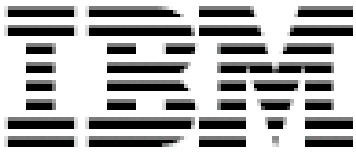

IBM Java JCE FIPS 140-2 Cryptographic Module with CPACF

Version 1.8

FIPS 140-2 Non-Proprietary Security Policy

Version 1.2

Last update: 2017-02-08

Prepared by:
atsec information security corporation
9130 Jollyville Road, Suite 260
Austin, TX 78759
www.atsec.com

Table of Contents

© 2017 IBM Corporation / atsec information security.

Copyrights and Trademarks

© 2017 IBM Corporation / atsec information security. This document can be reproduced and distributed only whole and intact, including this copyright notice.

Java is a registered trademark of Oracle. Inc.

z/OS and IBM are trademarks or registered trademarks of IBM Corporation in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds.

Red Hat is a trademark of Red Hat, Inc.

SuSE is a registered trademark of SuSE AG

Other company, product, and service names may be trademarks or service marks of others.

1. Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for the IBM Java JCE FIPS 140-2 Cryptographic Module with CPACF. The version of the module is 1.8. This policy contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 module.

© 2017 IBM Corporation / atsec information security.

2. Cryptographic Module Specification

The following section describes the cryptographic module and how it conforms to the FIPS 140-2 specification in each of the required areas.

2.1. Module Overview

The IBM Java JCE FIPS 140-2 Cryptographic Module with CPACF (hereafter referred to as "the module") is a scalable, multi-purpose cryptographic module that supports FIPS approved cryptographic operations.

The module acts as a Java Cryptographic Extension (JCE) provider in the JCE framework, providing cryptographic services to Java applications.

The block diagram below shows the logical boundary of the module and its interfaces with the calling application.

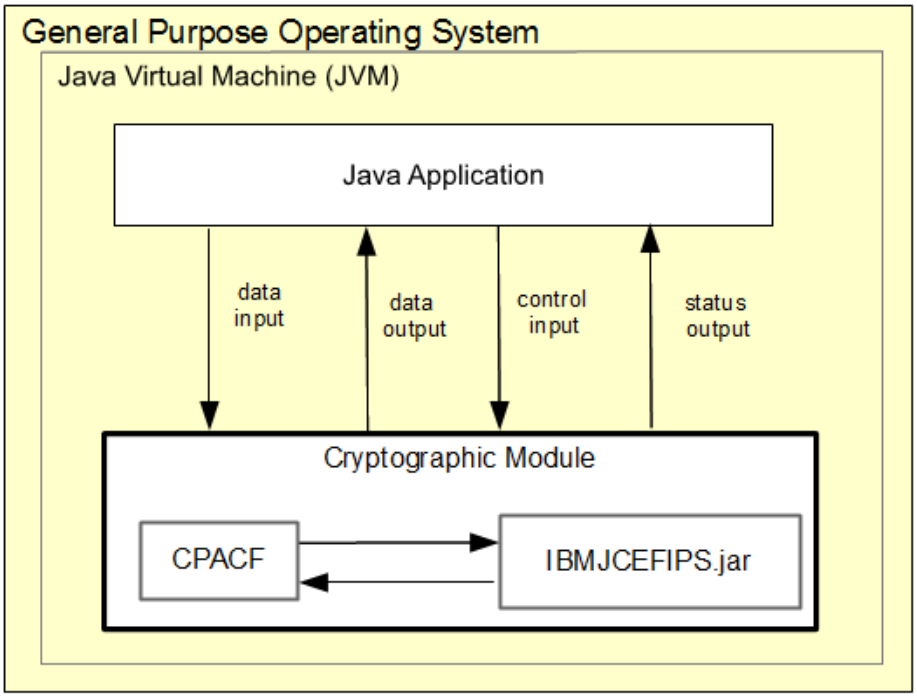

Figure 1 - Block Diagram

The module is implemented as a Java Archive (JAR). The logical boundary of the module consists of the ibmjcefips.jar file and the CPACF. The CP Assist for Cryptographic Function (CPACF) is a hardware device part of the Co- Processor Unit (CoP). It provides cryptographic algorithms implementations. This module makes use of AES, Triple-DES and SHA algorithms implemented in CPACF. The CPACF consists of the following components:

- Firmware – CP Assist for Cryptographic Functions Feature 3863 (aka FC3863) with System Driver Level 22H
- Hardware – COP chips integrated within processor unit

The module runs on a General Purpose Computer (GPC). The physical boundary of the module is the enclosure of the GPC on which the module is installed and executed, as shown with dotted lines in the diagram below:

© 2017 IBM Corporation / atsec information security.

