FIPS 140-2 Non-Proprietary Security Policy

Websense Java Crypto Module

Software Version 2.0.1

Document Version 0.9

August 10, 2015

Prepared For:

Websense, Inc.

10240 Sorrento Valley Road

San Diego, CA 92121

www.websense.com

Prepared By:

SafeLogic Inc.

459 Hamilton Ave, Suite 306

Palo Alto, CA 94301

www.safelogic.com

Abstract

This document provides a non-proprietary FIPS 140-2 Security Policy for Websense Java Crypto Module.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 About FIPS 140

Federal Information Processing Standards Publication 140-2 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic modules to be deployed in a Sensitive but Unclassified environment. The National Institute of Standards and Technology (NIST) and Communications Security Establishment (CSE) Cryptographic Module Validation Program (CMVP) run the FIPS 140 program. The NVLAP accredits independent testing labs to perform FIPS 140 testing; the CMVP validates modules meeting FIPS 140 validation. *Validated* is the term given to a module that is documented and tested against the FIPS 140 criteria.

More information is available on the CMVP website at http://csrc.nist.gov/groups/STM/cmvp/index.html.

1.2 About this Document

This non-proprietary Cryptographic Module Security Policy for the Websense Java Crypto Module from Websense provides an overview of the product and a high-level description of how it meets the security requirements of FIPS 140-2. This document contains details on the module's cryptographic keys and critical security parameters. This Security Policy concludes with instructions and guidance on running the module in a FIPS 140-2 mode of operation.

Websense Java Crypto Module may also be referred to as the "module" in this document.

1.3 External Resources

The Websense website (http://www.websense.com) contains information on Websense services and products. The Cryptographic Module Validation Program website contains links to the FIPS 140-2 certificate and Websense contact information.

1.4 Notices

This document may be freely reproduced and distributed in its entirety without modification.

1.5 Acronyms

The following table defines acronyms found in this document:

Acronym	Term
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CSE	Communications Security Establishment
CSP	Critical Security Parameter
DES	Data Encryption Standard
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
EC	Elliptic Curve
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
GUI	Graphical User Interface
HMAC	(Keyed-) Hash Message Authentication Code
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
OS	Operating System
PKCS	Public-Key Cryptography Standards
PRNG	Pseudo Random Number Generator
PSS	Probabilistic Signature Scheme
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
Triple-DES	Triple Data Encryption Algorithm
TLS	Transport Layer Security
USB	Universal Serial Bus

Table 1 – Acronyms and Terms

2 Websense Java Crypto Module

2.1 Cryptographic Module Specification

The Websense Java Crypto Module provides cryptographic functions for a variety of security solutions from Websense.

The module's logical cryptographic boundary is the shared library files and their integrity check HMAC files (fortress-7.8.3-fips.jar and fortress-7.8.3-fips.hmac). The module is a multi-chip standalone embodiment installed on a General Purpose Device. The module is a software module and relies on the physical characteristics of the host platform. The module's physical cryptographic boundary is defined by the enclosure around the host platform.

All operations of the module occur via calls from host applications and their respective internal daemons/processes.

2.1.1 Validation Level Detail

The following table lists the level of validation for each area in FIPS 140-2:

FIPS 140-2 Section Title	Validation Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services, and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
Electromagnetic Interference / Electromagnetic Compatibility	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A

Table 2 – Validation Level by FIPS 140-2 Section

2.1.2 Approved Cryptographic Algorithms

The module's cryptographic algorithm implementations have received the following certificate numbers from the Cryptographic Algorithm Validation Program:

Algorithm	CAVP Certificate
AES (128-, 192-, 256-bit keys in ECB, CBC, CFB128 and OFB modes)	3192
DSA (FIPS 186-4) • Signature verification o L=1024, N=160, SHA-1 through SHA-512 o L=2048, N=224, 256, SHA-1 through SHA-512 o L=3072, N=256, SHA-1 through SHA-512 • PQG generation (Probable Primes P and Q, Unverifiable and Canonical Generation G) o L=2048, N=224, SHA-224 through SHA-512 o L=2048, N=256, SHA-256 through SHA-512 o L=3072, N=256, SHA-256 through SHA-512 • Key Pair Generation o L=2048, N=224 o L=2048, N=256 o L=3072, N=256 • Signature Generation o L=2048, N=224, SHA-224 through SHA-512 o L=2048, N=256, SHA-256 through SHA-512 o L=3072, N=256, SHA-256 through SHA-512	914
ECDSA (FIPS 186-4) • Signature Verification (SHA-1 through SHA-512) o P–curves 192, 224, 256, 384, and 521 o K–curves 163, 233, 283, 409, and 571 o B–curves 163, 233, 283, 409, and 571 • Signature Generation (SHA-224 through SHA-512) o P–curves 224, 256, 384, and 521 o K–curves 233, 283, 409, and 571 o B–curves 233, 283, 409, and 571	583
RSA (FIPS 186-4) • Key Pair Generation (X9.31) o Appendix B.3.3 o Mod 2048, 3072 o Table C.3 Probabilistic Primality Tests (2^-100) • Signature Generation (PKCS v1.5, PSS) o Mod 2048, SHA-224 through SHA-512 o Mod 3072, SHA-224 through SHA-512	1622

