Distech Java Cryptographic Module

FIPS 140-2 Non-Proprietary
Security Policy

Firmware Version 1.0

DISTECH
CONTROLS™

Innovative Solutions for Greener Buildings

First printing – March 2017. Printed in Canada.

Version 1.0.0

TABLE OF CONTENTS

1. INTRODUCTION

An Innovative Leader in Energy Management Solutions, Distech Controls provides unique building management technologies and services that optimize energy efficiency and comfort in buildings, all the while reducing operating costs. We deliver innovative solutions for greener buildings through our passion for innovation, quality, customer satisfaction, and sustainability.

Distech Controls serves multiple market segments through its worldwide business divisions, service offices and a superior network of Authorized System Integrators and Distributors.

1.1. Module Overview

This document is a FIPS 140-2 Security Policy for the Distech Java Cryptographic Module, hereafter referred to as the Module. This policy describes how the Distech Java Cryptographic Module meets the FIPS 140-2 security requirements and how to operate the module in a FIPS compliant manner.

This policy was created as part of the FIPS 140-2, Level 1 validation effort of the module. Federal Information Processing Standards Publication 140-2 ***"Security Requirements for Cryptographic modules (FIPS 140-2)"*** details the United States Federal Government requirements for cryptographic modules. Detailed information about the FIPS 140-2 standard and validation program is available on the NIST website at *http://csrc.nist.gov/groups/STM/cmvp/index.html*.

The security levels supported by the firmware module are as follows:

Table 1: Summary of FIPS security requirements and compliance levels

Section	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	N/A
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

1.2. Modes of Operation and Cryptographic Functionality

The Distech Java Cryptographic Module provides cryptographic functionality to Distech's series of building management appliances. The module is classified under FIPS 140-2 as a firmware based, multi-chip embedded module embodiment. The physical cryptographic boundary is considered to be the area of the PCB within the Distech appliance that includes the RAM, CPU and storage. The logical cryptographic boundary of the module is a pre-compiled Java JAR file which provides the necessary cryptographic functions. The module executes on a non-modifiable purpose built proprietary OS with an underlying JRE. The hardware version on which the module was tested is the **Distech PCB Configuration "A"**.