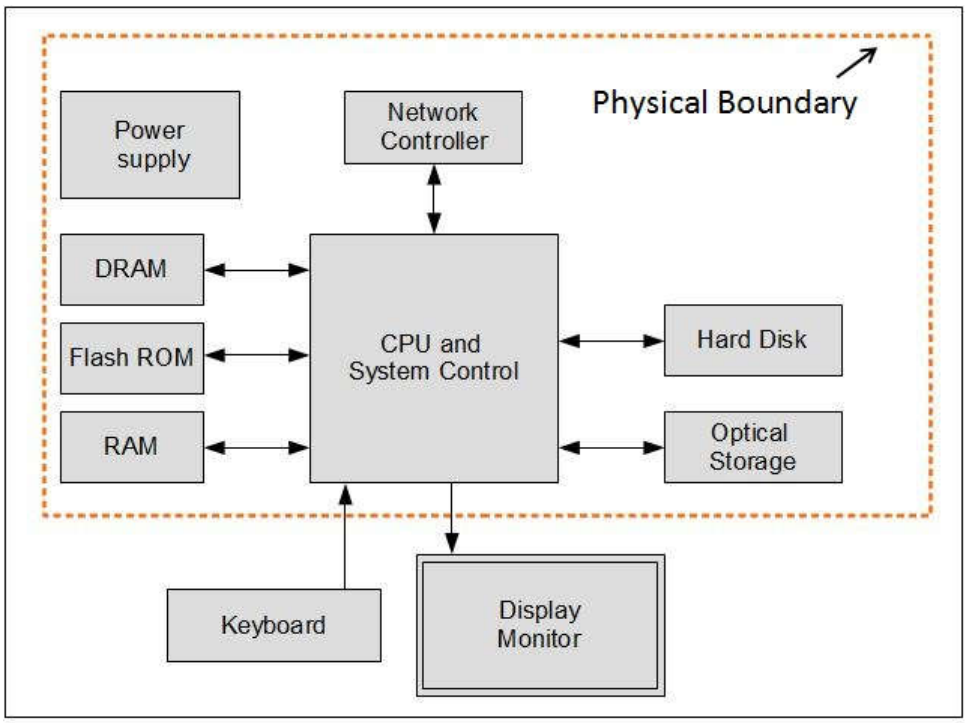

Figure 2 – Hardware Block Diagram

2.2. FIPS 140-2 Validation

For the purpose of the FIPS 140-2 validation, the module is a software-hybrid, multi-chip standalone cryptographic module validated at overall Security Level 1. The table below shows the security level claimed for each of the eleven sections that comprise the FIPS 140-2 standard:

FIPS 140-2 Section		Security Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services and Authentication	1
4	Finite State Model	1
5	Physical Security	1
6	Operational Environment	1
7	Cryptographic Key Management	1
8	EMI/EMC	1
9	Self-Tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	1
Overall Level		1

Table 1 - Security Levels

© 2017 IBM Corporation / atsec information security.

The module has been tested by the laboratory on the platforms shown in the table below. Each platform includes the hardware, processor name, operating system, and the hardware accelerator implemented in the module. The module runs on 64-bit Java Virtual Machine (JVM), Version 8.

Hardware Platform and Processor	Operating System
IBM z13 model N63	z/OS version 2 release 2
IBM z13 model N63	Red Hat Enterprise Linux Server release 7.2 for IBM z Systems

Table 2 - Tested Platforms

In addition to the configurations tested by the laboratory, the vendor affirmed testing was performed on the platform shown in the table below. Each platform includes the hardware, processor name, operating system and JDK Bit level. The module runs on Java Virtual Machine (JVM), Version 8.

