Entrust® Securing Digital Identities & Information

Title:	***FIPS 140-2 Cryptographic Module Security Policy***
Product:	***Entrust Authority™ Security Toolkit for the Java® Platform***
Date:	November 19, 2012
Revision:	1.4

Entrust® Authority™
Security Toolkit for Java

Table of Contents

1 About Entrust

Entrust [NASDAQ: ENTU] provides trusted solutions that secure digital identities and information for enterprises and governments in 2,000 organizations spanning 60 countries. Offering trusted security for less, Entrust solutions represent the right balance between affordability, expertise and service. These include SSL, strong authentication, fraud detection, digital certificates and PKI. For information, call 888-690-2424, e-mail entrust@entrust.com or visit www.entrust.com.

2 Introduction

This document contains a description of the Entrust Authority™ Security Toolkit for the Java® Platform (JavaTk) Cryptographic Module Security Policy. It contains a specification of the rules under which the JavaTk cryptographic module must operate. These security rules were derived from the requirements of FIPS 140-2 [FIPS].

2.1 Purpose of the Security Policy

There are three major reasons that a security policy is defined for and must be followed by the cryptographic module:

- It is required for FIPS 140-2 validation.
- It allows individuals and organizations to determine whether the cryptographic module, as implemented, satisfies the stated security policy.
- It describes the capabilities, protection, and access rights provided by the cryptographic module, allowing individuals and organizations to determine whether it will meet their security requirements.

3 Cryptographic Module

3.1 Validation

The Java toolkit validation is at FIPS 140-2 (level 2) overall.

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

Table 1: Security Level by FIPS 140-2 Section

3.2 Definition

This section defines the cryptographic module that is being submitted for validation to FIPS 140-2, Level 2. The JavaTk cryptographic module is defined as a multi-chip standalone cryptographic module according to FIPS 140-2.

The module consists of the following generic components:

1. A commercially available general-purpose hardware-computing platform (GPC). A generic high-level block diagram for such a platform is provided in Figure 1.
2. A commercially available Operating System (OS) that runs on the above platform.
3. A Java Runtime Environment (JRE). The following JRE's are supported by the module: Oracle J2RE 6 and Oracle J2RE 7
4. The JavaTk 8.0 patch security kernel; a software component consisting of a set of JavaTk '.jar' files that contain classes that provide cryptographic services. The JavaTk security kernel runs on the below platform, operating system, and Java runtime environment. This component is custom designed and written by Entrust in the Java programming language and is identical, at the source code level, for all identified hardware platforms and operating systems. The source code is compiled into Java byte-code for interpretation by the Java Virtual Machine on the below OS or Browser.

The cryptographic module supports multiple platform configurations, which are also the FIPS 140-2 test platforms (OS and JRE as noted below). For each configuration a GPC was used with the indicated OS and JRE

JavaTK was tested on the following GPC and OS's:

1. Dell Optiplex 755, 3.0 GHz Intel Core 2 Duo E8400, 64-bit
2. Microsoft Windows Server 2008 R2

Common Criteria Report:
http://www.commoncriteriaportal.org/files/epfiles/st_vid10390-vr.pdf

The following JRE's are supported by the module: Oracle J2RE 6 and J2RE 7.

As indicated by NIST in the FIPS 140-2 implementation guidance below, the GPC and OS were installed and configured to be CC EAL2 compliant. The JavaTk cryptographic module maintains its FIPS 140-2 validation status when used on any GPC provided that the GPC incorporates the specified CC evaluated EAL2 (or equivalent) operating system/mode/operational settings or another compatible CC evaluated EAL2 (or equivalent) operating system with like mode and operational settings. For details on the platforms on which the JavaTk is supported, refer to the Entrust Platform Support and Integration Center.

