Proofpoint Security Library

FIPS 140-2 Non-Proprietary Security Policy

Level 1 Validation

Version 2.0

Jan 2014
Multi-chip standalone

Table of Contents

1 Introduction

1.1 Purpose

This is a non-proprietary cryptographic module security policy for the Proofpoint Security Library (the Cryptographic Module), version 2.0[1]. This security policy describes how the Proofpoint Security Library meets the security requirements of FIPS 140-2, and how to operate the Proofpoint Security Library in a FIPS 140-2 compliant manner. This policy was prepared as part of the Level 1 FIPS 140-2 validation of the Proofpoint Security Library.

1.2 Terminology

Throughout this document the Proofpoint Security Library is also referred to as the module.

1.3 References

Additional information on Proofpoint can be found at http://www.proofpoint.com. Additional information on FIPS 140-2, including a list of FIPS-approved algorithms, can be found at http://www.nist.gov/cmvp.

2 The Proofpoint Security Library

The Proofpoint Security Library is a Java language cryptography component used by Proofpoint's security products.

2.1 Cryptographic Module

The module's environment consists of the following generic components:

1) A commercially available general-purpose hardware-computing platform.
2) A commercially available Operating System (OS) that runs on the above platform.
3) The Java Runtime Environment.

The Proofpoint Security Library then executes on the above platform, operating system, and Java runtime environment.

[1] The version of security library can be determined by examining the Specification Version header in the MANIFEST.MF file within the library

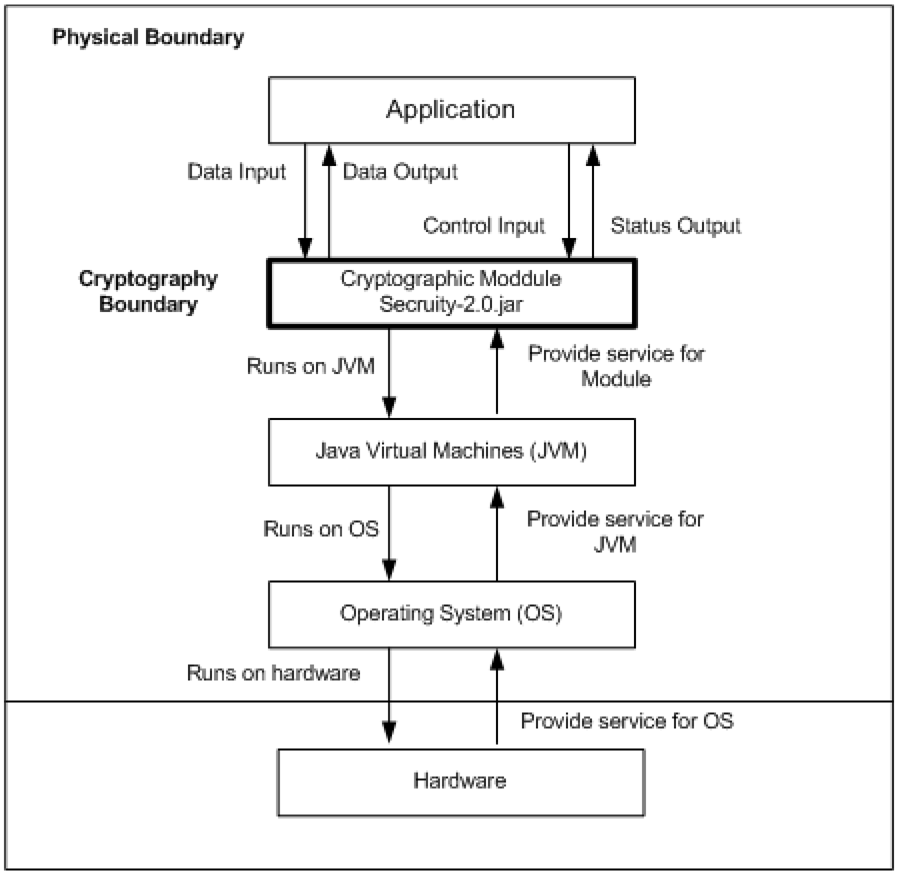

Figure 1- Proofpoint Security Library Logical and Physical Diagram

The logical boundary for this module is described in the diagram above. The physical boundary is the general-purpose PC and operating system. The module is suitable for any general-purpose PC and operating system capable of running JRE 1.6 or later. 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.

2.2 Module Interfaces

The physical interfaces of the module are those of the general-purpose hardware-computing platform hosting the module, including: a computer keyboard, mouse, screen, floppy drives, CDROM drives, speakers, microphone inputs, serial ports, parallel ports, and power plug. The logical interface is the Application Programming Interface (API) of the library. The API is classified in terms of the FIPS 140-2 logical interfaces as follows:

- Data input – input parameters to all functions available to operators assuming the User role
- Data output – output parameters from all functions that produce output
- Control input – input parameters to all functions available to operators assuming the Crypto Officer role
- Status output – information returned via exception

2.3 Roles and Services

2.3.1 Roles

The module supports two distinct roles: a Crypto Officer role and a User role.

