Distech SSL 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 SSL Cryptographic Module (Firmware Version: 1.0), hereafter referred to as the Module. This policy describes how the Distech SSL 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 Distech SSL 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 object file which provides the necessary cryptographic functions. The module executes on a non-modifiable purpose built proprietary OS (Distech OS V4.4.4 (kernel V3.2)). The hardware version on which the module was tested is the **Distech PCB-Configuration "A".** The module was tested both with and without PAA (NEON) support.

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	N/A
Overall Level	1

2. Modes of Operation and Cryptographic Functionality

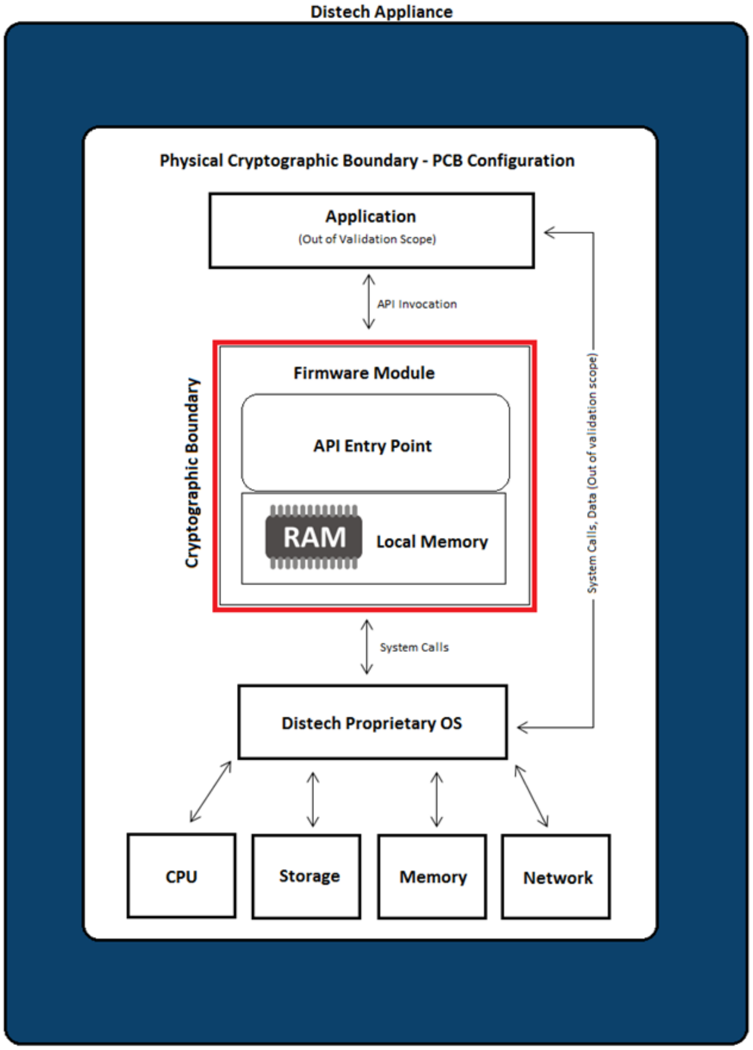

Figure 1: Block Diagram

The module supports both FIPS 140-2 Approved and non-Approved modes. There are also security functions which are non-Approved (but allowed). Tables 2, 3 and 4 list these categories respectively.