> *"For Level 2 Operational Environment, a software cryptographic module will remain compliant with the FIPS 140-2 validation when operating on any GPC provided that the GPC incorporates the specified CC evaluated EAL2 (or equivalent) operating system/mode/operational settings or another compatible CC evaluated EAL2 (or equivalent) operating system with like mode and operational settings." [IG Section G.5]*

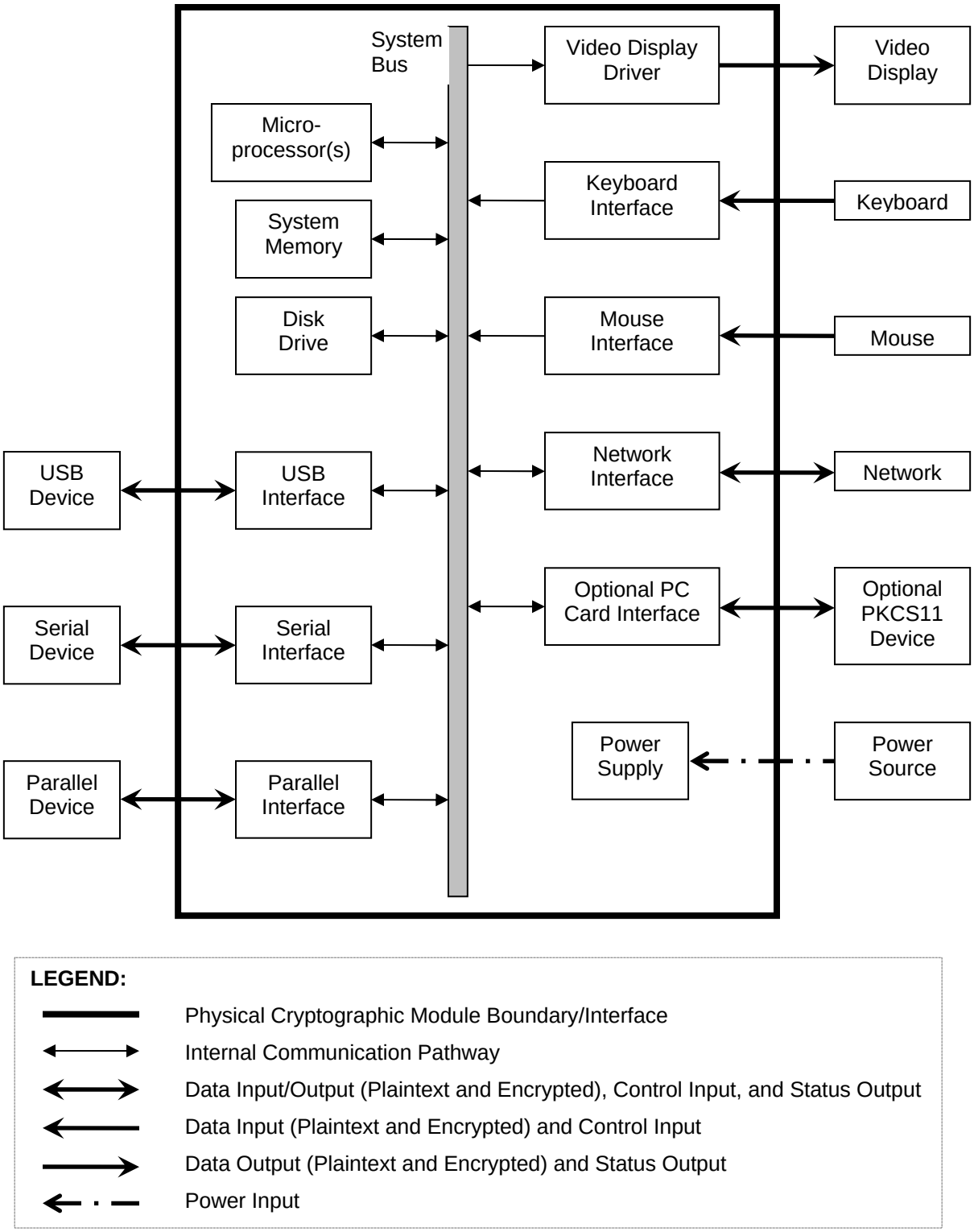

Figure 1: Cryptographic module block diagram for hardware (Physical).