Algorithm	CAVP Certificate
• Signature Verification (PKCS v1.5, PSS) o Mod 1024, SHA-1 through SHA-512 o Mod 2048, SHA-1 through SHA-512 o Mod 3072, SHA-1 through SHA-512	
HMAC using SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	2011
SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	2637
SP 800-90A based HMAC-DRBG, no reseed	668
Triple-DES (two- and three-key with ECB, CBC, CFB8 and OFB modes)[1]	1818

Table 3 – FIPS-Approved Algorithm Certificates

2.1.3 Non-Approved but Allowed Cryptographic Algorithms

The module supports the following non-FIPS 140-2 approved but allowed algorithms:

- Diffie-Hellman (key agreement; key establishment methodology provides between 112 and 219 bits of encryption strength)
- EC Diffie-Hellman (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength)

2.1.4 Non-Approved Cryptographic Algorithms

The module supports the following non-approved algorithms and modes:

Algorithm	Modes or Cipher Type
DSA[2]	PQGGen, KeyGen and SigGen; non-compliant less than 112 bits of encryption strength) including FIPS 186-2 signature generation and key generation
ECDSA[2]	KeyGen and SigGen; non-compliant less than 112 bits of encryption strength) including FIPS 186-2 signature generation and key generation
RSA[2]	KeyGen and SigGen; non-compliant less than 112 bits of encryption strength
Diffie-Hellman	key agreement; key establishment methodology providing between 80 and 112 bits of encryption strength
EC Diffie-Hellman	key agreement; key establishment methodology provides between 80 and 112 bits of encryption strength
AES[2]	GCM, CFB8, CTR, CMAC, CCM

[1] The use of two-key Triple DES for encryption is restricted: the total number of blocks of data encrypted with the same cryptographic key shall not be greater than 2^20
[2] Non-compliant

Algorithm	Modes or Cipher Type
ANSI X9.31 Appendix A.2.4 PRNG[2]	(AES-128)
Blowfish	SymmetricBlockCipher[3]
Camellia	SymmetricBlockCipher[3]
CAST5	SymmetricBlockCipher[3]
CAST6	SymmetricBlockCipher[3]
ChaCha	SymmetricStreamCipher[4]
DES	SymmetricBlockCipher[3]
TDES Key Wrapping[2]	SymmetricBlockCipher[3]
ElGamal	AsymmetricBlockCipher[5]
GOST28147	SymmetricBlockCipher[3]
GOST3411	Digest
Grain128	SymmetricStreamCipher[4]
Grainv1	SymmetricStreamCipher[4]
HC128	SymmetricStreamCipher[4]
HC256	SymmetricStreamCipher[4]
IDEA	SymmetricBlockCipher[3]
IES	Key Agreement and Stream Cipher based on IEEE P1363a (draft 10)
ISAAC	SymmetricStreamCipher[4]
MD2	Digest
MD4	Digest
MD5	Digest
Naccache Stern	AsymmetricBlockCipher[5]
Noekeon	SymmetricBlockCipher[3]
Password-Based-Encryption (PBE)	• PKCS5S1, any Digest, any symmetric Cipher, ASCII • PKCS5S2, SHA1/HMac, any symmetric Cipher, ASCII, UTF8 • PKCS12, any Digest, any symmetric Cipher, Unicode
RC2	SymmetricBlockCipher[3]
RC2 Key Wrapping	SymmetricStreamCipher[4]
RC4	SymmetricStreamCipher[4]
RC532	SymmetricBlockCipher[3]
RC564	SymmetricBlockCipher[3]
RC6	SymmetricBlockCipher[3]
RFC3211 Wrapping	SymmetricBlockCipher[3]