Hardware Platform and Processor	Operating System	JDK Bit level
IBM z13 model N63	SuSE SLES 11 SP4	64

Table 2(A) - Vendor Affirmed Platforms

As outlined in G.5 of the Implementation Guidance for FIPS 140-2, the module maintains its compliance on other operating systems provided:

- The operating system meets the operational environment requirements at the module's level of validation, and runs in a single-user mode.
- The module does not require modification to run in the new environment.

Note: *CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.*

2.3. Modes of Operation

The module supports both FIPS mode (i.e. Approved mode of operation) and non-FIPS mode (i.e., non-Approved mode of operation).

After successful completion of power-on self-tests, the module enters FIPS mode. Any calls to the non-Approved security functions as listed in Table 5, will cause the module to implicitly transition to the non-Approved mode of operation.

The keys and CSPs used for cryptographic operations are not shared between the modes of operation.

© 2017 IBM Corporation / atsec information security.

3. Cryptographic Module Ports and Interfaces

For the purpose of FIPS 140-2 validation, the physical ports are interpreted to be the physical ports of the hardware platform on which it runs.

The logical interfaces are the Application Program Interface (API) through which Java applications request services. The following table summarizes the four logical interfaces:

Logical Interface	Description
Data Input	API input parameters for data.
Data Output	API output parameters for data.
Control Input	API function calls, API input parameters for control.
Status Output	API return values, API output parameters for status, API error messages.

Table 3 - Ports and Interfaces

© 2017 IBM Corporation / atsec information security.

4. Roles, Services and Authentication

4.1. Roles

The module supports the following roles:

- User role: performs all services as listed in Table 4 and Table 5 except module installation and configuration.
- Crypto Officer role: performs module installation and configuration as listed in Table 4.

The User and Crypto Officer roles are implicitly assumed by the entity accessing the module services.

The module does not support the maintenance role.

4.2. Services

The module provides services to users that assume one of the available roles. All services are described in detail in the user documentation.

Table 4 lists the Approved services and the non-Approved but allowed services in FIPS mode of operation, the roles that can request the service, the algorithms involved with their corresponding CAVS certificate numbers (if applicable), the Keys/CSPs involved and how they are accessed:

Services	Algorithms	CAVS Certificates	Keys/CSPs	Access
AES key generation	NIST SP800-90A Hash_DRBG with SHA-256	#1124, #1125	128/192/256 bits AES key	Write
Triple-DES key generation			192 bits three-key Triple-DES key	Write
HMAC Key Generation			At least 112 bits HMAC key	Write
AES encryption and decryption	AES (ECB, CBC, OFB, CFB8, CFB128 and GCM modes)	#3909, #3910	128/192/256 bits AES key	Read
Three-key Triple-DES encryption and decryption	Three-key Triple-DES (ECB, CBC, CFB8, CFB64 and OFB modes)	#2145, #2146	192 bits three-key Triple-DES key	Read
Two-key Triple-DES decryption (Legacy Use)	Two-key Triple-DES (ECB, CBC, CFB8, CFB64 and OFB modes)		128 bits two-key Triple-DES key	Read
RSA key generation	FIPS186-4 Appendix B.3.3 RSA key generation	#1993, #1994	RSA public and private key pair with 2048/ 3072 bits modulus size	Write

© 2017 IBM Corporation / atsec information security.