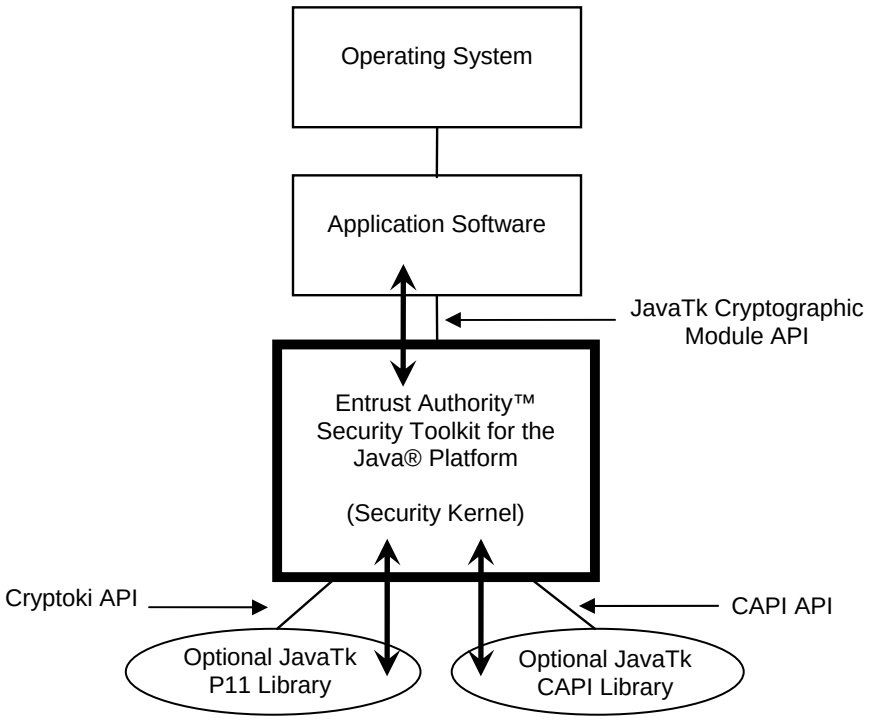

Figure 2: Cryptographic module block diagram for software (Logical).

3.3 Security Kernel

The JavaTk security kernel is one component of the JavaTk cryptographic module; it contains the set of JavaTk classes that implement cryptographic algorithms and provide cryptographic services. The security kernel provides a set of functions (API) that allows developers to integrate security into the applications they design. This API (the cryptographic module API) is the logical interface to the cryptographic module and is described in detail in the JavaTk JavaDoc documentation distribution [API].

The JavaTk can be deployed either using the single combined JAR file, or an operator selected set of uncombined JAR files. In either case, the JAR files listed below make-up the JavaTk security kernel:

Combined JAR:

- enttoolkit.jar

Uncombined JARs:

- entbase.jar
- entcapi.jar
- entp5.jar
- entp11.jar

- entp12.jar
- entroaming.jar

3.4 Ports and Interfaces

The JavaTk cryptographic module is considered according to the requirements of FIPS 140-2 to be a multi chip standalone module. The table below describes a mapping of logical interfaces to physical ports:

FIPS 140-2 Interface	Logical Interface	Physical Interface
Data Input Interface	Input parameters of module function calls	Ethernet/Network Port, USB Port, Parallel Port, Serial Port, PKCS 11 Port
Data Output Interface	Output parameters and return values of module function calls	Ethernet/Network Port, USB Port, Parallel Port, Serial Port, PKCS 11 Port
Control Input Interface	Module control function calls	Ethernet/Network Port, USB Port, Parallel Port, Serial Port, PKCS 11 Port, Keyboard and Mouse
Status Output Interface	Return values from module status function calls	Ethernet/Network Port, USB Port, Parallel Port, Monitor, Serial Port, PKCS 11 Port
Power Interface	Initialization function	Power Interface

Table 2: Mapping Logical Interfaces to Physical Ports

4 Specification of the Security Policy

4.1 Identification and Authentication Policy

FIPS 140-2 requires that roles be defined for operators of the cryptographic module. In order to perform a service using the cryptographic module the operator must first assume a role. As described in [FIPS] the following roles are mandatory (all of which are supported by the cryptographic module):

> **User**: "*The role assumed to perform general security services, including cryptographic operations and other Approved security functions.*"
>
> **Crypto Officer**: "*The role assumed to perform a set of cryptographic initialization or management functions (e.g., module initialization, input/output of cryptographic keys and CSPs, and audit functions).*"

This cryptographic module uses role based authentication. Both the User and Crypto Officer roles are authenticated via password when the operator logs into the OS. As described section 4.2 Access Control Policy, the operator implicitly assumes either the User or Crypto Officer role for each call to a Java toolkit API.