[3] Symmetric Block Ciphers can be used with the following modes and padding: ECB, CBC, CFB, CCM, CTS, GCM, GCF, EAX, OCB, OFB, CTR, OpenPGPCFB, GOST OFB, AEAD-CCM, AEAD-EAX, AEAD-GCM, AEAD-OCB, PKCS7Padding, ISO10126d2Padding, ISO7816d4Padding, X932Padding, ISO7816d4Padding, ZeroBytePadding, TBCPadding
[4] Symmetric Stream Ciphers can only be used with ECB mode.
[5] Asymmetric Block Ciphers can be used with ECB mode and the following encodings: OAEP, PKCS1, ISO9796d1

Algorithm	Modes or Cipher Type
RFC3394 Wrapping	SymmetricBlockCipher[3]
Rijndael	SymmetricBlockCipher[3]
Ripe MD128	Digest
Ripe MD160	Digest
Ripe MD256	Digest
Ripe MD320	Digest
RSA Encryption	AsymmetricBlockCipher[5]
Salsa 20	SymmetricStreamCipher[4]
SEED	SymmetricBlockCipher[3]
SEED Wrapping	SymmetricBlockCipher[3]
Serpent	SymmetricBlockCipher[3]
Shacal2	SymmetricBlockCipher[3]
SHA-3[2]	Digest
SHA-512/t[2]	Digest
Skein-256-*	Digest
Skein-512-*	Digest
Skein-1024-*	Digest
Skipjack[2]	SymmetricBlockCipher[3]
SP 800-90A DRBG[2]	CTR, Hash
TEA	SymmetricBlockCipher[3]
TDES[2]	CFB64
Threefish	SymmetricBlockCipher[3]
Tiger	Digest
TLS v1.0 KDF[2]	Key Derivation Function
Twofish	SymmetricBlockCipher[3]
VMPC	SymmetricStreamCipher[4]
Whirlpool	Digest
XSalsa20	SymmetricStreamCipher[4]
XTEAEngine	SymmetricBlockCipher[3]

Table 4 - Non Approved Algorithms

2.2 Module Interfaces

The figure below shows the module's physical and logical block diagram:

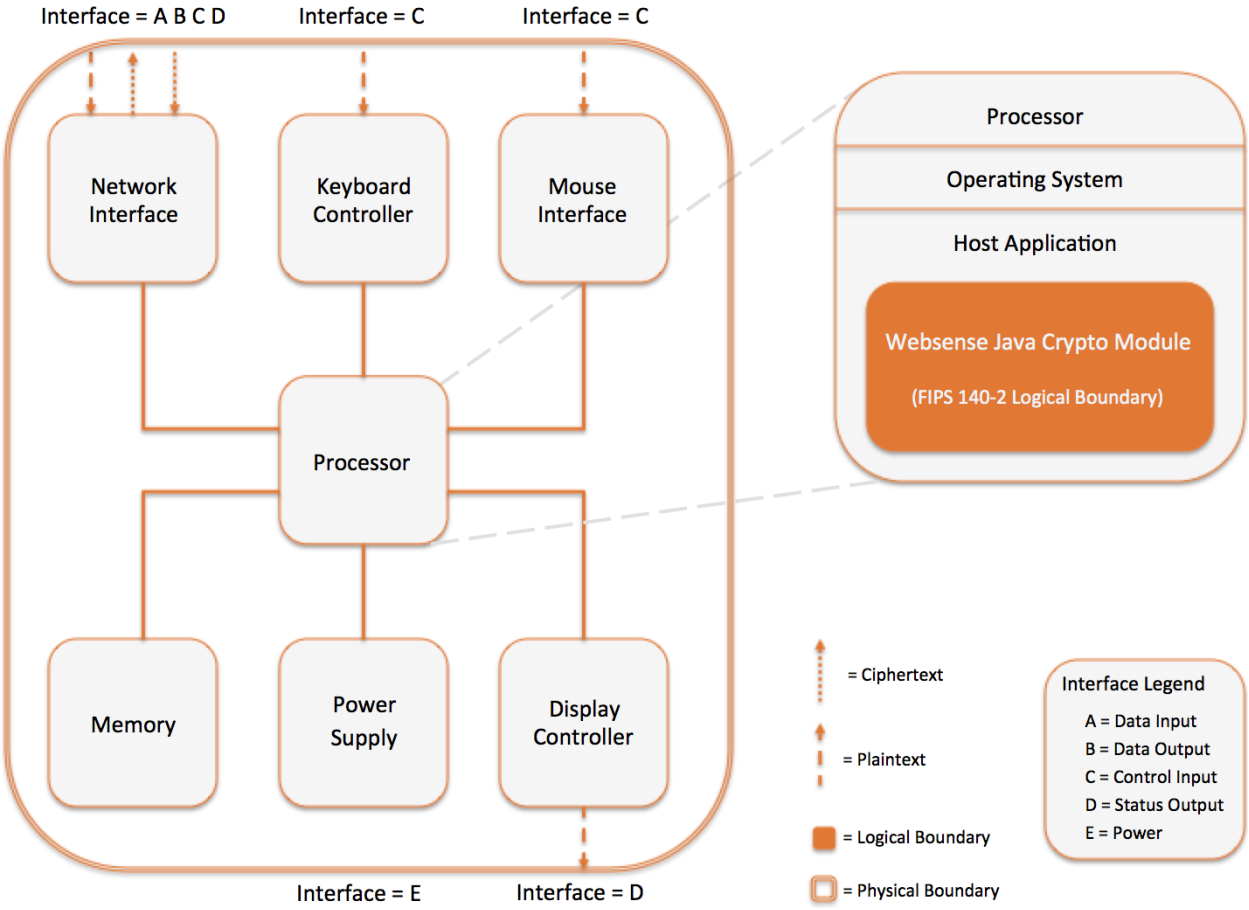

Figure 1 – Module Boundary and Interfaces Diagram

The interfaces (ports) for the physical boundary include the computer keyboard port, mouse port, network port, USB ports, display and power plug. When operational, the module does not transmit any information across these physical ports because it is a software cryptographic module. Therefore, the module's interfaces are purely logical and are provided through the Application Programming Interface (API) that a calling daemon can operate. The logical interfaces expose services that applications directly call, and the API provides functions that may be called by a referencing application (see Section 2.3 – Roles, Services, and Authentication for the list of available functions). The module distinguishes between logical interfaces by logically separating the information according to the defined API.

The API provided by the module is mapped onto the FIPS 140- 2 logical interfaces: data input, data output, control input, and status output. Each of the FIPS 140- 2 logical interfaces relates to the module's callable interface, as follows:

FIPS 140-2 Interface	Logical Interface	Module Physical Interface
Data Input	Input parameters of API function calls	Network Interface
Data Output	Output parameters of API function calls	Network Interface
Control Input	API function calls	Keyboard Interface, Mouse Interface
Status Output	Function calls returning status information and return codes provided by API function calls.	Display Controller
Power	None	Power Supply

Table 5 – Logical Interface / Physical Interface Mapping

As shown in Figure 1 – Module Boundary and Interfaces Diagram and Table 6 – Module Services, Roles, and Descriptions, the output data path is provided by the data interfaces and is logically disconnected from processes performing key generation or zeroization. No key information will be output through the data output interface when the module zeroizes keys.

2.3 Roles, Services, and Authentication

The module supports a Crypto Officer and a User role. The module does not support a Maintenance role. The User and Crypto-Officer roles are implicitly assumed by the entity accessing services implemented by the Module.

2.3.1 Operator Services and Descriptions

The module supports services that are available to users in the various roles. All of the services are described in detail in the module's user documentation. The following table shows the services available to the various roles and the access to cryptographic keys and CSPs resulting from services:

Service	Roles	CSP / Algorithm	Permission
Initialize module	CO	None	None
Show status	CO	None	None
Run self-tests on demand	CO	None	None
Zeroize key	CO	AES key DH components DRBG Entropy DRBG Seed DSA private/public key	Write

Service	Roles	CSP / Algorithm	Permission
		ECDH components ECDSA private/public key HMAC key RSA private/public key Triple-DES key	
Generate asymmetric key pair	User	RSA private/public key DSA private/public key	Write
Generate keyed hash (HMAC)	User	HMAC key	Read / Execute
Generate message digest (SHS[6])	User	None	None
Generate random number and load entropy (DRBG)	User	DRBG Seed DRBG Entropy	Read / Execute
Key agreement	User	DH components ECDH components	Write
Signature Generation	User	RSA private key DSA private key ECDSA private/public	Read / Execute
Signature Verification	User	RSA public key DSA public key ECDSA private/public	Read / Execute
Symmetric decryption	User	AES key Triple-DES key	Read / Execute
Symmetric encryption	User	AES key Triple-DES key	Read / Execute

Table 6 – Module Services, Roles, and Descriptions

When in Non-FIPS approved mode of operation, the module allows access to each of the services listed above, with exception of FIPS self-tests. When in non-FIPS-approved mode of operation the module **also** provides a service (API function call) for each non-approved algorithm listed in Section 2.1.4. These function calls are assigned to the User, and have Read/Write/Execute permission to the module's memory while in operation.