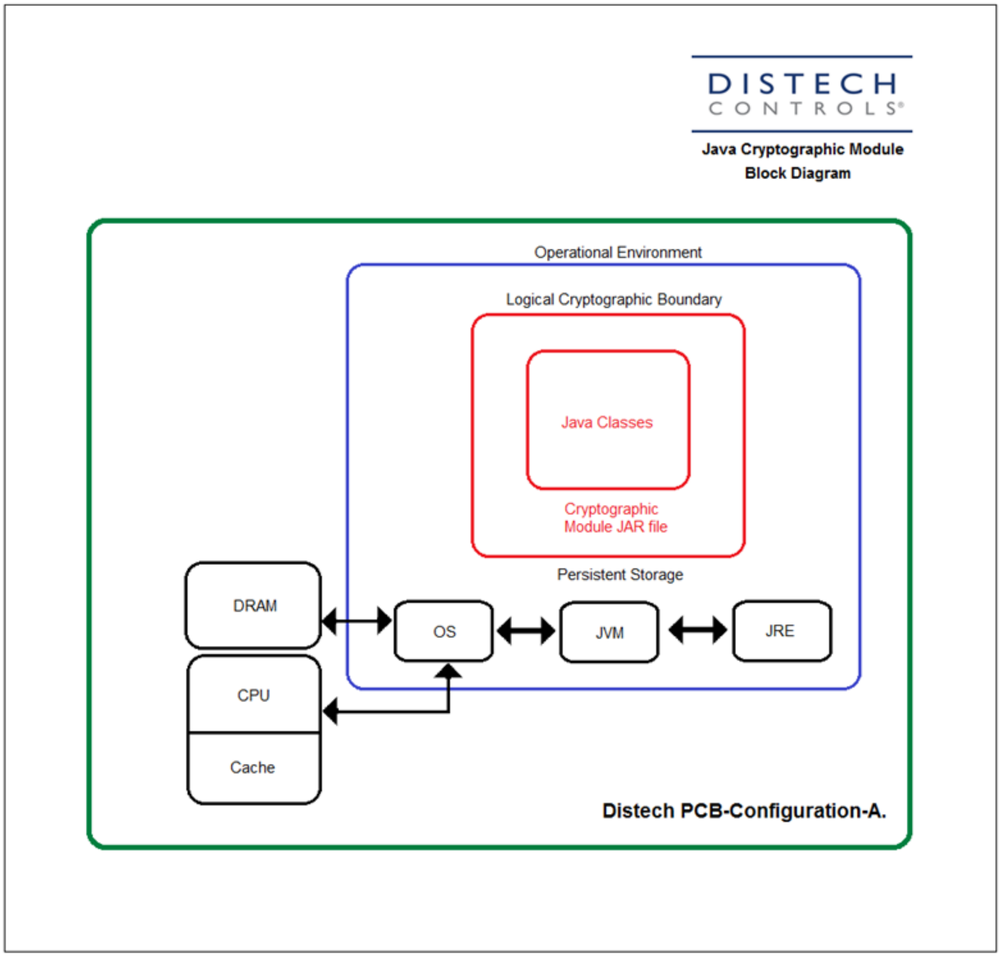

Figure 1: Cryptographic Boundary Block Diagram

The diagram in Figure 1 specifies the logical cryptographic boundary of the module, and how it interfaces with the **Distech PCB Configuration "A"** hardware, which represents the physical boundary. The module supports both FIPS 140-2 Approved and non-Approved modes. There are also security functions which are non-Approved (but allowed) as well as vendor affirmed. Tables 2, 3, 4 and 5 contain the relevant service information.

The cryptographic module is a firmware module, and therefore, control of the physical ports is outside of the module's scope. The module provides a set of logical interfaces which are mapped to the following FIPS 140‐2 defined logical interfaces: data input, data output, control input, status output, and power. When the module performs self‐tests, is in an error state, is generating keys, or performing zeroization, it prevents all output on the logical data output interface as only the thread performing the operation has access to the data. The module is single‐threaded, and in an error state, the module does not return any output data; only an error value. The mapping of the FIPS 140‐2 logical interfaces to the module is described in Table 2.

Table 2: FIPS 140-2 Logical Interfaces

FIPS 140-2 Interface	Module Equivalent
Data Input	API input parameters – plaintext and/or ciphertext data.
Data Output	API output parameters and return values – plaintext and/or ciphertext data.
Control Input	API method calls – method calls, or input parameters, that specify commands and/or control data used to control the operation of the module.
Status Output	API output parameters and return/error codes that provide status information used to indicate the state of the module.
Power	Startup/Shutdown of a process containing the module.

1.2.1 Modes of Operation

The default operation of the module will start with both Approved and non-Approved services enabled. If a FIPS Approved or allowed security function is invoked as per Tables 3, 4 and 5, then the module is operating in the FIPS Approved mode. If any security function listed in Table 6 is invoked, then the module will be operating in a non-FIPS Approved mode. In the event that the module detects that the system property ‘persist.security.fips.enabled’ is set to 1, then the module will start in the FIPS Approved Mode and non-FIPS Approved security functions will be unavailable.

2. APPROVED AND ALLOWED CRYPTOGRAPHIC FUNCTIONS

The Module implements the FIPS Approved and Non-Approved but Allowed cryptographic functions listed in Table 3 to Table 5, below.

Table 3: FIPS Approved Cryptographic Functions

Algorithm	Function	Options	Cert. #
AES	Encryption and Decryption	[FIPS 197, SP 800-38A] Modes: ECB, CBC, OFB, CFB8, CFB128, CTR Key sizes: 128, 192, 256 bits	Cert. #4306
CCM	Generation and Authentication	[SP 800-38C] Key sizes: 128, 192, 256 bits	Cert.#4306
CMAC	Generation and Authentication	[SP 800-38B] Functions: Key sizes: AES with 128, 192, 256 bits and Triple-DES with 2-key[1], 3-key	Cert. #4306 Triple-DES Cert. #2327
GCM[2]	Generation and Authentication	[SP 800-38D] Key sizes: 128, 192, 256 bits	Cert. #4306
DRBG	Hash DRBG, HMAC DRBG, CTR DRBG	[SP 800-90A] Security Strengths: 112, 128, 192, and 256 bits	Cert. #1367
DSA[3]	PQG Generation, PQG Verification, Key Pair Generation, Signature Generation, Signature Verification	[FIPS 186-4] Functions: Key sizes: 1024, 2048, 3072 bits (1024 only for SigVer)	Cert. #1146
ECDSA	Signature Generation Component, Public Key Generation, Signature Generation, Signature Verification, Public Key Validation	[FIPS 186-4] Curves/Key sizes: P-192*, P-224, P-256, P-384, P-521, K-163, K-233, K-283, K-409, K-571, B-163*, B-233, B-283, B-409, B-571 * Curves only used for Signature Verification and Public Key Validation	Cert. #1014 Cert. #1022 (CVL)
HMAC	Generation and Authentication	[FIPS 198-1] SHA sizes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256	Cert. #2842
KAS[4]	Key Agreement	[SP 800-56A-rev2] Parameter sets/Key sizes: FB, FC, EB, EC, ED, EE	Cert. #103

[1] In approved mode of operation, the use of 2-key Triple-DES to generate MACs for anything other than verification purposes is non-compliant.