3. APPROVED AND ALLOWED CRYPTOGRAPHIC FUNCTIONS

Table 2: FIPS Approved Cryptographic Functions

Algorithm	Function	Options	Cert. #
AES [FIPS 197] AES [SP 80038B] CMAC [SP 80038C] CCM [SP 80038D] GCM [SP 80038E] XTS[1]	Encryption, Decryption and CMAC	ECB Mode: Encrypt/Decrypt Key Size: 128, 192, 256. CBC Mode: Encrypt/Decrypt Key Size: 128, 192, 256. OFB Mode: Encrypt/Decrypt Key Size: 128, 192, 256. CFB1 Mode: Encrypt/Decrypt Key Size: 128, 192, 256. CFB8 Mode: Encrypt/Decrypt Key Size: 128, 192, 256. CFB128 Mode:Encrypt/Decrypt Key Size: 128, 192, 256. CTR Mode: Encrypt only Key Size: 128 192 256 CMAC Generation using AES (128, 192, 256) CMAC Verification using AES 128, 192, 256) CCM using (128, 192, 256) AES GCM Mode(s) tested: Encrypt Decrypt Keysize(s) tested: 128 192 256 XTS KeySize Tested: XTS-AES128 and XTS-AES256 State(s) Tested: Encrypt/Decrypt Data Unit Lengths Tested: 128, 256, 136, 200 and 2^16	Cert. #4238
CVL	Key Agreement	SP 800-56A ECC CDH Primitive (Section 5.7.1.2) Component Curves tested: P-224 P-256 P-384 P-521 K-233 K-283 K-409 K-571 B-233 B-283 B-409 B-571	Cert. #985
DRBG (NIST SP 800-90A)	Random Number Generation Symmetric Key Generation	Hash_DRBG (SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512) HMAC_DRBG (SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512) CTR DRBG (AES-128, AES-192 and AES-256)	Cert. #1318

DSA	Digital Signature Operations	PQG Generation:
		L= 2048, 3072
		N= 224, 256
		SHA = 224, 256, 384 and 512
		
		PQG Verification:
		L= 1024, 2048, 3072
		N= 224, 256
		SHA = 224, 256, 384 and 512
		
		Key Pair:
		L= 2048, 3072
		N= 224, 256
		
		Signature Generation:
		L= 2048, 3072
		N= 224, 256
		SHA = 224, 256, 384 and 512
			Cert.#1131

[1] As per NIST SP 800-38E, the AES-XTS mode was designed for storage applications only and not for the encryption of data in transit. The AES-XTS implementation in this module shall only be used for storage applications.

		Signature Verification:
		L= 1024, 2048, 3072
		N= 160, 224, 256
		SHA = 1, 224, 256, 384 and 512	

ECDSA	Elliptic Curve Digital Signature Operations (The Module supports only NIST defined curves for use with ECDSA and ECDH.)	Key Pair Generation Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Public Key Validation Curves: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 Signature Generation Curve/SHA pairs tested: P = 224, 256, 384 and 521 /w SHA-224, 256, 384 and 512. K = 233, 283, 409 and 571 /w SHA-224, 256, 384 and 512. B = 233, 283, 409 and 571 /w SHA-224, 256, 384 and 512. Signature Verification Curve/SHA pairs tested: P = 192, 224, 256, 384 and 521 /w SHA-1, 224, 256, 384 and 512. K = 163, 233, 283, 409 and 571 /w SHA-1, 224, 256, 384 and 512. B = 163, 233, 283, 409 and 571 /w SHA-1, 224, 256, 384 and 512.	Cert. #980
HMAC	Keyed Hashing Operations	HMAC SHA1: KeySizes tested: KS < BS KS = BS KS > BS MAC sizes tested: 10 12 16 20 HMAC SHA224: KeySizes tested: KS < BS KS = BS KS > BS MAC sizes tested: 14 16 20 24 28 HMAC SHA256: KeySizes tested: KS < BS KS = BS KS > BS MAC sizes tested: 16 24 32 HMAC SHA384: KeySizes tested: KS < BS KS = BS KS > BS MAC sizes tested: 24 32 40 48 HMAC SHA512: KeySizes tested: KS < BS KS = BS KS > BS MAC sizes tested: 32 40 48 56 64	Cert. #2777
RSA	RSA Digital Signature Operations	FIPS 186-2 Signature Generation 9.31: Modulus lengths: 4096 SHAs: SHA-256, SHA-384, SHA-512 Signature Generation PKCS1.5: Modulus lengths: 4096 SHAs: SHA-224, SHA-256, SHA-384, SHA-512 Signature Generation PSS:	Cert. #2287