[6] SHA – Secure Hash Standard

2.3.2 Operator Authentication

As required by FIPS 140-2, there are two roles (a Crypto Officer role and User role) in the module that operators may assume. As allowed by Level 1, the module does not support authentication to access services. As such, there are no applicable authentication policies. Access control policies are implicitly defined by the services available to the roles as specified in Table 6 – Module Services, Roles, and Descriptions.

2.4 Physical Security

This section of requirements does not apply to this module. The module is a software-only module and does not implement any physical security mechanisms.

2.5 Operational Environment

The module operates on a general purpose computer (GPC) running a general purpose operating system (GPOS). For FIPS purposes, the module is running on this operating system in single user mode and does not require any additional configuration to meet the FIPS requirements.

The module was tested on the following platforms:

- OEM PowerEdge R420 running 64-bit Windows Server 2012 with Java Runtime Environment (JRE) v1.7.0_17.

The module is also supported on the following platform for which operational testing was not performed:

- OEM PowerEdge R420 running 64-bit Windows Server 2012 with Java Runtime Environment (JRE) v1.8.0_45

Compliance is maintained for other environment where the module is unchanged. No claim can be made as to the correct operation of the module or the security strengths of the generated keys when ported to an operational environment which is not listed on the validation certificate.

The GPC(s) used during testing met Federal Communications Commission (FCC) FCC Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part15, Subpart B. FIPS 140-2 validation compliance is maintained when the module is operated on other versions of the GPOS running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

2.6 Cryptographic Key Management

The table below provides a complete list of Critical Security Parameters used within the module:

Keys and CSPs	Storage Locations	Storage Method	Input Method	Output Method	Zeroization
AES key AES128, 192, 256 bit key for encryption, decryption	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
Triple-DES key Triple-DES 112, 168 bit key for encryption, decryption	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
HMAC key HMAC key for message Authentication with SHS	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
RSA private key RSA 2048, 3072 bit key for signature and key generation	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
RSA public key RSA 1024, 2048, 3072 bit key for signature verification and key generation	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
DSA private key DSA 2048, 3072-bit for signature generation	RAM	Plaintext	Input electronically in plaintext	Never	power cycle
	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
DSA public key DSA 1024, 2048, 3072-bit key for signature verification	Module Binary	Plaintext	Input electronically in plaintext	Never	power cycle
	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
ECDSA private key All NIST defined B, K,	RAM	Plaintext	Input electronically in plaintext	Never exits the module	power cycle

Keys and CSPs	Storage Locations	Storage Method	Input Method	Output Method	Zeroization
and P Curves for signature generation	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
ECDSA public key All NIST defined B, K, and P Curves for signature verification	RAM	Plaintext	Input electronically in plaintext	Never exits the module	power cycle
	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
DH public components Public components of DH protocol	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
DH private component Private exponent of DH protocol	RAM	Plaintext	Internally generated	Never	power cycle
EC DH public components Public components of EC DH protocol	RAM	Plaintext	Internally generated	Output electronically in plaintext	power cycle
EC DH private component Private exponent of EC DH protocol	RAM	Plaintext	Internally generated	Never exits the module	power cycle
DRBG seed Random data 440-bit or 880-bit to generate random number using the DRBG	RAM	Plaintext	Internally generated using nonce along with DRBG entropy input string	Never exits the module	power cycle
DRBG Entropy Input String 512-bit value to generate seed and determine random number using the DRBG	RAM	Plaintext	Externally generated; Input electronically in plaintext	Never exits the module	power cycle

R = Read W = Write D = Delete

Table 7 – Module Keys/CSPs

The application that uses the module is responsible for appropriate destruction and zeroization of the key material. The module provides functions for key allocation and destruction which overwrite the memory that is occupied by the key information with zeros before it is deallocated.

2.6.1 Random Number Generation

The module uses SP800-90A DRBG for creation of asymmetric and symmetric keys.

The module accepts input from entropy sources external to the cryptographic boundary for use as seed material for the module's Approved DRBG. Therefore, the module generates cryptographic keys whose strengths are modified by available entropy, and no assurance is provided for the strength of the generated keys.

The module performs continual tests on the output of the approved RNG to ensure that consecutive random numbers do not repeat.