[2] GCM with an internally generated IV, see section 7.2 concerning external IVs. IV generation is compliant with IG A.5.

[3] DSA signature generation with SHA-1 is only allowed by TLS, SSH and IKE protocols.

[4] Keys are not established directly into the module using the key agreement algorithms.

		(Keys are not established directly into the module using the key agreement algorithms.)	
KDF, Existing Application-Specific[5]	Key Derivation TLS v1.0/1.1 KDF, TLS 1.2 KDF, SSH KDF, X9.63 KDF, IKEv2	[SP 800-135] KDF, SRTP KDF.	Cert. #1020
KBKDF, using Pseudorandom Functions[6]	CMAC-based KBKDF with AES, 2-key Triple-DES, 3-key Triple-DES or HMAC-based KBKDF with SHA-1, SHA-224, SHA-256, SHA-384,SHA-512	[SP 800-108] Counter Mode, Feedback Mode, Double-Pipeline Iteration Mode	Cert. #115
Key Wrapping Using Block Ciphers[7]	Key Wrapping	[SP 800-38F] AES KW, KWP Key sizes: 128, 192, 256 bits (provides between 128 and 256 bits of strength)	Cert. #4306
		[SP 800-38F] Mode: Triple-DES TKW Key size: 3-key (provides 112 bits of strength)	Triple-DES Cert. #2327
RSA	Key Pair Generation, Signature Generation, Signature Verification, Component Test	[FIPS 186-4, FIPS 186-2, ANSI X9.31-1998 and PKCS #1 v2.1 (PSS and PKCS1.5)] Key sizes: 2048, 3072 bits (1024, 1536, 4096 only for SigVer)	RSA Cert. #2327 Cert. #1021 (CVL)
SHS	Digital Signature Generation, Digital Signature Verification, non-Digital Signature Applications	[FIPS 180-4] SHA sizes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256	Cert. #3545
SHA3, SHAKE	Hashing	[FIPS 202] SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256	Cert. #12
Triple-DES	Encryption and Decryption	[SP 800-67] Modes: TECB, TCBC, TCFB64, TCFB8, TOFB, CTR Key sizes: 2-key (Decryption only)[8], 3-key[9]	Triple-DES Cert. #2327

[5] These protocols have not been reviewed or tested by the CAVP and CMVP.

[6] Note: CAVP testing is not provided for use of the Pseudorandom Functions (PRFs) SHA-512/224 and SHA-512/256. These must not be used in the approved mode.

[7] Keys are not established directly into the module using key unwrapping.

[8] 2-key encryption is disabled.

[9] The limit of 2^{32} encryptions with the same Triple-DES key is enforced by IETF protocols within the module. See section 7.4

Table 4: Approved Cryptographic Functions Tested with Vendor Affirmation

Algorithm	Description	IG Reference
AES‐CBC Ciphertext Stealing (CS)	[Addendum to SP 800‐38A, Oct 2010] Modes: CBC‐CS1, CBC‐CS2, CBC‐CS3 Key sizes: 128, 192, 256 bits	Vendor Affirmed IG A.12
KAS[10] using SHA-512/224 or SHA-512/256	[SP 800-56A-rev2] Parameter sets/Key sizes: FB, FC, EB, EC, ED, EE[11]	Vendor Affirmed IG D.1-rev2
KDF, Password-Based	[SP 800-132] Options: PBKDF with Option 1a Functions: HMAC-based KDF using SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	Vendor Affirmed IG D.6
Key Wrapping[10] Using RSA	[SP 800-56B] RSA-KEM-KWS with, and without, key confirmation. Key sizes: 2048, 3072 bits	Vendor Affirmed IG D.4
Key Transport[10] Using RSA	[SP 800-56B] RSA-OAEP with, and without, key confirmation. Key sizes: 2048, 3072 bits	Vendor Affirmed IG D.4
Key Generation	[SP 800-133] Section 5	Vendor Affirmed IG D.12

Table 5: Non-Approved but Allowed Cryptographic Functions

Algorithm	Description
Non-SP 800-56Arev2 compliant Diffie-Hellman	[IG D.8] Diffie-Hellman 2048-bit key agreement primitive for use with system-level key establishment; not used by the module to establish keys within the module. (Key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength)
Non NIST SP800-56B compliant RSA Key Transport	[IG D.9] RSA 2048 or 3072-bit may be used by a calling application as part of a key encapsulation scheme (key wrapping; key establishment methodology provides 112 or 128 bits of encryption strength)
MD5 within TLS	[IG D.2]
HMAC-MD5 within TLS	[IG D.2]
NDRNG	[IG 7.15] non-deterministic random number generator used to seed Approved NIST SP 800-90A DRBG

[10] Keys are not directly established into the module using key agreement or transport techniques.

