or access service provided by the module. The Crypto Officer may access all services the same as a User.

Approved Services
In Approved mode, the module supports the following services:

Service	**Algorithm**	**Standard**	**Mode**
Symmetric Encryption/Decryption	AES (128, 192, 256 bit key)	FIPS 197	CBC,CFB,OFB
Symmetric Encryption/Decryption	Triple DES (168 bit, 3 key)	FIPS 46-3	CBC,CFB, OFB
Message Digest	SHA-1	FIPS 180-2	
Message Digest	HMAC-SHA-1	FIPS 198	
Message Digest	SHA-256	FIPS 180-2	
Message Digest	HMAC-SHA-256	FIPS 198	
Message Digest	SHA-512	FIPS 180-2	
Message Digest	HMAC-SHA-512	FIPS 198	
Message Digest	CBC-MAC / Triple DES	FIPS 180-2	
Digital Signing/Verification	RSA Digital Signature (minimum 1024 bits)	PKCS #1 (RSA)	
Digital Signing/Verification	DSA Digital Signature (minimum 1024 bits)	FIPS 186-2 (DSA)	
Random Number Generator	ANSI X9.31 with 2-key Triple DES		
Self Test			
Show Status			
Zeroize			
File Deletion			

Non-Approved services

In non-FIPS mode, the module supports the following services:

Service	**Algorithm**
Encryption	Blowfish
Encryption	CAST
Encryption	DES (56-bit key)
Encryption	Arcfour
Message Digest	Ripemd160
Message Digest	MD5
Message Digest	MD4
Message Digest	MD2
Message Digest	RC5
Message Digest	RC2
Message Digest	HMAC-MD5
Message Digest	HMAC-MD4
Message Digest	HMAC-MD2
Message Digest	HMAC-RIPEMD-160
Message Digest	CBC-MAC / DES
Asymmetric Encryption/Decryption	RSA Encryption/Decryption
Digital Signature generation and verification	RSA (minimum 512 bits)
Key Establishment (allowed in FIPS mode)	- Diffie-Hellman - The key establishment methodology provides 56, 80, 112, or 152 bits of encryption strength. - RSA Key wrapping (minimum 1024 bits) - key establishment methodology provides between 80-bits and 152 bits of encryption strength.
Passphrase based key derivation	

Algorithm and Certificate Numbers

Algorithm	Certificate Number
Triple-DES	447
AES	417
SHA	486
DSA	173
RSA	208
RNG	212
HMAC	191

Critical Security Parameters (CSPs)

The following are CSPs contained in the module:

- Secret Key
- Private Key
- Public Key

The following table identifies CSP's and the available access for the supported services.

Service	**CSP**	**Access**
Encryption/Decryption (AES 128, 192, 256, Triple DES)	Secret key	RW
Message Digest (HMAC)	Secret key and checksum	RW
Key Establishment (Diffie-Hellman) (512, 1024, 2048, and 4096 bits)	Public and Private key	RW
DSA Signing (1024 bit modulus size)	DSA Private key	RW
RSA Signing (1024, 1536, 2048, 3072, and 4096 bits modulus sizes)	RSA Private key	RW
DSA Verification (1024 bit modulus size)	DSA Public key	RW
RSA Verification (1024, 1536, 2048, 3072, and 4096 bits modulus sizes)	RSA Public key	RW
ANSI X9.31 Random Number Generator (2 key Triple-DES)	RNG Seed key (2 key Triple-DES)	RW
ANSI X9.31 Random Number Generator	RNG Seed	W

The module does not store keys in any persistent storage media; all keys are stored in RAM, except the HMAC-SHA-1 key which is used for the data integrity test.

5. Finite State Model

The Finite State Model is located in the Finite State Model document.

6. Physical Security

The cryptographic module is a software module tested for use on standard hardware running Windows, Solaris, HP-UX, and Linux operating systems running in single user mode. See Section 2.5 for more detailed platform information. Each platform provides production grade hardware and enclosure.

For each validated platform the hardware meets the applicable Federal Communication Commission (FCC) Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for Class B home or business use as defined in Subpart B of FCC Part 15.

7. Key Management

The module implements a number of functions that are either used internally or exposed in the API to meet the FIPS 140-2 level 1 requirements for key management. All keys generated by the module are generated using the approved RNG.

7.1 Key Generation

Symmetric keys for encryption/decryption and HMAC algorithms can be generated calling the Random Number Generator Service to produce a desired number of bytes.