Role	**Type of authentication**	**Authentication data**
User	None	N/A
Crypto Officer	None	N/A

As allowed by FIPS 140-2 level 1, the module does not support user identification or authentication. Only one role may be active at a time. The module does not allow concurrent operators.

Authentication mechanism	**Strength of mechanism**
None	N/A

2.3.2 Services

The module provides several types of cryptographic services. The following table describes the type of access to cryptographic keys and CSPs available to operators exercising each type of service. All services from the module are provided in the form of a callable method.

Approved Services:

Service/Method	Description	Cryptographic keys and CSPs	Types of access
Cipher AES	Cipher – an abstract class for a symmetric cipher. AES - implementation of Advanced Encryption	Symmetric keys	R/W/X

	Standard.		
ECDSA	Implementation of the Elliptic Curve Digital Signature Algorithm. All methods dealing with keys are static, so no keys are stored in an ECDSA instance. Signature Generation with SHA-1 is not allowed in the approved mode.	ECDSA private/public keys	R/W/X
Hash SHA1, SHA-224, SHA-256, SHA-384, SHA-512	HASH - abstract class for a secure hash. Implantation for SHA1, SHA-224, SHA-256, SHA-384, SHA-512.	None	
POST	Power-On Self-Test. Verifies algorithms in security library pass Known Answer Tests, that the signature file is present in the JAR, and that signature is verified	None	
Random SHA1PRNG	SHA1PRNG - a pseudo random number generator based on SHA-1.	DRNG secret values	W/X
RSA	Implementation of the RSA Signature Algorithm. Keys are provided to static methods and not stored. Signature Generation with SHA-1 is not allowed in the approved mode.	RSA Public/Private keys	X

Non-approved Services:

Service/Method	Description	Cryptographic keys and CSPs	Types of access
Cipher RC2 Triple-DES	Cipher – an abstract class for a symmetric cipher. RC2 - implementation of Rivest Cipher 2. Triple-DES - implementation of Triple Data Encryption Standard. For all implementations, keys are passed in and keys are zeroed by calling a finalize method.	Symmetric keys	R/W/X
DiffieHellman	Implementation of Diffie-Hellman key exchange algorithm. Secret and session keys and DHPublicValue public key. Keys are zeroed by calling finalize method. (Although Diffie-Hellman is not a FIPS approved service, it can be used in a FIPS approved mode.)	DH Private keys	R/W/X
DSA	Implementation of Digital Signature Algorithm. All methods dealing with keys are static, so no keys are stored in a DSA instance.	DSA Public/Private keys	R/W/X
ECDSA signature generation with SHA-1	Generation of ECDSA Signature with SHA-1	None	
ESRP	Extended Secure Remote Password protocol implementation. Keys are generated based on	DH Private keys	R/W/X

	parameters provided to constructors and are stored in DHPublicValue and BigInteger objects. Keys are zeroed in the finalize call.		
Random AESPRNG	AESPRNG - a pseudo random number generator based on AES.	DRNG secret values	W/X
RSA Signature Generation with SHA-1	Generation of RSA Signature with SHA-1	None	
MAC	Message Authentication Code implementation using HMAC-SHA1. Keys are zeroed in the finalize call.	HMAC-SHA-1 keys	R/W/X

The authorized services available to each role are described below.

2.3.2.1 Crypto Officer Services

Crypto Officers may execute power-up self-tests on demand. Operators assuming the Crypto Officer role have no access to any critical security parameters, including cryptographic keys.

Role	**Authorized Services**
Crypto Officer	POST: On-demand execution of power-on self-tests

2.3.2.2 User Services

An operator assuming the User role can exercise all services provided by the module except for the on-demand invocation of power-up self-tests, which is reserved for Crypto Officers. Operators assuming the User role may read/write critical security parameters, including cryptographic keys, via invocation of API methods.

Role	**Authorized Services**
User	All services in section 2.3.2 other than POST, which allows: Symmetric key cryptography Asymmetric key cryptography Hash

	Key agreement Random number generation Zeroization by deleting the key object

2.4 Physical Security

The module is a software module intended for use on a variety of platforms including Microsoft Windows XP, Vista, and Win7, Linux, Solaris and other UNIX variants. Since the module is a software module, it can be exempted from the physical security requirements of the FIPS 140-2 standard.

2.5 Software and Operating System Security

The Proofpoint Security Library is a software module tested on the CentOS 5 operating system running with Sun JRE 1.6.0. The library will also operate under Windows XP, Vista, and Win7, Linux, Solaris and other UNIX variants.

The module consists of a single, signed JAR file. As explained below, a cryptographic mechanism is used within the module to ensure that the code has not been accidentally or ineptly modified from its validated configuration.

2.6 Cryptographic Key Management

The Proofpoint Security Library securely administers cryptographic keys, including ephemeral session keys. All session keys are ephemeral and are discarded immediately after use.