[11] Note: HMAC SHA-512/224 must not be used with EE.

2.1. Non-Approved Cryptographic Functions

The following cryptographic algorithms and schemes may not be used in an Approved mode of operation. Any use of these schemes and algorithms will cause the module to be operating in a non-Approved mode.

Table 6: Non-Approved Cryptographic Functions

- AES (non-compliant[12])
- ARC4 (RC4)
- Blowfish
- Camellia
- CAST5
- DES
- Diffie-Hellman KAS (non-compliant[13])
- DSA (non-compliant[14])
- DSTU4145
- ECDSA (non-compliant[15])
- ElGamal
- GOST28147
- GOST3410-1994
- GOST3410-2001
- GOST3411
- HMAC-GOST3411
- HMAC-MD5
- HMAC-RIPEMD128
- HMAC-RIPEMD160
- HMAC-RIPEMD256
- HMAC-RIPEMD320
- HMAC-TIGER
- HMAC-WHIRLPOOL
- IDEA
- KBKDF using SHA-512/224 or SHA-512/256 (non-compliant)
- MD5
- OpenSSL PBKDF (non-compliant)
- PKCS#12 PBKDF (non-compliant)
- PKCS#5 Scheme 1 PBKDF (non-compliant)
- PRNG - X9.31
- RC2
- RIPEMD128
- RIPEMD-160
- RIPEMD256
- RIPEMD320
- RSA (non-compliant[16])
- RSA KTS (non-compliant[17])
- SCrypt
- SEED
- Serpent
- SipHash
- SHACAL-2
- TIGER
- Triple-DES (non-compliant[18])
- Twofish
- WHIRLPOOL

[12] Support for additional modes of operation.

[13] Support for additional key sizes and the establishment of keys of less than 112 bits of security strength.

[14] Deterministic signature calculation, support for additional digests, and key sizes.

[15] Deterministic signature calculation, support for additional digests, and key sizes.

[16] Support for additional digests and signature formats, PKCS#1 1.5 key wrapping, support for additional key sizes.

[17] Support for additional key sizes and the establishment of keys of less than 112 bits of security strength.

[18] Support for additional modes of operation.

3. CRITICAL SECURITY PARAMETERS AND PUBLIC KEYS

All CSPs used by the Module are described in Table 7. All usage of these CSPs by the Module (including all CSP lifecycle states) is described in the services detailed in Table 10.

Table 7: Module CSPs

CSP Name	Description
AES Encryption Key[19]	[FIPS-197, SP 800-56C, SP 800-38D, SP 800-38C, Addendum to SP 800-38A] AES (128/192/256) encrypt key19
AES Decryption Key	[FIPS-197, SP 800-56C, SP 800-38D, SP 800-38C, Addendum to SP 800-38A] AES (128/192/256) decrypt key
AES Authentication Key	[FIPS-197] AES (128/192/256) CMAC/GMAC key
AES Wrapping Key	[SP 800-38F] AES (128/192/256) key wrapping key
DH Agreement key	[SP 800-56A-rev2] Diffie-Hellman (>= 2048) private key agreement key
DRBG(CTR AES)	V (128 bits) and AES key (128/192/256), entropy input (length dependent on security strength)
DRBG(CTR Triple-DES)	V (64 bits) and Triple-DES key (192), entropy input (length dependent on security strength)
DRBG(Hash)	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
DRBG(HMAC)	V (160/224/256/384/512 bits) and Key (160/224/256/384/512 bits), entropy input (length dependent on security strength)
DSA Signing Key	[FIPS 186-4] DSA (2048/3072) signature generation key
EC Agreement Key	[SP 800-56A-rev2] EC (All NIST defined B, K, and P curves >= 224 bits) private key agreement key
EC Signing Key	[FIPS 186-4] ECDSA (All NIST defined B, K, and P curves >= 224 bits) signature generation key.
HMAC Authentication Key	[FIPS 198-1] Keyed-Hash key (SHA-1, SHA-2). Key size determined by security strength required (>= 112 bits)
IKEv2 Derivation Function Secret Value	[SP 800-135] Secret value used in construction of key for the specified IKEv2 PRF.
PBKDF Secret Value	[SP 800-132] Secret value used in construction of Keyed-Hash key for the specified PRF.
RSA Signing Key	[FIPS 186-4] RSA (>= 2048) signature generation key
RSA Key Transport Key	[SP 800-56B] RSA (>=2048) key transport (decryption) key
SP 800-56A-rev2 Concatenation Derivation Function	[SP 800-56A-rev2] Secret value used in construction of key for underlying PRF.
SP 800-108 KDF Secret Value	[SP 800-108] Secret value used in construction of key for the specified PRF.
SRTP Derivation Function Secret Value	[SP 800-135] Secret value used in construction of key for the specified SRTP PRF.
SSH Derivation Function Secret Value	[SP 800-135] Secret value used in construction of key for the specified SSH PRF.
TLS KDF Secret Value	[SP 800-135] Secret value used in construction of Keyed-Hash key for the specified TLS PRF.
Triple-DES Authentication Key	[SP 800-67] Triple-DES (128/192) CMAC key
Triple-DES Encryption Key	[SP 800-67] Triple-DES (192) encryption key
Triple-DES Decryption Key	[SP 800-67] Triple-DES (128/192) decryption key