Role	Type of Authentication	Authentication Data
User	Password	Minimum 8 characters on the range [a-z,A-Z,0-9]
Crypto Officer	Password	Minimum 8 characters on the range [a-z,A-Z,0-9]

Table 3: Roles and Required Identification and Authentication

Authentication Mechanism	Strength of Mechanism
Password	For 8 characters on the range [a-z,A-Z,0-9] the probability that a random attempt will succeed is less than one in 1,000,000, and the probability that multiple attempts over a one minute period on the GPC will succeed is less than one in 100,000. The strength of the password authentication mechanism depends on the range from which the password is selected. When the OS is configured to require a minimum 8 character password on the range [a-z,A-Z,0-9] this mechanism provides 62^8 possibilities. During a one minute period, $62^8/10^5$ attempts are required for a one in 100,000 probability that one of the attempts will succeed. That would require more than 36 million operations per second and is not feasible on the GPC. (motivated by AS03.25, AS03.26, AS03.21)

Table 4: Strengths of Authentication Mechanisms

4.2 Access Control Policy

Each service offered by the cryptographic module has been assigned a role. To perform a service the operator calls a cryptographic module API and by doing so implicitly assumes the assigned role. The operator can input/output data including cryptographic keys and critical security parameters (CSP) only through the parameters and the return value provided by the API. The type of access the operator has to cryptographic keys and CSPs (read/write/execute) is restricted by the API and how it operates on each value.

The services that are provided to each authorized role are listed in the table below. The module does not support any unauthenticated services.

Role	Authorized Services
Crypto Officer	Module Initialization (module self-tests) Perform Self-Tests (module self-tests) Show Status (module status) Key Generation
User	Perform Approved Security Function • Symmetric Cipher Encryption/Decryption • Digital Signature Generation/Verification • Hash Generation • Random Number Generation • MAC Generation/Verification • Key Transport (primitive) • Key Agreement (scheme/primitive) Perform Regular Function • Key Input/Output (logical port) • Key Derivation (primitive) • Modify Object • Query Object • Non-Cryptographic Operation • Cryptographic Operation • Initialization • Zeroization

Table 5: Services Authorized for Roles

The access that each service provides to security-related information (keys and CSPs) is listed in the table below.

Service	Cryptographic Keys and CSPs	Type(s) of Access (e.g., RWE)
Module Initialization (module self-tests)	None	None
Perform Self-Tests (module self-tests)	None	None
Show Status (module status)	None	None
Key Generation	AES enc/dec keys Triple-DES enc/dec keys Triple-DES mac keys RSA signing keys RSA transport keys DSA signing keys ECDSA signing keys ECDH agreement keys DH agreement keys HMAC keys	RWE RWE RWE RWE RWE RWE RWE RE RWE RWE
Symmetric Cipher Encryption/Decryption	AES enc/dec keys Triple-DES enc/dec keys	RE RE
Digital Signature Generation/Verification	RSA signing keys DSA signing keys ECDSA signing keys	E E E
Hash Generation	None	None
Random Number Generation	Seed Seed Key Entropy String	E R R
MAC Generation/Verification	Triple-DES mac keys HMAC keys Triple-DES CMAC keys AES-CMAC Keys	E E E E
Key Agreement (scheme/primitive)	ECDH agreement keys DH agreement keys Shared Secret	E E R
Key Transport (primitive)	RSA transport keys	E
Key Input/Output (logical port)	AES enc/dec keys Triple-DES enc/dec keys Triple-DES mac keys Triple-DES integrity keys RSA signing keys RSA transport keys DSA signing keys ECDSA signing keys ECDH agreement keys DH agreement keys HMAC keys	RW RW RW RW RW RW RW RW RW RW RW
Key Derivation (primitive)	Secret	RW
Modify Object	None	None
Query Object	None	None
Non-Cryptographic Operation	None	None
Cryptographic Operation	None	None

Initialization	None	None
Zeroization	None	None

Table 6: Access Rights to CSPs within Services

4.3 Physical Security Policy

The cryptographic module is software-based and does not provide any physical security mechanisms. There are no actions required by the operator to ensure that the physical security of the module is maintained.

4.4 Mitigation of Other Attacks Policy

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

Other Attacks	Mitigation Mechanism	Specific Limitations
None	N/A	N/A

Table 7: Mitigation of Other Attacks

5 Cryptographic Algorithm Support

The following table contains the set of validated FIPS Approved algorithms (including appropriate algorithm validation certificates) that can be used in FIPS mode.