2.6.1 Key Generation

The module generates keys using a FIPS approved PRNG (FIPS 186-2, Appendix 3.1, using SHA-1 to construct the function G). The PRNG allows the use of an optional XSEED. The module also implements a non-approved RNG (AES RNG) , which is not allowed for use in FIPS mode.

The module does not provide symmetric key generation. The application must always pass in the key as an argument to the classes for each of the approved symmetric algorithms.

2.6.2 Key Storage

The module does not store secret or private key material.

2.6.3 Key Zeroization

All ephemeral key data resides in internally allocated data structures that are zeroized by deletion of the object. An operator can initiate key zeroization by deleting the key object.

2.7 Cryptographic Algorithms

When operating in FIPS mode, the Proofpoint Security Library supports the following algorithms for the following purposes, key sizes, and cipher modes:

- ECDSA (Cert. #250)
    - All key sizes
- Secure Hashing Algorithm (SHA1, SHA-224, SHA-256, SHA-384, SHA-512) (Cert. #1591)
    - Byte oriented mode
- Advanced Encryption Standard (AES) – FIPS 197 (Cert. #1814)
    - Encryption/decryption
    - 128, 192, 256 bit keys
    - ECB or CBC modes
- RNG – FIPS 186-2, Appendix 3.1 (Cert. #956)
- RSA (Cert. #909)
    - Signature verification
    - 1024-4096 bit modulus supported

In addition to the above approved cryptographic algorithms, the module also provides the following non-approved algorithms which are not allowed in FIPS mode unless noted:

- AES RNG – AES based PRNG
- Diffie-Hellman key agreement (Although Diffie-Hellman key agreement is not a FIPS approved algorithm, it can be used in a FIPS approved mode. Key establishment methodology provides between 112 and 256 bits of encryption strength; non-compliant less than 112-bits of encryption strength)
- DSA (non-compliant)
- Extended Remote Password (ESRP)
- Secure Remote Password (SRP)
- Triple DES (non-compliant)
- Entropy Gathering Daemon(EGD) (RNG) (Although the EGD is not a FIPS approved algorithm it is allowed in approved mode because it seeds the approved RNG)

The module generates cryptographic keys whose strengths are modified by available entropy. Each of the generated keys provides a minimum of 112-bits of encryption strength.

2.8 Self-Tests

The module performs a number of startup and conditional self-tests to ensure proper operation (see Table 1 for a list of all self-tests performed by the module). If the module fails a self-test it will enter an error state and inhibit all cryptographic functions and data output. Self-tests include integrity checks over the library at load time, cryptographic algorithm known answer tests (KATs) and other critical startup tests. Additionally, a continuous random number generator tests monitors output from the module’s FIPS-approved random number generator, as required by FIPS 140-2.

Test	**Type**
FIPS 186-2 RNG Continuous random number generator test	Conditional Self-Test

AES RNG Continuous random number generator test	Conditional Self-Test
Entropy Gathering Daemon	Conditional Self-Test
Pairwise consistency test for RSA	Conditional Self-Test
Pairwise consistency test for ECDSA	Conditional Self-Test
ECDSA KAT	Power-up Self-Test
RSA Sign/Verify	Power-up Self-Test
Module integrity check	Power-up Self-Test
SHA-1 KAT	Power-up Self-Test
SHA-256 KAT	Power-up Self-Test
SHA-512 KAT	Power-up Self-Test
SHA-384 KAT	Power-up Self-Test
SHA-224 KAT	Power-up Self-Test
Triple DES KAT	Power-up Self-Test
AES KAT	Power-up Self-Test
PRNG KAT	Power-up Self-Test

Table 1 – Summary of FIPS required self-tests

2.9 Mitigation of other attacks

The cryptographic module is not designed to mitigate any specific attacks.

Other attacks	**Mitigation mechanism**	**Specific limitations**
None	N/A	N/A

3 Secure Operation of the Proofpoint Security Library

The module does not require any special configuration to operate in conformance with FIPS 140- 2 requirements. FIPS 140-2 requires that only FIPS-approved algorithms be used when operating a FIPS 140-2 compliant manner. Thus, to operate the module in conformance with FIPS 140-2 requirements, only the FIPS-approved algorithms listed in section 2.7 may be used.

Note: It is the User’s responsibility to understand which algorithms are FIPS-approved and which are not. NIST supports a web site (referenced in section 1.3) that lists validated implementations of NIST-approved cryptographic algorithms. Only random numbers generated using the approved random number generator may be used during key generation in FIPS mode.

4 Acronym List

Acronym	**Definition**
AES	Advanced Encryption Standard
API	Application Programming Interface
DSS	Digital Signature Standard

EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
ESRP	Extended Secure Remote Password
FCC	Federal Communication Commission
FIPS	Federal Information Processing Standard
JAR	Java Archive
JRE	Java Runtime Environment
KAT	Known Answer Test
NIST	National Institute of Standards and Technology
OS	Operating System
PC	Personal Computer
SHA1	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
Triple DES	Triple Data Encryption Standard