[19] The AES-GCM key and IV is generated randomly per IG A.5, and the Initialization Vector (IV) is a minimum of 96 bits. In the event module power is lost and restored, the consuming application must ensure that any of its AES-GCM keys used for encryption or decryption are re-distributed.

CSP Name	Description
Triple-DES Wrapping Key	[SP 800-38F] Triple-DES (192 bits) key wrapping/unwrapping key, (128 unwrapping only).
X9.63 KDF Secret Value	[SP 800-135] Secret value used in construction of Keyed-Hash key for the specified X9.63 PRF.

Table 8: Module Public Keys

CSP Name	*Description*
DH Agreement Key	[SP 800-56A-rev2] Diffie-Hellman (>= 2048) public key agreement key
DSA Verification Key	[FIPS 186-4] DSA (1024/2048/3072) signature verification key
EC Agreement Key	[SP 800-56A-rev2] EC (All NIST defined B, K, and P curves) public key agreement key
EC Verification Key	[FIPS 186-4] ECDSA (All NIST defined B, K, and P curves) signature verification key
RSA Key Transport Key	[SP 800-56B] RSA (>=2048) key transport (encryption) key.
RSA Verification Key	[FIPS 186-4] RSA (>= 1024) signature verification key

4. ROLES, SERVICES & AUTHENTICATION

4.1. Assumption of Roles

The module supports two distinct operator roles, User and Cryptographic Officer (CO). The cryptographic module implicitly maps the two roles to the services. A user is considered the owner of the thread that instantiates the module and, therefore, only one concurrent user is allowed.

Table 9 lists all operator roles supported by the module. The module does not support a maintenance role and/or bypass capability. The module does not support authentication.

Table 9: Role Description

Role ID	Role Description	Authentication Type
CO	Cryptographic Officer – Powers on and off the module.	N/A – Authentication not required for Level 1
User	User – The user of the complete API.	N/A – Authentication not required for Level 1

4.2. Services

All services implemented by the Module are listed in Table 10 below and Table 11 describes all usage of CSPs by the service. Table 10 lists the services. The second column provides a description of each service and availability to the Cryptographic Officer and User, in columns 3 and 4, respectively.

Table 10: Services

Service	Description	CO	U
Initialize Module and Run Self-Tests on Demand	The JRE will call the static constructor for self-tests on module initialization.	X	
Show Status	A user can call *FipsStatus.IsReady()* at any time to determine if the module is ready. CryptoServicesRegistrar.*IsInApprovedOnlyMode()* can be called to determine the FIPS mode of operation.		X
Zeroize / Power-off	The module uses the JVM garbage collector on thread termination.		X
Data Encryption	Used to encrypt data.		X
Data Decryption	Used to decrypt data.		X
MAC Calculation	Used to calculate data integrity codes with CMAC.		X
Signature Authentication	Used to generate signatures (DSA, ECDSA, RSA).		X
Signature Verification	Used to verify digital signatures.		X
DRBG (SP800-90A) output	Used for random number, IV and key generation.		X
Message Hashing	Used to generate a SHA-1, SHA-2, or SHA-3 message digest, SHAKE output.		X
Keyed Message Hashing	Used to calculate data integrity codes with HMAC.		X
TLS Key Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a master secret in TLS from a pre-master secret and additional input.		X
SP 800-108 KDF	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X
SSH Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X
X9.63 Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X
SP 800-56A-rev2 Concatenation Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X

Service	Description	CO	U
IKEv2 Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X
SRTP Derivation Function	(secret input) (outputs secret) Used to calculate a value suitable to be used for a secret key from an input secret and additional input.		X
PBKDF	(secret input) (outputs secret) Used to generate a key using an encoding of a password and an additional function such as a message hash.		X
Key Agreement Schemes	Used to calculate key agreement values (SP 800-56A, Diffie-Hellman).		X
Key Wrapping	Used to encrypt a key value. (RSA, AES, Triple-DES)		X
Key Unwrapping	Used to decrypt a key value. (RSA, AES, Triple-DES)		X
NDRNG Callback	Gathers entropy in a passive manner from a user-provided function		X
Utility	Miscellaneous utility functions, does not access CSPs		X

Note: The module services are the same in the approved and non-approved modes of operation. The only difference is the function(s) used (approved/allowed or non-approved/non-allowed).