Services	Algorithms	CAVS Certificates	Keys/CSPs	Access
RSA signature generation	PKCS#1 v1.5 and PSS RSA signature generation with SHA-224, SHA-256, SHA-384 and SHA-512		RSA private key with 2048/3072 bits modulus size	Read
RSA signature verification	PKCS#1 v1.5 and PSS RSA signature verification with SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512		RSA public key with 1024/2048/3072 bits modulus size	Read
RSA key transport	NIST SP800-56B RSA key transport with OAEP	Vendor-affirmed according to IG D.4	RSA public & private key pair with 2048/ 3072 bits modulus	Read
	Non SP800-56B Key Transport with PKCS1 or zero or no padding	(IG D.9 allowed till December 31, 2017)	RSA public & private key pair with 2048/ 3072 bits modulus	Read
DSA key generation	FIPS186-4 DSA key pair generation	#1067, #1068	DSA public and private key pair with L=2048, N=256; L=3072, N=256	Write
DSA domain parameter generation	DSA domain parameter generation with SHA-256, SHA-384 and SHA-512		DSA domain parameters with L=2048, N=224; L=2048, N=256; L=3072, N=256	Write
DSA signature generation	DSA signature generation with SHA-256		DSA private key with L=2048, N=224; L=2048, N=256; L=3072, N=256	Read
DSA signature verification	DSA signature verification with SHA-1, SHA-224 and SHA-256		DSA public key with L=1024, N=160; L=2048, N=224; L=2048, N=256; L=3072, N=256	Read
ECDSA key generation	FIPS186-4 Appendix B.4.1 ECDSA key generation	#852, #853	ECDSA public and private key pair according to P-224, P-256, P-384 and P-521 curves	Write
ECDSA signature generation	ECDSA signature generation with SHA-224, SHA-256, SHA-384 and SHA-512		ECDSA private key according to P-224, P-256, P-384 and P-521 curves	Read

Services	Algorithms	CAVS Certificates	Keys/CSPs	Access
ECDSA signature verification	ECDSA signature verification with SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512		ECDSA public key according to P-192, P-224, P-256, P-384 and P-521	Read
Message digest	SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512	#3221, #3222	n/a	n/a
Message authentication	HMAC with SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512	#2538, #2539	At least 112 bits HMAC key	Read
Random number generation	NIST SP800-90A Hash_DRBG with SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512	#1124, #1125	Entropy input string, V and C values	Read, Write
	NDRNG (Used by the module to internally seed the DRBG.)	Non-approved but allowed to be used in FIPS mode		Read
Diffie-Hellman key agreement	NIST SP800-56A KAS FFC except KDF	CVL#769, #771	Diffie-Hellman public and private primitives with 2048 bits key size	Write Read
EC Diffie-Hellman key agreement	NIST SP800-56A KAS ECC except KDF; Section 5.7.1.2 ECC CDH Primitive		EC Diffie-Hellman public and private primitives with P-224, P-256, P-384 and P-521 curves	Write Read
Key derivation	NIST SP800-135 key derivation in TLS 1.0, 1.1 and 1.2	CVL#768, #770	TLS pre-master secret and TLS master secret	Write Read
Show status	n/a	n/a	n/a	n/a
Self-Tests	n/a	n/a	HMAC key for module integrity test	Read
Zeroization	n/a	n/a	All aforementioned Keys/CSPs	Zeroize
Module installation and configuration	n/a	n/a	n/a	n/a

Table 4 - Services in FIPS mode of operation

Note: TLS protocol implementation has not been reviewed or tested by the CAVP and CMVP.

© 2017 IBM Corporation / atsec information security.

Table 5 lists the non-Approved services only available in non-FIPS mode of operation.

Services	Algorithms	Access
AES encryption and decryption	AES (CTS and PCBC modes)	Read
Three-key or Two-key Triple-DES encryption and decryption	Three-key or Two-key Triple-DES (CTS and PCBC modes)	Read
Two-key Triple-DES encryption	Two-key Triple-DES (ECB, CBC, CFB8, CFB64 and OFB modes)	Read
RSA key generation	RSA key generation with modulus size between 512 and 16384 bits excluding 2048 and 3072 bits	Write
RSA signature generation	RSA signature generation with modulus size between 512 and 16384 bits excluding 2048 and 3072 bits or RSA signature generation using SHA-1	Read
RSA signature verification	RSA signature verification with modulus size between 512 and 16384 bits excluding 1024, 2048 and 3072 bits	Read
SP800-56B compliant RSA key transport	RSA key transport with modulus size between 512 and 16384 bits excluding 2048 and 3072 bits	Read
Non SP800-56B compliant RSA Key Transport	RSA key transport using PKCS1-v1_5 or zero or no padding with modulus size between 512 and 16384 bits excluding 2048 and 3072 bits	Read
RSA signature generation and verification without hashing	RSA signature generation and verification without hashing and with modulus size between 512 and 16384 bits	Read
DSA key generation	DSA key generation with key size between 512 and 3072 bits which is multiple of 64 but excluding 2048 and 3072 bits	Write
DSA signature generation	DSA signature generation with key size between 512 and 3072 bits which is multiple of 64 but excluding 2048 and 3072 bits	Read
DSA signature verification	DSA signature verification with key size between 512 and 3072 bits which is multiple of 64 but excluding 1024, 2048 and 3072 bits	Read
DSA signature generation and verification without hashing	DSA signature generation and verification without hashing and with key size between 512 and 3072 bits which is multiple of 64	Read
ECDSA key generation	ECDSA key generation with P-192 curve	Write
ECDSA signature generation	ECDSA signature generation with P-192 curve or ECDSA signature generation using SHA-1	Read