Asymmetric key pairs (DSA, RSA and Diffie-Hellman) can also be generated for encryption/decryption, digital signing and verification, and key establishment.

Intermediate key generation values are never output from the Module.

7.2 Key Entry, Output and Storage

All keys generated or processed by the module reside in the applications process space. This means that the application has full access to the keys.

It is the responsibility of the authorized user to use the API's in a manner that will ensure FIPS 140-2 compliance.

7.3 Key Agreement

The module provides Diffie-Hellman API's that allow the user to establish a shared secret. The shared secret is output from the module and is then in full control of the user.

7.4 Key Zeroization

Keys that are stored in memory are overwritten with random data before the memory is released. Keys that are store on disk are destroyed by overwriting the data multiple times to ensure it is not recoverable.

It is the responsibility of the user to ensure the proper API's are used to accomplish this.

The API's that perform Key Zeroization are:

`fipsZeroize`, EVP_PKEY_free, RSA_free, DSA_free, DH_free, OPENSSL_cleanse

All keys except the RNG seed can be zeroized by the above mentioned APIs. The RNG seed can be zeroized by reloading the module.

7.5 Random Number Generator

The module implements the ANSI X9.31 with 2-key Triple DES random number generator.

The module does not output intermediate key generation values. The RNG is seeded on power-up. There is no periodic seeding.

The module also implements a non-Approved RNG to initially seed the Approved RNG. The RNG uses OS calls to gather random data.

Continuous PRNG testing is performed on both RNGs, each time the PRNGs produce random bytes. Moreover, the Approved RNG also tests the seed and seed key each time to ensure that they are not equal. A failure of either test will report a critical error, which will cause the module to be zeroized.

8. Self Tests

The cryptographic module runs Power-Up and Conditional Self-Tests to verify compliant operation.

Power-Up Self-Tests automatically execute during module initialization, and do not require any input or output by the user. If any of the tests fail the module enters an error state and prevents any cryptographic operation from being performed. See the Finite State Model document for state transition details. Output of keys and plaintext data is inhibited during module Self-Tests.

Conditional Self-Tests are executed when specific conditions are met. In addition, the module Self-Test service is available via an API.

Prior to each use, the internal RNG shall be tested using the conditional test specified in FIPS 140-2 §4.9.2.

Power up Self-Tests

The integrity test, as well as the known answer tests, are run when the module is put into approved mode. This is done by calling fipsInitialize or ssccm_init. Both of these functions return an integer indicating success (non-zero) or failure (zero). In case of a failure then more information can be found by calling fipsStatus which returns a more specific error code. For example, if the integrity check fails then fipsStatus will return FIPS_ERR_DIGEST_DOES_NOT_MATCH. More details are included in the Functional Specification and API documentation.

Software Integrity Test

- HMAC SHA-1 hash verification (outlined in the Integrity Check document).

Cryptographic Algorithm Known Answer Tests (KATs)

- Triple-DES KAT
- AES KAT (128, 192, 256 bit)
- SHA-1 KAT
- SHA-256 KAT
- SHA-512 KAT
- HMAC-SHA-1 KAT
- HMAC-SHA-256 KAT
- HMAC-SHA-512 KAT
- PRNG KAT
- DSA sign/verify test
- RSA sign/verify test

Conditional Self-Tests

- Continuous Random Number Generator tests for Approved and non-Approved RNGs
- Seed and Seed Key test (PRNG)

Pairwise Consistency Tests

The module performs a pair-wise consistency test, as required by FIPS 140.2, section 4.9.2, each time an asymmetric key pair is generated. If this test fails the module enters a critical error state which will not allow the keys to be output and will zeroize the key.

- RSA pairwise consistency test
- DSA pairwise consistency test

On Demand Self-Testing

At any time the cryptographic module is in an idle state, the operator can command the module to perform the Power-Up Self-Test by calling the Self-Test service.

9. Design Assurance

Attachmate manages and records source code and associated documentation files using the Perforce source control system. http://www.perforce.com/

For both Windows and Unix, all source code files, vendor documentation, and results binaries are under source control with the Perforce source control system. Perforce uses what is called a Changelist to uniquely track revision history. In addition, the compiled cryptographic module is versioned per convention on the target operating system. The final version for the cryptographic module is: 1.0.170

10. Mitigation of Other Attacks Policy

During RSA decryption and signing this module uses blinding techniques to mitigate timing analysis attacks.