Services in the module are accessed via the public APIs of the Jar file. The ability of a thread to invoke non-approved services depends on whether it has been registered with the module as approved mode only. In Approved only mode, there are only Approved services accessible. In the presence of a Java SecurityManager, Approved mode services specific to a context, such as DSA and ECDSA for use in TLS, require specific permissions to be configured in the JVM configuration by the Cryptographic Officer or User. In the absence of a Java SecurityManager specific services related to protocols such as TLS are available, however must only be used in relation to those protocols.

Table 11 defines the relationship between access to CSPs and the different module services. The modes of access shown in the table are defined as:

G = **Generate**: The module generates the CSP.

R = **Read**: The module reads the CSP. The read access is typically performed before the module uses the CSP.

E = **Execute**: The module executes using the CSP.

W = **Write**: The module writes the CSP. The write access is typically performed after a CSP is imported into the module, when the module generates a CSP, or when the module overwrites an existing CSP.

Z = **Zeroize**: The module zeroizes the CSP.

J = **JVM:** Zeroized as part of JVM garbage collection.

Table 11: CSP Access Rights within Services

Service	CSPs									
	AES Keys	DH Keys	DRBG Keys	DSA Keys	EC Agreement Key	ECDSA Key	HMAC Keys	KDF Secret Values	RSA Keys	Triple-DES Keys
Initialize Module and Run Self-Tests on Demand										
Show Status										
Zeroize / Power-off	Z	Z	Z	Z	Z	Z	Z	J	Z	Z
Data Encryption	R									R
Data Decryption	R									R
MAC Calculation	R						R			R
Signature Authentication				R		R			R	
Signature Verification				R		R			R	
DRBG (SP800-90A) output	G	G	G.R	G	G	G	G		G	G
Message Hashing										
Keyed Message Hashing							R			
TLS Key Derivation Function								R		
SP 800-108 KDF								R		
SSH Derivation Function								R		
X9.63 Derivation Function								R		
SP 800-56A-rev2 Concatenation Derivation Function								R		
IKEv2 Derivation Function								R		
SRTP Derivation Function								R		
PBKDF							G,R			
Key Agreement Schemes	G	R			R		R		R	G
Key Wrapping/Transport (RSA,AES, Triple-DES)	R						R		R	R
Key Unwrapping (RSA, AES, Triple-DES)	R						R		R	R
NDRNG Callback			G							
Utility										

5. SELF-TESTS

Each time the module is powered up, it tests that the cryptographic algorithms still operate correctly and that sensitive data have not been damaged. Power-up self–tests are available on demand by power cycling the module. Power-up self-tests and health tests are performed regardless of whether the module is in the Approved mode or not.

On power-up or reset, the module performs the self-tests that are described in Table 12 below. All KATs must be completed successfully prior to any other use of cryptography by the module. If one of the KATs fails, the module enters the self-test failure error state. The module will output a detailed error message when FipsStatus.isReady() is called. The error state can only be cleared by reloading the module and calling FipsStatus.isReady() again to confirm successful completion of the KATs.

Table 12: Power-Up Self-Tests

Algorithm	Description
Software Integrity	HMAC-SHA256
AES	KATs: Encryption, Decryption Modes: ECB Key sizes: 128 bits
CCM	KATs: Generation, Verification Key sizes: 128 bits
AES-CMAC	KATs: Generation, Verification Key sizes: AES with 128 bits
FFC KAS	KATs: Per IG 9.6 – Primitive "Z" Computation Parameter Sets/Key sizes: FB
DRBG	KATs: HASH_DRBG, HMAC_DRBG, CTR_DRBG Security Strengths: 256 bits
DSA	PWCT: Signature Generation, Signature Verification Key sizes: 2048 bits
ECDSA	PWCT: Signature Generation, Signature Verification Curves/Key sizes: P-256
GCM/GMAC	KATs: Generation, Verification Key sizes: 128 bits
HMAC	KATs: Generation, Verification SHA sizes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256
ECC KAS	KATs: Per IG 9.6 – Primitive "Z" Computation Parameter Sets/Key sizes: FB
RSA	KATs: Signature Generation, Signature Verification Key sizes: 2048 bits
SHS	KATs: Output Verification SHA sizes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512
Triple-DES	KATs: Encryption, Decryption Modes: TECB, Key sizes: 3-Key
Triple-DES-CMAC	KATs: Generation, Verification Key sizes: 3-Key
Extendable-Output functions (XOF)	KATs: Output Verification XOFs:SHAKE128, SHAKE256
Key Agreement Using RSA	KATs: SP 800-56B specific KATs per IG D.4 Key sizes: 2048 bits
Key Transport Using RSA	KATs: SP 800-56B specific KATs per IG D.4 Key sizes: 2048 bits