		Modulus lengths: 4096 SHAs: SHA-224, SHA-256, SHA-384, SHA-512 Signature Verification 9.31: Modulus lengths: 1024, 1536, 2048, 3072, 4096 SHAs: SHA-1, SHA-256, SHA-384, SHA-512 Signature Verification PKCS1.5 Modulus lengths: 1024, 1536, 2048, 3072, 4096 SHAs: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 Signature Verification PSS: Modulus lengths: 1024, 1536, 2048, 3072, 4096 SHAs: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 FIPS 186-4 Signature Generation 9.31: Mod 2048 SHA: SHA-256, SHA-384, SHA-512 Mod 3072 SHA: SHA-256, SHA-384, SHA-512 Signature Generation PKCS1.5: Mod 2048 SHA: SHA-224, SHA-256, SHA-384, SHA-512 Mod 3072 SHA: SHA-224, SHA-256, SHA-384, SHA-512 Signature Generation PSS: Mod 2048: SHA-224: Salt Length: 0 SHA-256: Salt Length: 0 SHA-384: Salt Length: 0 SHA-512: Salt Length: 0 Mod 3072: SHA-224: Salt Length: 0 SHA-256: Salt Length: 0 SHA-384: Salt Length: 0 SHA-512: Salt Length: 0	
SHS	Hashing	SHA-1 Byte only SHA-224 Byte only SHA-256 Byte only SHA-384 Byte only SHA-512 Byte only	Cert. #3476
Triple-DES[2]	Encryption, Decryption and CMAC	CBC, CFB1, CFB8, CFB64, OFB and ECB Modes: Encrypt/Decrypt Key Option = 1 (K1, K2, K3 independent) CMAC Verification using TDES (3-Key)	Cert. #2295

[2] As per the SP 800-67rev1 Transition specified in the CMVP Implementation Guidance, please be advised that this module shall not be used to perform more than 2^{32} Triple-DES encryption operations using the same Triple-DES key.

Table 3: Allowed Cryptographic Functions

Category	Algorithm	Description
Key Encryption, Decryption	RSA	The RSA algorithm is used by the calling application for encryption or decryption of keys. No claim is made for SP 800-56B compliance, and no CSPs are established into or exported out of the module using these services. If the implemented RSA is used in a key transport scheme, please be advised that the supported key strengths range from 1024 to 16384 bits. You must ensure that only keys between 2048 and 16384 bits (providing 112 to 270 bits of encryption strength) are used for this purpose. Failure to use this range of keys will result in a non-compliant module.

4. NON-APPROVED CRYPTOGRAPHIC FUNCTIONS

The following cryptographic algorithms and schemes shall 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. Keys and secret critical security parameters defined in the approved mode of operation, shall not be accessed or shared while in a non-approved mode of operation. Furthermore, critical security parameters shall not be generated while in a non-approved mode. The approved DRBG may be used in a non-approved mode. However, the approved DRBGs seed or seed key shall not be accessed or shared in the non-approved mode.

Table 4: Non-Approved Cryptographic Functions

Non-Approved Random Number Generation	ANSI X9.31 RNG (non-compliant)
Non-Approved RSA Functions	186-2 RSA Key Generation – Use of 1024 bit keys (non-compliant) 186-2 RSA - Use of SHA-1 for Digital Signature Generation (non-compliant)
Non-Approved DSA Functions	186-2 DSA Key Generation – Use of 1024 bit keys (non-compliant) 186-2 DSA - Use of SHA-1 for Digital Signature Generation (non-compliant) 186-4 DSA Key Generation – Use of 1024 bit keys (non-compliant) 186-4 DSA - Use of SHA-1 for Digital Signature Generation (non-compliant)
Non-Approved ECDSA Functions	186-2 ECDSA – Use of curves PKG: CURVES(P-192 K-163 B-163) SIG(gen): CURVES(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) 186-4 ECDSA – Use of curves PKG: CURVES(P-192 K-163 B-163) SigGen: CURVES(P-192: (SHA-1, 224, 256, 384, 512) P-224:(SHA-1) P-256:(SHA-1)P-384: (SHA-1) P-521:(SHA-1) K-163: (SHA-1, 224, 256, 384, 512) K-233:(SHA-1) K-283:(SHA-1) K-409:(SHA-1) K-571:(SHA-1) B-163: (SHA-1, 224, 256, 384, 512) B-233:(SHA-1) B-283: (SHA-1) B-409:(SHA-1) B-571:(SHA-1))
Non-Approved EC Diffie-Hellman Functions	[SP 800-56A] (5.7.1.2) - All NIST Recommended B, K and P curves sizes 163 and 192

5. CRITICAL SECURITY PARAMETERS AND PUBLIC KEYS

All CSPs used by the module are described below. The CSP names are generic, corresponding to API parameter data structures.