Important: the overall bits of security of an algorithm depends both bits of security offered by the algorithm itself and bits of security by the key it is used with; the overall bits of security of the algorithm is always the lower of the two.

Algorithm (Bits of Security)	Supported Key Size/Type (Bits of Security)	Certificate Number
Symmetric Cipher Algorithms		
AES (depends on key)	128 bit 192 bit 256 bit	#1935
Triple-DES (depends on key) • For two-key TDES, the number of blocks encrypted with a given key must be less than or equal to 2^20.	160 bit (80) 192 bit (112	#1261
Digital Signature Algorithms		
FIPS 186-2 DSA • DSA-SHA1 (80*)	DSA-1024	#617
FIPS 186-3 ECDSA • ECDSA-SHA1 (80*) • ECDSA-SHA224 (112) • ECDSA-SHA256 (128) • ECDSA-SHA384 (192) • ECDSA-SHA512 (256)	EC-ansix9p160k1 (80) EC-ansix9p160r1 (80) EC-ansix9p160r2 (80) EC-brainpoolP160r1 (80) EC-brainpoolP160t1 (80) EC-ansix9p192k1 (80) EC-P-192 (80) EC-brainpoolP192r1 (80) EC-brainpoolP192t1 (80) EC-ansix9p224k1 (112) EC-P-224 (112) EC-brainpoolP224r1 (112) EC-brainpoolP224r1 (112)	#277

	EC-ansix9p256k1 (128) EC-P-256 (128) EC-brainpoolP256r1 (128) EC-brainpoolP256t1 (128) EC-brainpoolP320r1 (128) EC-brainpoolP320t1 (128) EC-P-384 (192) EC-brainpoolP384r1 (192) EC-brainpoolP384t1 (192) EC-brainpoolP512r1 (256) EC-brainpoolP512t1 (256) EC-P-521 (256)	
FIPS 186-3 RSA • RSA-SHA1 (80*) • RSA-SHA224 (112) • RSA-SHA256 (128) • RSA-SHA384 (192) • RSA-SHA512 (256) • RSAPSS-SHA1 (80*) • RSAPSS-SHA224 (112) • RSAPSS-SHA256 (128) • RSAPSS-SHA384 (192) • RSAPSS-SHA512 (256)	RSA-1024 (80) RSA-2048 (112) RSA-3072 (128)	#1001
Hash Algorithms		
SHS • SHA1 (80*) • SHA224 (112) • SHA256 (128) • SHA384 (192) • SHA512 (256)	N/A	#1700
Random Number Generation Algorithms		
ANSI X9.31 using Triple-DES	N/A	#1019
ANSI X9.31 using 256-bit AES	N/A	#1019
DRBG using SHA512	N/A	#170
FIPS 186-2 using SHA1	N/A	#1019
Message Authentication Code (MAC) Algorithms		
HMAC • HMAC-SHA1 (160) • HMAC-SHA224 (224) • HMAC-SHA256 (256) • HMAC-SHA384 (384) • HMAC-SHA512 (512)	80 bit (80) 112 bit (112) 128 bit (128) 192 bit (192) 256 bit (256)	#1168
Triple-DES MAC (64)	192 bit (112)	#1261 vendor affirmed
AES CMAC (128)	128 bit 192 bit 256 bit	#1935
Triple-DES CMAC (64)	192 bit (112)	#1261
AES GCM	128 bit 192 bit 256 bit	#1954

Key Agreement Schemes		
EC Diffie-Hellman from SP800-56A - 6.1.2.2 Ephemeral Unified Model, C(2, 0, ECC CDH) - 6.2.2.2 One-Pass Diffie-Hellman, C(1, 1, ECC CDH) - 6.3.2 Static Unified Model, C(0, 2, ECC CDH) - (key agreement; key establishment methodology provides between 80 and 256 bits of encryption strength)	EC-ansix9p160k1 (80) EC-ansix9p160r1 (80) EC-ansix9p160r2 (80) EC-brainpoolP160r1 (80) EC-brainpoolP160t1 (80) EC-ansix9p192k1 (80) EC-P-192 (80) EC-brainpoolP192r1 (80) EC-brainpoolP192t1 (80) EC-ansix9p224k1 (112) EC-P-224 (112) EC-brainpoolP224r1 (112) EC-brainpoolP224r1 (112) EC-ansix9p256k1 (128) EC-P-256 (128) EC-brainpoolP256r1 (128) EC-brainpoolP256t1 (128) EC-brainpoolP320r1 (128) EC-brainpoolP320t1 (128) EC-P-384 (192) EC-brainpoolP384r1 (192) EC-brainpoolP384t1 (192) EC-brainpoolP512r1 (256) EC-brainpoolP512t1 (256) EC-P-521 (256)	CVL #16
SP 800-135 KDF	X9.63 SHA-1 KDF	Vendor Affirmed

Table 8: FIPS-Approved Algorithms

* SHA-1 provides less than 80-bits of security when used in a digital signature algorithm; it provides only 60-bits of security strength against collisions and is thus not recommended for use in new applications.