© 2017 IBM Corporation / atsec information security.

Services	Algorithms	Access
ECDSA signature generation and verification without hashing	ECDSA signature generation and verification without hashing and with P-192, P-224, P-256, P-384 and P-521 curves	Read
Message digest	MD5	n/a
Message authentication	HMAC with keys less than 112 bits	Read
Diffie-Hellman key agreement	Diffie-Hellman key agreement with key size between 256 and 2048 bits which is multiple of 64 but excluding 2048 bits	Write
EC Diffie-Hellman key agreement	EC Diffie-Hellman key agreement with P-192 curve	Write

Table 5 - Services in non-FIPS mode of operation

Note: The module implements non-Approved NDRNG algorithm which is allowed to be used in FIPS mode to internally seed the DRBG. NDRNG is not available as an external service provided by the module.

4.3. Operator Authentication

The module does not implement user authentication. The role of the user is implicitly assumed depending on the service used.

Notice that at Security Level 1, authentication is not required.

© 2017 IBM Corporation / atsec information security.

5. Physical Security

The module is a software-hybrid module and physical security is applicable. The module inherits the physical characteristics of the host running it. The module has no physical security characteristics of its own.

CPACF is made of production grade components and included within the physical boundary of the module, being the IBM z13 mainframe computer.

© 2017 IBM Corporation / atsec information security.

6. Operational Environment

6.1. Applicability

The module operates in a modifiable operational environment per FIPS 140-2 Security Level 1 specifications. The module runs on a GPC as specified in Table 2 of section 2.2.

6.2. Policy

The operating system is restricted to a single operator (i.e., concurrent operators are explicitly excluded).

The Java application that requests cryptographic services is the single user of the module.

© 2017 IBM Corporation / atsec information security.

7. Cryptographic Key Management

The management of all Keys/CSPs used by the module is summarized in the table below:

Name	Generation	Storage	Zeroization
AES keys	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
Triple-DES keys	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
RSA key pair	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
DSA key pair	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
ECDSA key pair	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
HMAC key	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
DRBG entropy input string	Obtained from the NDRNG	RAM	Zeroized by zeroize() method
DRBG V and C values	Derived from the entropy string as defined in NIST SP800-90A	RAM	Zeroized by zeroize() method
Diffie-Hellman Primitives	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
EC Diffie-Hellman Primitives	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
TLS pre-master secret	NIST SP800-90A Hash_DRBG	RAM	Zeroized by zeroize() method
TLS master secret	Use of the TLS pre-master secret and PRF	RAM	Zeroized by zeroize() method

Table 6 - Life cycle of Keys/CSPs

The following sections describe how Keys/CSPs are managed during its life cycle.

7.1. Key/CSP Generation

The module implements symmetric key generation using a Hash_DRBG compliant with [NIST SP800-90A]. For generating RSA, DSA and ECDSA key pairs, the module implements asymmetric key generation services compliant with [FIPS 186-4].

7.2. Key/CSP Establishment

The module implements Diffie-Hellman key agreement scheme, EC Diffie-Hellman key agreement scheme and vendor affirmed RSA key transport scheme.

- Diffie-Hellman with 2048 bit key provides 112 bits of encryption strength; non-compliant less than 112 bits of encryption strength.
- EC Diffie-Hellman with P-224, P-256, P-384 and P-521 curves provides between 112 and 256 bits of encryption strength; non-compliant less than 112 bits of encryption strength

© 2017 IBM Corporation / atsec information security.