Table 5: Module CSPs

CSP Name	Description
RSA SGK	RSA (1024 to 16384 bits) signature generation key
RSA KDK	RSA (1024 to 16384 bits) key decryption (private key transport) key
DSA SGK	[FIPS 186-4] DSA (1024/2048/3072) signature generation key or [FIPS 186-2] DSA (1024) signature generation key
ECDSA SGK	ECDSA (All NIST defined B, K, and P curves except sizes 163 and 192) signature generation key
EC DH Private	EC DH (All NIST defined B, K, and P curves except sizes 163 and 192) private key agreement key.
AES EDK	AES (128/192/256) encrypt / decrypt key
AES CMAC	AES (128/192/256) CMAC generate / verify key
AES GCM	AES (128/192/256) encrypt / decrypt / generate / verify key
AES XTS	AES (256/512) XTS encrypt / decrypt key
TDES EDK	TDES (3-Key) encrypt / decrypt key
TDES CMAC	TDES (3-Key) CMAC generate / verify key
HMAC Key	Keyed hash key (160/224/256/384/512)
Hash_DRBG CSPs	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
HMAC_DRBG CSPs	V (160/224/256/384/512 bits) and Key (160/224/256/384/512 bits), entropy input (length dependent on security strength)
CTR_DRBG CSPs	V (128 bits) and Key (AES 128/192/256), entropy input (length dependent on security strength)

Table 6: Module Public Keys

CSP Name	Description
RSA SVK	RSA (1024 to 16384 bits) signature verification public key
RSA KEK	RSA (1024 to 16384 bits) key encryption (public key transport) key
DSA SVK	[FIPS 186-4] DSA (1024/2048/3072) signature verification key or [FIPS 186-2] DSA (1024) signature verification key
ECDSA SVK	ECDSA (All NIST defined B, K and P curves) signature verification key
EC DH Public	EC DH (All NIST defined B, K and P curves) public key agreement key.

6. KEY MANAGEMENT

6.1. Key/CSP Storage

The Module stores DRBG state values for the lifetime of the DRBG instance. The module uses CSPs passed in by the calling application on the stack. The Module does not store any CSP persistently (beyond the lifetime of an API call), with the exception of DRBG state values used for the Module's default key generation service.

6.2. Key/CSP Generation

The Module implements NIST SP 800-90A compliant DRBG services for the creation of symmetric keys, and for generation of DSA, elliptic curve, and RSA keys. The calling application is responsible for storage of generated keys returned by the module.

6.3. Key/CSP Entry

All CSPs enter the Module's logical boundary in plaintext as API parameters, associated by memory location. However, none cross the physical boundary.

6.4. Key/CSP Output

The Module does not output CSPs, other than as explicit results of key generation services. However, none cross the physical boundary.

6.5. Key/CSP Destruction

Zeroization of sensitive data is performed automatically by API function calls for temporarily stored CSPs. In addition, the module provides functions to explicitly destroy CSPs related to random number generation services. The calling application is responsible for parameters passed in and out of the module.

Private and secret keys are provided to the module by the calling application, and are destroyed when released by the appropriate API function calls. Keys residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the Module defined API. The operating system protects memory and process space from unauthorized access. Only the calling application that creates or imports keys can use or export such keys. All API functions are executed by the invoking calling application in a non-overlapping sequence such that no two API functions will execute concurrently. An authorized application as user (Crypto-Officer and User) has access to all key data generated during the operation of the module.

The AES‐GCM key and IV is generated as per IG A.5, and the Initialization Vector (IV) is a minimum of 96 bits. In the event that module power is lost and restored, the calling application shall ensure that any AES-GCM keys used for encryption or decryption are redistributed.