The following table contains the set of FIPS Allowed algorithms (including appropriate key sizes) that can also be used in FIPS mode.

Algorithm (Bits of Security)	Supported Key Size/Type (Bits of Security)
Key Transport Primitives (Asymmetric Cipher Algorithms)	
RSA • PKCS1-v1.5 • PKCS1-v2 OAEP • (key transport; key establishment methodology provides between 80 and 128 bits of encryption strength)	RSA-1024 (80) RSA-2048 (112) RSA-3072 (128)
Key Agreement Primitives	
Diffe-Hellman (depends on key) • (key agreement; key establishment methodology provides 80 of encryption strength)	Diffie-Hellman-1024 (80)

EC Diffie-Hellman • Standard primitive for the single pass scheme with ANSI X9.63 SHA-1 KDF. • Modified (aka: cofactor) primitive for the single pass scheme with ANSI X9.63 SHA-1 KDF. • (key agreement; key establishment methodology provides between 80 and 256 bits of encryption strength)	EC-ansix9p160k1 (80) EC-ansix9p160r1 (80) EC-ansix9p160r2 (80) EC-brainpoolP160r1 (80) EC-brainpoolP160t1 (80) EC-ansix9p192k1 (80) EC-P-192 (80) EC-brainpoolP192r1 (80) EC-brainpoolP192t1 (80) EC-ansix9p224k1 (112) EC-P-224 (112) EC-brainpoolP224r1 (112) EC-brainpoolP224r1 (112) EC-ansix9p256k1 (128) EC-P-256 (128) EC-brainpoolP256r1 (128) EC-brainpoolP256t1 (128) EC-brainpoolP320r1 (128) EC-brainpoolP320t1 (128) EC-P-384 (192) EC-brainpoolP384r1 (192) EC-brainpoolP384t1 (192) EC-brainpoolP512r1 (256) EC-brainpoolP512t1 (256) EC-P-521 (256)	
AESWrap (depends on key) • (key agreement; key establishment methodology provides between 128 and 256 bits of encryption strength)	128 bit 192 bit 256 bit	#1935

Table 9: FIPS Allowed Algorithms

The following table contains the set of non-FIPS Approved algorithms that are implemented but must not be used when operating in FIPS mode, or if used, must not be considered to provide any security.

Algorithm	Supported Key Size/Type
Symmetric Cipher Algorithms	
CAST3	40 bit 48 bit 56 bit 64 bit
CAST128	80 bit 128 bit Above are the most common CAST128 key sizes that are supported, however, all CAST128 keys between 40 and 128 bits (in 8 bit increments) are supported as well
DES	64 bit
IDEA	128 bit

RC2	128 bit Above are the most common RC2 key sizes that are supported; all RC2 keys between 8 and 1024 bits (in 8 bit increments) are supported as well
RC4	128 bit Above are the most common RC4 key sizes that are supported, however, all RC4 keys between 8 and 2048 bits (in 8 bit increments) are supported as well
Rijndael-256	128 bit 160 bit 192 bit 224 bit 256 bit
Digital Signature Algorithms	
RSA • RSA-MD2 • RSA-MD5 • RSA-SSL	RSA-1024 (80) RSA-2048 (112) RSA-3072 (128) RSA-7680 (192) RSA-15360 (256) Above are the most common RSA key sizes that are supported, however, all RSA keys 360 bits and larger (in 1 bit increments) are supported as well
Message Digest Algorithms	
MD2	N/A
MD5	N/A
RIPEMD-160	N/A
SSL3-SHA-MD5	N/A
Message Authentication Code (MAC) Algorithms	
HMAC • HMAC-MD5	
CAST128 MAC	80 bit 128 bit Above are the most common CAST128 key sizes that are supported, however, all CAST128 keys between 40 and 128 bits (in 8 bit increments) are supported as well
DES MAC	64 bit
IDEA MAC	128 bit