Table 13: Conditional Self-Tests

Algorithm	Description
NDRNG	NDRNG Continuous Test performed when a random value is requested from the NDRNG.
DH	DH Pairwise Consistency Test performed on every DH key pair generation.
DRBG	DRBG Continuous Test performed when a random value is requested from the DRBG.
DSA	DSA Pairwise Consistency Test performed on every DSA key pair generation.
ECDSA	ECDSA Pairwise Consistency Test performed on every EC key pair generation.
RSA	RSA Pairwise Consistency Test performed on every RSA key pair generation.
DRBG Health Checks	Performed conditionally on DRBG, per SP 800-90A Section 11.3. Required per IG C.1.
SP 800-56A Assurances	Performed conditionally per SP 800-56A Sections 5.5.2, 5.6.2, and/or 5.6.3. Required per IG 9.6.

5.1. Use of External NDRNG

The module makes use of the JVM's configured SecureRandom entropy source to provide entropy when required. The module will request entropy as appropriate to the security strength and seeding configuration for the DRBG that is using it. In the Approved mode, the minimum amount of entropy that would be requested is 112 bits with a larger minimum being set if the security strength of the operation requires it. The module will wait until the SecureRandom.generateSeed() returns the requested amount of entropy; blocking if necessary.

6. MITIGATION OF OTHER ATTACKS POLICY

The Module implements basic protections to mitigate against timing based attacks against its internal implementations. There are two counter-measures used. The first is Constant Time Comparisons, which protect the digest and integrity algorithms by strictly avoiding “fast fail” comparison of MACs, signatures, and digests so the time taken to compare a MAC, signature, or digest is constant regardless of whether the comparison passes or fails. The second is made up of Numeric Blinding and decryption/signing verification which both protect the RSA algorithm.

Numeric Blinding prevents timing attacks against RSA decryption and signing by providing a random input into the operation which is subsequently eliminated when the result is produced. The random input makes it impossible for a third party observing the private key operation to attempt a timing attack on the operation as they do not have knowledge of the random input and consequently the time taken for the operation tells them nothing about the private value of the RSA key.

Decryption/signing verification is carried out by calculating a primitive encryption or signature verification operation after a corresponding decryption or signing operation before the result of the decryption or signing operation is returned. The purpose of this is to protect against Lenstra's CRT attack by verifying the correctness the private key calculations involved. Lenstra's CRT attack takes advantage of undetected errors in the use of RSA private keys with CRT values and, if exploitable, can be used to discover the private value of the RSA key.

7. SECURITY RULES AND GUIDANCE

7.1. Basic Enforcement

The module design corresponds to the Module security rules. This section documents the security rules enforced by the cryptographic module to implement the security requirements of this FIPS 140-2 Level 1 module.

1. The module provides two distinct operator roles: User and Cryptographic Officer.
2. The module does not provide authentication.
3. The module may be commanded to perform the power up self-tests by cycling power or resetting the module.
4. Power up self-tests do not require any operator action.
5. Data output is inhibited during key generation, self-tests, zeroization, and error states.
6. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the module.
7. There are no restrictions on which keys or CSPs are zeroized by the zeroization service.
8. The module does not support concurrent operators.
9. The module does not have any external input/output devices used for entry/output of data.
10. The module does not enter or output plaintext CSPs from the module’s physical boundary.
11. The module does not output intermediate key values.

7.2. Enforcement and Guidance for GCM IVs

IVs for GCM can be generated randomly; where an IV is not generated randomly the module supports the importing of GCM IVs. In approved mode, when a GCM IV is generated randomly, the module enforces the use of an approved DRGB in line with Section 8.2.2 of SP 800-38D. The minimum length of an IV that can be used with AES-GCM is 96 bits.

Please note that importing an IV externally is a non-compliant use of GCM as per FIPS 140-2, IG A.5. The module will be operating in a non-Approved mode of operation under this condition.

Per IG A.5, in the event module power is lost and restored, the consuming application must ensure that any of its AES-GCM keys used for encryption or decryption are redistributed.