- RSA key transport with 2048 or 3072 bit modulus provides 112 or 128 bits of encryption strength; non-compliant less than 112 bits of encryption strength

7.3. Key/CSP Entry and Output

All the keys enter into the module's logical boundary as API input parameters. They are associated with memory locations and do not persist across power cycles. The module does not support manual key entry. The module does not output intermediate key generation values or other CSPs. The module provides the keys as output parameters of the key generation service API to the Java application, but they do not cross the module's physical boundary.

7.4. Key/CSP Storage

The HMAC key used for integrity test is stored in the module and relies on the operating system for protection. The module does not perform persistent storage for any other Keys/CSPs. The module does not store any Key/CSP beyond the lifetime of the API call.

7.5. Key/CSP Zeroization

Except for the HMAC key used for integrity test, all keys/CSPs are ephemeral and are destroyed when released by the appropriate API function calls. The application is responsible for calling the zeroization methods as listed Table 6. The zeroization methods overwrite the memory occupied by keys with "zeros" and deallocate the memory with the regular memory deallocation operating system call.

7.6. Random Number Generation

The module employs a NIST SP800-90A-compliant HASH (SHA-256) DRBG as a random number generator for the generation of Keys/CSPs. The module uses a CPU jitter based random number generator as NDRNG which serves as the entropy source for seeding the DRBG and provides 256 bits of entropy.

© 2017 IBM Corporation / atsec information security.

8. Self-Tests

8.1. Power-Up Tests

The module performs power-up tests automatically when the module is loaded into memory without any operator intervention; power-up tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected.

While the module is performing the power-up tests, no cryptographic service is available and all data output is inhibited. Once the power-up tests are completed successfully, the module enters operational mode and cryptographic services are available. If any of the power-up tests fails, the module throws an exception and enters ERROR state. In ERROR state, all data output is inhibited and no cryptographic operation is allowed. The module needs to be reloaded in order to recover from the ERROR state.

8.1.1. Integrity Tests

The integrity of the module is verified by comparing a HMAC-SHA-1 value calculated at run time with the HMAC value pre-stored in the module.

8.1.2. Cryptographic algorithm tests

The module performs self-tests on all FIPS-Approved cryptographic algorithms supported in the approved mode of operation, using the known answer test (KAT) or Pair-wise Consistency Test (PCT) shown in the following table:

Algorithm	Test
AES	KATs for AES ECB mode; encryption and decryption are tested separately
	KATs for AES GCM mode; encryption and decryption are tested separately
Triple-DES	KATs for Triple-DES ECB mode; encryption and decryption are tested separately
RSA	KATs for RSA signature scheme with 2048 bits modulus size; signature generation and verification are tested separately
	KATs for RSA key transport scheme with 2048 bits modulus size; encryption and decryption are tested separately
DSA	PCT for DSA with (L=2048, N=256)
ECDSA	PCT for ECDSA with NIST P-256 curve
SHS	KATs for SHA-1, SHA-256, SHA-384 and SHA-512
HMAC	KATs for HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512
Hash_DRBG	KATs for Hash_DRBG with SHA-1, SHA-256, SHA-384 and SHA-512
Diffie-Hellman	Primitive “Z” Computation KAT

© 2017 IBM Corporation / atsec information security.

Algorithm	Test
EC Diffie-Hellman	Primitive “Z” Computation KAT
NIST SP800-135 KDF in TLS	Covered by KATs of SHS

Table 7- Self-Tests

8.2. On-Demand self-tests

The on-demand self-tests can be invoked by powering-off the module and reloading it. This performs all the cryptographic algorithm tests as listed in section 8.1.2. During the execution of the on-demand self-tests, no cryptographic service is available and all data output is inhibited. If any of the tests fails, the module will enter ERROR state.

8.3. Conditional Tests

The module performs conditional tests on the cryptographic algorithms shown in the following table. If any of the conditional tests fail, the module throws an exception with an error message and enters ERROR state.