Key Transport Primitives (Asymmetric Cipher Algorithms)	
ElGamal	DH-1024 Above are the most common DH key sizes that are supported, however, all DH keys 512 bits and larger (in 1 bit increments) are supported as well Note: DH keys are used with ElGamal
Key Agreement Primitives	
SPEKE	SPEKE keys of all sizes 8 bits and larger (in 8 bit increments) are supported

Table 10: non-FIPS Approved Algorithms

6 Self Tests

The cryptographic module contains the following self-tests to verify its correct operation; these tests are automatically run during initialization in the FIPS Approved Mode of operation.

Power-On Self-Tests:

- Software integrity test using HMAC-SHA1
- Cryptographic algorithm known-answer tests
  - Symmetric cipher algorithms (encryption and decryption)
    - AES
    - Triple-DES
  - Random number generation algorithms
    - ANSI X9.31 using Triple-DES
    - ANSI X9.31 using 256-bit AES
    - DRBG using SHA512
    - FIPS 186-2 using SHA1
  - Message authentication code (MAC) algorithms
    - HMAC-SHA1
    - HMAC-SHA224
    - HMAC-SHA256
    - HMAC-SHA384
    - HMAC-SHA512
    - Triple-DES MAC
    - AES CMAC
    - AES GCM
    - Triple-DES CMAC
  - ECDH Key Agreement KAT (including Primitive “Z” Computation)
- Cryptographic algorithm pair-wise consistency tests
  - Digital signature algorithms
    - DSA-SHA1
    - ECDSA-SHA1
    - RSA-SHA1
    - RSAPSS-SHA1

Notes:

- SHA-X (where "X" represents the digest size) known answer tests are not required because coverage is derived from the HMAC-SHA-X tests; permitted by [IG Section 9.1].
- Triple-DES MAC known answer test is not required because coverage is derived from the Triple-DES cipher test (these algorithms are very similar); permitted by [IG Section 9.1].
- HMAC-SHA1 known answer test is not required because coverage is derived from the software integrity test which also uses HMAC-SHA1; permitted by [IG Section 9.3].
- Digital signature algorithm pair-wise consistency tests are only required using one message digest algorithm; permitted by [IG Section 9.4].

Conditional Tests:

- Random number generation algorithm continuous tests
  - ANSI X9.31 using Triple-DES
  - ANSI X9.31 using 256-bit AES
  - DRBG using SHA512
  - FIPS 186-2 using SHA1
- Key pair generation pair-wise consistency tests
  - DSA
    - DSA-SHA1 digital signature sign/verify
  - Elliptic Curve
    - ECDSA-SHA1 digital signature sign/verify
    - ECDH key agreement (Q = dG)
  - RSA
    - RSA-SHA1 digital signature sign/verify
    - RSA key transport encrypt/decrypt

If any of the above self-tests fail, the module will enter the FIPS ERROR state and any attempt to access cryptographic algorithms will throw a Fips140ErrorStateException with the following text: "Cannot perform any cryptographic operations while in FIPS Error state".

7 Operator Guidance

7.1 Assumptions

The following assumptions are made about the operating environment of the cryptographic module:

- Unauthorized reading, writing, or modification of the module's memory space (code and data) by an intruder (human or machine) is not possible; this is prevented by the process memory management of the OS.
- Replacement or modification of the legitimate cryptographic module code by an intruder (human or machine) is not feasible.
- The module is initialized to the FIPS 140-2 mode of operation.
- Critical security parameters shall not be shared between approved and non-approved mode.

7.2 Module Installation and Initialization

The following steps must be performed to install and initialize the JavaTk cryptographic module for operating in a FIPS 140-2 compliant manner:

- All the jar files and native libraries shipped with the JavaTk must be copied to the machine on which the JavaTk is being used.
- The Java runtime environment must be configured to recognize the JavaTk jar files either by setting the CLASSPATH environment variable or by using the JavaTk as an installed extension.
- To operate the JavaTk in a FIPS 140-2 compliant the cryptographic module must be initialized to operate in OPERATIONAL_FIPS mode; this is done by calling `SecurityEngine.initialize(true)`. This will authenticate the cryptographic module and run the necessary FIPS 140-2 start-up tests.