7. INSTRUCTIONS FOR OPERATING IN THE APPROVED MODE

The Distech SSL Cryptographic Module is a firmware module which comes pre-installed and distributed as part of Distech's line of hardware products. Tables 2 and 3 in this document, serve as the benchmark for cryptographic algorithms and schemes which allow the module to operate in the FIPS 140-2 compliant mode of operation. Since the module does not support distinct, enforced, Approved and non-Approved modes of operation, it is implied that when the algorithms in Tables 2 and 3 are used that the module is automatically operating in the Approved mode. For every security function that is executed from Table 4, the module will be automatically operating in the non-Approved mode during the time such functions are active. The calling Distech application is responsible for invoking the module using an API call, which returns a "1" for success and "0" for failure. If the initialization process fails for any reason, then all cryptographic services fail from this point on. The specifics of the error are translated by the calling application.

8. PORTS AND INTERFACES

The physical ports of the module are the same as the hardware on which it is executing. The logical interface is a C language application program interface (API).

Table 7: Mapping for Logical Interfaces to FIPS 140-2

Logical interface type	Description
Control Input	API entry point and corresponding stack parameters
Data Input	API entry point data input stack parameters
Data Output	API entry point data output stack parameters
Status Output	API entry point return values and status stack parameters

As a firmware module, control of the physical ports is outside the scope of the module. However, when the module is performing self-tests, or is in an error state, all output on the logical data output interface is inhibited. The module is single-threaded and when in the error state, returns only an error value. (No data output is returned).

9. Roles Services and Authentication

The Module implements the required User and Crypto-Officer roles; however, authentication for those roles is not supported by the Distech SSL Cryptographic Module. Only one role may be active at a time, as the module does not allow concurrent operators. The User and Crypto-Officer roles are assumed implicitly.

The underlying, proprietary operating system segregates operator processes into separate process spaces. Each process space is logically separated from all other processes by the operating system software and hardware. The module functions entirely within the process space of the calling application, and implicitly satisfies the FIPS 140-2 requirement for a single user mode of operation.

Both roles have access to all of the services provided by the module.

- User Role (User): Loading the Module and calling any of the API functions.
- Crypto Officer Role (CO): Installation of the Module within the non-modifiable OE and calling of any API functions

Table 8: Services & CSP Access

Service	Role	Description
Initialize	User, CO	Module initialization. Does not access CSPs.
Self-test	User, CO	Perform self-tests (FIPS_selftest). Does not access CSPs.
Show status	User, CO	Functions that provide module status information: - Version (as unsigned long or const char *) - FIPS Mode (Boolean) Does not access CSPs.
Zeroize	User, CO	Functions that destroy CSPs: - fips_drbg_uninstantiate: for a given DRBG context, overwrites DRBG CSPs (Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs) All other services automatically overwrite CSPs stored in allocated memory.
Random number generation	User, CO	Used for random number and symmetric key generation. - Seed or reseed a DRBG instance - Determine security strength of a DRBG instance - Obtain random data Uses and updates Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs.
Asymmetric key generation	User, CO	Used to generate DSA, ECDSA and RSA keys: RSA SGK, RSA SVK; DSA SGK, DSA SVK; ECDSA SGK, ECDSA SVK There is one supported entropy strength for each mechanism and algorithm type, the maximum specified in SP800-90
Symmetric encrypt/decrypt	User, CO	Used to encrypt or decrypt data. Executes using AES EDK, AES GCM, AES XTS, TDES EDK (passed in by the calling process).
Symmetric digest	User, CO	Used to generate or verify data integrity with CMAC. Executes using AES CMAC, TDES, CMAC (passed in by the calling process).
Message digest	User, CO	Used to generate a SHA-1 or SHA-2 message digest. Does not access CSPs.
Keyed Hash	User, CO	Used to generate or verify data integrity with HMAC. Executes using HMAC Key (passed in by the calling process).
Key transport	User, CO	Used to encrypt or decrypt a key value on behalf of the calling process (does not establish keys into the module). Executes using RSA KDK, RSA KEK (passed in by the calling process).
Key agreement	User, CO	Used to perform key agreement primitives on behalf of the calling process (does not establish keys into the module). Executes using EC DH Private, EC DH Public (passed in by the calling process).
Digital signature	User, CO	Used to generate or verify RSA, DSA or ECDSA digital signatures. Executes using RSA SGK, RSA SVK; DSA SGK, DSA SVK; ECDSA SGK, ECDSA SVK (passed in by the calling process).
Utility	User, CO	Miscellaneous helper functions. Does not access CSPs.