Algorithm	Test
RSA	PCT for RSA key generation
DSA	PCT for DSA key generation
ECDSA	PCT for ECDSA key generation
NDRNG	Continuous Random Number Generator Test

Table 8 - Conditional Tests

© 2017 IBM Corporation / atsec information security.

9. Guidance

9.1. Crypto Officer Guidance

The IBMJCEFIPS provider jar file must be accessible via the Java CLASSPATH and should be installed in the directory $JAVA_HOME/jre/lib/ext as this is a secure location and is also automatically available via the JVM without a CLASSPATH update.

The application will be required to call the IBMJCEFIPS provider (as opposed to another JCE provider) through the normal Java 2 mechanisms such as specifically adding the provider name to the getInstance call as part of the instantiation of a cryptographic object or by placing the IBMJCEFIPS provider higher in the provider list (in java.security) and allowing the JVM to select the first provider that has the requested cryptographic capability.

9.2. User Guidance

This section contains guidance for application programmers to avoid practices that could potentially compromise the secure use of this cryptographic module.

- **Key Zeroization** - the zeroization method should be used to remove the key from memory when a cryptographic key object is no longer needed. While normal Java garbage collection will zeroize the key from memory as part of the object finalizer method, it is a safer coding practice to explicitly call the zeroization method when an application is finished with a key object.
- **Avoid using static objects** – the Java architecture creates objects that are unique to the application, which allows for “single” user access to the cryptographic operations and data, so it is recommended that an application should not create static objects. Static objects are shared in the Java architecture and the creation of a static object would be counter to the unique object method of controlling access and data.
- In case the module’s power is lost and then restored, the AES GCM key shall be re-distributed.

© 2017 IBM Corporation / atsec information security.

10. Mitigation of Other Attacks

The module has been obfuscated using the commercial product KlassMaster. This level of optimized code makes it difficult to decompile and reuse the derived source code. IBM's tests with popular de-compilers (e.g. Jasmine) have shown that de-compiled IBMJCEFIPS code for Java code cannot be compiled and used without extensive alteration. The product KlassMaster is not part of the module and obfuscation is not a service provided by the module, instead it’s a method used for secure delivery of the module.

RSA Blinding has been added to the RSA Signing and RSA encryption function to help mitigate timing attacks.

© 2017 IBM Corporation / atsec information security.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
CBC	Cipher Block Chaining
CFB	Cipher Feedback
CMVP	Cryptographic Module Validation Program
CPACF	CP Assist for Cryptographic Function
CSP	Critical Security Parameter
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
KAS	Key Agreement Schema
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
NDRNG	Non-Deterministic Random Number Generator
OFB	Output Feedback
PSS	Probabilistic Signature Scheme
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard

© 2017 IBM Corporation / atsec information security.

Appendix B. References

FIPS140-2 **FIPS PUB 140-2 - Security Requirements For Cryptographic Modules**
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

FIPS140-2_IG **Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program**
http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf

FIPS180-4 **Secure Hash Standard (SHS)**
http://csrc.nist.gov/publications/fips/fips180-4/fips 180-4.pdf

FIPS186-4 **Digital Signature Standard (DSS)**
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

FIPS197 **Advanced Encryption Standard (AES)**
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

FIPS198-1 **The Keyed Hash Message Authentication Code (HMAC)**
http://csrc.nist.gov/publications/fips/fips198 1/FIPS-198 1_final.pdf

NIST SP800-135 Revision 1 **Recommendation for Existing Application-Specific Key Derivation Functions**
http://csrc.nist.gov/publications/nistpubs/800-135-rev1/sp800-135-rev1.pdf

NIST SP800-38A **Recommendation for Block Cipher Modes of Operation Methods and Techniques**
http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

NIST SP800-38D **Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC**
http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf

NIST SP800-56A Revision 1 **Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**
http://csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf

NIST SP800-56B Revision 1 **Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography**
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br1.pdf

NIST SP800-67 Revision 1 **Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf

NIST SP800-90A **Recommendation for Random Number Generation Using Deterministic Random Bit Generators**
http://csrc.nist.gov/publications/nistpubs/800-90A/SP800-90A.pdf

© 2017 IBM Corporation / atsec information security.