7.3. Enforcement and Guidance for Use of the Approved PBKDF

In line with the requirements for SP 800-132, keys generated using the approved PBKDF must only be used for storage applications. Any other use of the approved PBKDF is non-conformant. In approved mode the module enforces that any password used must encode to at least 14 bytes (112 bits) and that the salt is at least 16 bytes (128 bits) long. The iteration count associated with the PBKDF should be as large as practical.

7.4. Guidance for Use of Triple-DES as Per SP 800-67rev1

The limit of 2^{32} encryptions with the same Triple-DES key is enforced by the IETF protocols within the module as follows:

- **RFC 5246 "The Transport Layer Security (TLS) Protocol Version 1.2".**
- **RFC 3370 "Cryptographic Message Syntax (CMS) Algorithms"**
- **RFC 4880 "OpenPGP Message Format"**
- **RFC 5751 "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification"**
- **RFC 5208 "Public-Key Cryptography Standards (PKCS) #8: Private-Key Information Syntax Specification Version 1.2"**
- **RFC 7292 "PKCS #12: Personal Information Exchange Syntax v1.1"**

8. ABBREVIATIONS & ACRONYMS

Table 14: Acronyms and Definitions

Abbreviation	Full Specification Name
ANSI X9.31	X9.31-1998, Digital Signatures using Reversible Public Key Cryptography for the Financial Services Industry (rDSA), September 9, 1998
FIPS 140-2	Security Requirements for Cryptographic modules, May 25, 2001
FIPS 180-4	Secure Hash Standard (SHS)
FIPS 186-4	Digital Signature Standard (DSS)
FIPS 197	Advanced Encryption Standard
FIPS 198-1	The Keyed-Hash Message Authentication Code (HMAC)
FIPS 202	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions
IG	Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program
PKCS#5	Password-Based Cryptography Standard
PKCS#12	Personal Information Exchange Syntax Standard
SP 800-38A	Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode
SP 800-38B	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
SP 800-38C	Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality
SP 800-38D	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC
SP 800-38F	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping
SP 800-56A	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography
SP 800-56B	Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography
SP 800-56C	Recommendation for Key Derivation through Extraction-then-Expansion
SP 800-67	Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher
SP 800-90A	Recommendation for Random Number Generation Using Deterministic Random Bit Generators
SP 800-108	Recommendation for Key Derivation Using Pseudorandom Functions
SP 800-132	Recommendation for Password-Based Key Derivation
SP 800-135	Recommendation for Existing Application–Specific Key Derivation Functions
AES	Advanced Encryption Standard
API	Application Programming Interface
BC	Bouncy Castle
CBC	Cipher-Block Chaining
CCM	Counter with CBC-MAC
CFB	Cipher Feedback Mode
CMAC	Cipher-based Message Authentication Code
CMVP	Crypto Module Validation Program
CO	Cryptographic Officer
CPU	Central Processing Unit
CS	Ciphertext Stealing
CSP	Critical Security Parameter
CTR	Counter-mode
CVL	Component Validation List
DES	Data Encryption Standard
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Authority
DSTU4145	Ukrainian DSTU-4145-2002 Elliptic Curve Scheme
EC	Elliptic Curve
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Authority
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FIPS	Federal Information Processing Standards
GCM	Galois/Counter Mode
GMAC	Galois Message Authentication Code
GOST	Gosudarstvennyi Standard Soyuza SSR/Government Standard of the Union of Soviet Socialist Republics
HMAC	key-Hashed Message Authentication Code

Abbreviation	Full Specification Name
IG	See References
JAR	Java ARchive
JRE	Java Runtime Environment
JVM	Java Virtual Machine
IV	Initialization Vector
KAS	Key Agreement Scheme
KAT	Known Answer Test
KDF	Key Derivation Function
KW	Key Wrap
KWP	Key Wrap with Padding
MAC	Message Authentication Code
MD5	Message Digest algorithm MD5
N/A	Non-Applicable
NDRNG	Non-Deterministic Random Number Generator
OFB	Output Feedback
OS	Operating System
PBKDF	Password-Based Key Derivation Function
PKCS	Public Key Cryptography Standards
PQG	Diffie-Hellman Parameters P, Q and G
RC	Rivest Cipher, Ron’s Code
RIPEMD	RACE Integrity Primitives Evaluation Message Digest
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
TCBC	TDEA Cipher-Block Chaining
TCFB	TDEA Cipher Feedback Mode
TDEA	Triple Data Encryption Algorithm
TECB	TDEA Electronic Codebook
TOFB	TDEA Output Feedback
TLS	Transport Layer Security
XOF	Extendable-Output Function