2.6.2 Key/CSP Storage

Public and private keys are provided to the module by the calling process and are destroyed when released by the appropriate API function calls or during power cycle. The module does not perform persistent storage of keys.

2.6.3 Key/CSP Zeroization

The application is responsible for calling the appropriate destruction functions from the API. The destruction functions then overwrite the memory occupied by keys with zeros and deallocates the memory. This occurs during process termination / power cycle. Keys are immediately zeroized upon deallocation, which sufficiently protects the CSPs from compromise.

2.7 Self-Tests

FIPS 140-2 requires that the module perform self tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. In addition some functions require continuous verification of function, such as the random number generator. All of these tests are listed and described in this section.

If any self-test fails, the module will enter a critical error state, during which cryptographic functionality and all data output is inhibited. To clear the error state, the CO must reboot the host system, reload the module, or restart the calling application. No operator intervention is required to run the self-tests.

The following sections discuss the module's self-tests in more detail.

2.7.1 Power-On Self-Tests

Power-on self-tests are executed automatically when the module is loaded into memory. The module verifies the integrity of the runtime executable using a HMAC-SHA-512 digest computed at build time. If the fingerprints match, the power-up self-tests are then performed. If the power-up self-tests are successful, the module is in FIPS mode.

TYPE	DETAIL
Software Integrity Check	• HMAC-SHA512 on all module components
Known Answer Tests	• AES encrypt and decrypt KATs • Triple-DES encrypt and decrypt KATs • HMAC SHA1 KAT • HMAC SHA-256 KAT • HMAC SHA-512 KAT • RSA sign and verify KATs • SP 800-90A DRBG KAT (HMAC)
Pair-wise Consistency Tests	• DSA • ECDSA • Diffie-Hellman • EC Diffie-Hellman

Table 8 – Power-On Self-Tests

Input, output, and cryptographic functions cannot be performed while the Module is in a self-test or error state because the module is single-threaded and will not return to the calling application until the power-up self tests are complete. If the power-up self tests fail, subsequent calls to the module will also fail - thus no further cryptographic operations are possible.

2.7.2 Conditional Self-Tests

The module implements the following conditional self-tests upon key generation, or random number generation (respectively):

TYPE	DETAIL
Pair-wise Consistency Tests	• DSA • ECDSA • Diffie-Hellman • EC Diffie-Hellman
Continuous RNG Tests	• SP 800-90A DRBG (HMAC)

Table 9 – Conditional Self-Tests

2.8 Mitigation of Other Attacks

The Module does not contain additional security mechanisms beyond the requirements for FIPS 140-2 Level 1 cryptographic modules.

3 Guidance and Secure Operation

3.1 Crypto Officer Guidance

3.1.1 Software Installation

The module is provided directly to solution developers and is not available for direct download to the general public. The module and its host application is to be installed on an operating system specified in Section 2.5 or one where portability is maintained.

In order to remain in FIPS-approved mode, the following steps must be taken during the installation process:

1. The Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 7 must be installed in the JRE. Instructions for installation are found in the download file located here: http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html

2. The module must be configured as the JRE's default Security Provider by modifying the jre/lib/security/java.security file and adding the following line to the list of providers:

   ```
   security.provider.1=com.websense.fortress.jce.provider.BouncyCastleProvider
   ```

3.1.2 Additional Rules of Operation

1. The writable memory areas of the module (data and stack segments) are accessible only by the application so that the operating system is in "single user" mode, i.e. only the application has access to that instance of the module.

2. The operating system is responsible for multitasking operations so that other processes cannot access the address space of the process containing the module.

3.2 User Guidance

3.2.1 General Guidance

The module is not distributed as a standalone library and is only used in conjunction with the solution.

The end user of the operating system is also responsible for zeroizing CSPs via wipe/secure delete procedures.

If the module power is lost and restored, the calling application can reset the IV to the last value used.

3.2.2 FIPS-Approved Mode of Operation

In order to maintain the FIPS-approved mode of operation, the following requirements must be observed:

1. The calling application must instantiate and operate the module through the JCE interface provided by the JDK.
2. The calling application may not share CSPs between non-FIPS-approved-mode and FIPS-approved-mode of operation. The operator must reset the module before switching to FIPS-approved-mode of operation.
3. The calling application must restrict the use of two-key Triple DES encryption: the total number of blocks of data encrypted with the same cryptographic key shall not be greater than 2^20.
4. The module requires that a minimum of 256 bits of entropy be provided for each use of the DRBG / load entropy service.