7.3 Operator Responsibilities

Secure operation of the cryptographic module in a FIPS compliant manner requires the operator fulfill the following responsibilities:

Crypto Officer:

- Input and output of plaintext private keys, secret keys, and CSPs via any physical port of the module is prohibited.
- Input and output of encrypted private keys, secret keys, and CSPs via any physical port of the module is only permitted when encrypted using an Approved algorithm (any Approved symmetric cipher algorithm).
- Key generation algorithms are accessed through the Java Cryptography Architecture and must be requested from the Entrust and or IAIK cryptographic service provider. All FIPS approved algorithms must always be requested from the Entrust cryptographic service provider only.
- Key generation must be done using an Approved RNG. By default, all key generation algorithms accessed through the Java Cryptography Architecture from the Entrust provider use an Approved RNG. However, this can be overridden by the operator of the module when the key generation algorithm is initialized. The operator must ensure that if they override the default RNG used by a key generation algorithm that they only do so using another Approved RNG. Refer to Table 8 for a list of Approved RNGs.
- No key generated by the cryptographic module shall be considered to offer more than 256-bits of security, regardless of its size.

User:

- Approved security functions (cryptographic algorithms) are accessed through the Java Cryptography Architecture and must be requested from the Entrust and or IAIK cryptographic service provider. All FIPS approved algorithms must always be requested from the Entrust cryptographic service provider only.
- Only FIPS approved algorithms (see Table 8) and FIPS allowed algorithms (see Table 9) can be used when operating in FIPS mode; all other algorithms are considered non-FIPS approved and must not be used when operating in FIPS mode.
- When using the Triple-DES algorithm with 2 keys (2-key Triple DES), the user shall not encrypt more than 2^{20} blocks of data with the same key.
- When performing ECDH
    - The bits of security of the EC key shall be at least as large as the bits of security of the key being agreed upon.
    - Ephemeral private keys shall be securely destroyed after a single use.
    - Static public keys shall
        - Be received over a trusted channel that asserts:
          The sender is the entity with whom ECDH should be performed.
          The sender possesses the matching private key.
          The sender does not use this key for other purposes.
        - Or a corresponding certificate shall be verified to confirm:

The sender is the entity with whom ECDH should be performed, because their name is in the certificate.
The sender posesses the matching private key because the certification authority performed proof-of-possession when the cert was issued.
The sender does not use this key for other purposes because the keyUsage does not contain digitalSignature.

7.4 Module Interfaces

All physical ports are available to all operators of the module. For the set of APIs (the logical interface) that is available to the Crypto Officer and User, refer to [API]. Each JavaTk API that is part of the cryptographic module interface is labeled "FIPS 140-2 Cryptographic Module API". Each cryptographic module API that is available to the Crypto Officer is labeled as "FIPS 140-2 Operator: Crypto Officer". Each cryptographic module API that is available to the User is labeled as "FIPS 140-2 Operator: User". Documentation for the API available to the Crypto Officer provided in [API] includes details on any relevant security events (e.g. exceptions) and security parameters (i.e. API parameters).

7.5 Single Operator Mode of Operation

FIPS 140-2 states that when using a software cryptographic module, the operating system shall be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded). NIST has since provided further implementation guidance regarding this matter:

> *"Software cryptographic modules implemented in client/server architecture are intended to be used on both the client and the server. The cryptographic module will be used to provide cryptographic functions to the client and server applications. When a crypto module is implemented in a server environment, the server application is the user of the cryptographic module. The server application makes the calls to the cryptographic module. Therefore, the server application is the single user of the cryptographic module, even when the server application is serving multiple clients. " [IG Section 6.1]*

This indicates that for an application built on the JavaTk cryptographic module, the application is always the single user of the cryptographic module even when multiple applications are running concurrently. This permits multiple concurrent JavaTk based applications to be running on the same machine in a FIPS 140-2 compliant manner.

8 References

Author	Title
NIST	[FIPS] FIPS PUB 140-2: Security Requirements For Cryptographic Modules, December 2002 (http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf)
NIST	[IG] Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, August 2010 (http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf)
Entrust	[API] FIPS 140-2 Cryptographic Module JavaDoc for the Entrust Authority™ Security Toolkit for the Java® Platform, January 2011 (available to customers for download from Entrust TrustedCare https://secure.entrust.com/trustedcare/)