10. PHYSICAL SECURITY

The module maintains physical security by using production grade components and standard passivation, as allowed by FIPS 140-2 level 1.

11. MODULE SELF-TESTS

The module performs the applicable power-up self-tests listed below, when initialized (or on-demand):

Table 9: Module Power-Up Self-Tests

Algorithm/Scheme	Type	Description
Firmware Integrity Test	Known Answer Test	HMAC-SHA1
HMAC	Known Answer Test	One KAT per SHA1, SHA224, SHA256, SHA384 and SHA512 (Per IG 9.3, this testing covers SHA POST requirements.)
AES	Known Answer Test	Separate encrypt and decrypt, ECB mode, 128 bit key length
AES CCM	Known Answer Test	Separate encrypt and decrypt, 192 key length
AES GCM	Known Answer Test	Separate encrypt and decrypt, 256 key length
XTS-AES	Known Answer Test	128, 256 bit key sizes to support either the 256-bit key size (for XTSAES128) Or the 512bit key size (for XTSAES256)
AES-CMAC	Known Answer Test	Sign and verify CBC mode, 128, 192, 256 key lengths
Triple-DES	Known Answer Test	3-Key Triple-DES with separate encrypt and decrypt, ECB mode.
Triple-DES-CMAC	Known Answer Test	3-Key Triple-DES with CMAC generate and verify, CBC mode.
RSA	Known Answer Test	Sign and verify using 2048 bit key, SHA256, PKCS#1
DSA	Known Answer Test	Sign and verify using 2048 bit key, SHA384
NIST SP 800-90A DRBG	Known Answer Test	CTR_DRBG: AES, 256-bit with and without derivation function HASH_DRBG: SHA256 HMAC_DRBG: SHA256
ECDSA	Known Answer Test	Keygen, sign, verify using P224, K233 and SHA512. (The K233 self-test is not performed for operational environments that support prime curve only.)
EC Diffie-Hellman	Known Answer Test	Shared secret calculation per NIST SP 800-56A §5.7.1.2, IG 9.6

The initialization API call invokes all power-up self-tests automatically and without operator intervention. If any component of the power-up self-test fails, an internal flag is set to prevent subsequent invocation of any cryptographic function calls. The module will only enter the FIPS Approved mode if the module is reloaded and the re-initialization succeeds. The power-up self-tests can be performed on-demand by re-initializing the module. Any failure of a power-up self-test represents a hard error, which means the module must be replaced. The operator may attempt to restart the module in an attempt to clear any error, however hard errors will require replacement of the module. Upon cryptographic service failure (including initialization, self-tests and conditional failures), the operator can call the last error API function to get the error associated with the failure.

The module performs the applicable conditional self-tests listed below:

Table 10: Module Conditional Self-Tests

Algorithm/Scheme	Type	Description
NIST SP 800-90A DRBG	Known Answer Test	As per Section 11.3 of NIST SP 800-90A - Conditional upon Instantiation, Generation, Reseed and Uninstantiation.
NIST SP 800-90A DRBG	Continuous Test	Continuous test for DRBG stuck fault.
NDRNG	Continuous Test	OS based entropy source. Blocks calls until sufficiently random.
ANSI X9.31 DRNG	Continuous Test	Continuous test for DRNG stuck fault. (non-compliant DRNG)
RSA	Pairwise Consistency Test	Performed upon the condition of RSA keypair generation.
DSA	Pairwise Consistency Test	Performed upon the condition of DSA keypair generation.
ECDSA	Pairwise Consistency Test	Performed upon the condition of ECDSA keypair generation.

Notes:

- In the event of a DRBG self-test failure, it is necessary for the calling application to uninstantiate and reinstantiate the DRBG as per the requirements of SP 800-90A.
- Pairwise Consistency Tests are performed for both Sign/Verify and Encrypt/Decrypt.
- The Module supports all NIST defined curves.
- The resulting symmetric key or generated seed is an unmodified output from the DRBG.

12. MITIGATION OF OTHER ATTACKS

The module is not designed to mitigate against attacks which are outside of the scope of FIPS 140-2.

DISTECH
CONTROLS™