RSA BSAFE®

Crypto-J Software Module

Security Policy (jsafe)

Version 4.0
September 23, 2008

Cryptographic components for Java

The Security Division of EMC

Contact Information

See our Web sites for regional Customer Support telephone and fax numbers.

RSA Security Inc. RSA Security Ireland Limited.

Trademarks

ACE/Agent, ACE/Server, Because Knowledge is Security, BSAFE, ClearTrust, Confidence Inspired, e-Titlement, IntelliAccess, Keon, RC2, RC4, RC5, RSA, the RSA logo, RSA Secured, the RSA Secured logo, RSA Security, SecurCare, SecurID, SecurWorld, Smart Rules, The Most Trusted Name in e-Security, Transaction Authority, and Virtual Business Units are either registered trademarks or trademarks of RSA Security Inc. in the United States and/or other countries. EMC is a registered trademark of EMC Corporation. All other goods and/or services mentioned are trademarks of their respective companies.

License Agreement

This software and the associated documentation are proprietary and confidential to RSA Security Inc., are furnished under license and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright below. This software and any copies thereof may not be provided or otherwise made available to any other person.

Neither this software nor any copies thereof may be provided to or otherwise made available to any third party. No title to or ownership of the software or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software may be subject to civil and/or criminal liability.

This software is subject to change without notice and should not be construed as a commitment by RSA Security Inc.

Note on Encryption Technologies

This product may contain encryption technology. Many countries prohibit or restrict the use, import or export of encryption technologies and current use, import and export regulations should be followed when exporting this product.

Distribution

This document may be freely reproduced and distributed whole and intact including this Copyright Notice.

RSA Security Inc. Notice

The RC5® Block Encryption Algorithm With Data-Dependent Rotations is protected by U.S. Patent #5,724,428 and #5,835,600.

Efficient field multiplication in a normal basis is protected by U.S. Patent #6,389,442.

Compaq MultiPrime™ technology is protected by U.S. Patent #5,848,159 and is the subject of patent applications in other countries.

Other trademarks in this document are held by their respective owners.

Published September 23, 2008

Table of Contents

1 Introduction

This is a non-proprietary cryptographic toolkit security policy for the RSA BSAFE® Crypto-J 4.0 (Crypto-J) cryptographic toolkit from RSA, the Security Division of EMC (RSA). This security policy describes how the Crypto-J toolkit meets the security requirements of FIPS 140-2, and how to securely operate it. This policy is prepared as part of the Level 1 FIPS 140-2 validation of the Crypto-J toolkit.

Crypto-J provides a JSAFE and a JCE Application Programming Interface (API), in the `jsafeFIPS.jar` and `jsafeJCEFIPS.jar` files, respectively. All references to the Crypto-J toolkit apply to both interfaces unless explicitly noted.

Note: This security policy document deals only with the JSAFE interface to Crypto-J. For information about how the FIPS 140-2 validation applies to the JCE interface, see the *RSA BSAFE Crypto-J 4.0 (JCE) Security Policy*.

FIPS 140-2 (Federal Information Processing Standards Publication 140-2 – *Security Requirements for Cryptographic Modules*) details the U.S. Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the NIST website.

1.1 References

This document deals only with operations and capabilities of Crypto-J in the technical terms of a FIPS 140-2 cryptographic toolkit security policy. More information on Crypto-J and the entire RSA BSAFE product line is available at:

- http://www.rsa.com, for information on the full line of RSA products and services.
- http://www.rsa.com/node.aspx?id=1319, for an overview of Crypto-J.
- http://www.rsa.com/node.asp?id=1204, for an overview of the RSA BSAFE product range.
- Contacting RSA on page 26, for answers to technical or sales related questions.

1.2 Document Organization

This Security Policy document is one document in the complete FIPS 140-2 Certification Submission package. In addition to this document, the complete submission package contains:

- Executive Summary document
- Vendor Evidence document
- Finite State Machine document
- Toolkit software listing
- Other supporting documentation as additional references.

This document explains the Crypto-J features and functionality relevant to FIPS 140-2, and contains the following sections:

- This section, "Introduction", on page 4 provides an overview and introduction to the Security Policy.
- "Crypto-J Cryptographic Toolkit", on page 6, describes Crypto-J and how it meets the FIPS 140-2 requirements.
- "Secure Operation of Crypto-J", on page 17, addresses the required configuration for the FIPS140-mode of operation.
- "Services" on page 22, lists all the functions provided by Crypto-J.
- "Acronyms", on page 23, lists the definitions for the acronyms used in this document.
- "Contacting RSA", on page 26, lists RSA contact information for RSA news and information, product information, sales and service.

With the exception of this non-proprietary security policy, the FIPS 140-2 Certification Submission Documentation is RSA-proprietary, and releasable only under appropriate non-disclosure agreements. For access to these documents, contact RSA.

2 Crypto-J Cryptographic Toolkit

This section provides an overview of the Crypto-J toolkit, and contains the following topics:

2.1 Introduction

More than a billion copies of the RSA BSAFE technology are embedded in today's most popular software applications and hardware devices. Encompassing the most widely-used and rich set of cryptographic algorithms as well as secure communications protocols, RSA BSAFE software is a set of complementary security products relied on by developers and manufacturers worldwide.

The Crypto-J software library is the world's most trusted Java-language cryptography component, and is at the heart of the RSA BSAFE product line. It includes a wide range of data encryption and signing algorithms, including AES, Triple-DES, RC5, the RSA Public Key Cryptosystem, the Elliptic Curve Cryptosystem, the DSA government signature algorithm, and the MD5 and SHA1 message digest routines. Its software libraries, sample code and complete standards-based implementation enable near-universal interoperability for your networked and e-business applications. Any programmer using the RSA BSAFE Crypto-J tools can easily create secure applications without a background in cryptography, mathematics or number theory.

2.2 Cryptographic Toolkit

Crypto-J is classified as a FIPS 140-2 multi-chip standalone module. As such, Crypto-J is tested on particular operating systems and computer platforms. The cryptographic boundary includes Crypto-J running on selected platforms that are running selected operating systems, while configured in single user mode.

Crypto-J is validated for all FIPS 140-2 Level 1 security requirements. Crypto-J is packaged in a Java Archive (JAR) file containing all the code for the toolkit. In addition, Crypto-J relies on the physical security provided by the host on which it runs.

The JSAFE application programmer interface to the Crypto-J toolkit is provided in the `jsafeFIPS.jar` file.

Crypto-J is tested on the following platforms:

- Microsoft® Windows® XP SP2 (32-bit) with Sun™ JRE™ 1.4.2
- Microsoft Windows XP SP2 (32-bit) with Sun JRE 1.5
- Microsoft Windows XP SP2 (32-bit) with Sun JRE 1.6.

In order to place Microsoft Windows XP SP2 into single user mode, Guest accounts, Server services, Terminal services, Remote registry service, Remote Desktop service, and Remote assistance must be disabled. For detailed instructions, please refer to Microsoft's support site.

Compliance is maintained on platforms for which the binary executable remains unchanged. This includes (but is not limited to):

- Microsoft
  - Windows 2000 Professional, Service Pack 4, Sun JRE 1.4.2/5.0/6.0, IBM® JRE 1.4.2/5.0
  - Windows XP SP1, Sun JRE 1.4.2/5.0/6.0, IBM JRE 1.4.2/5.0
  - Windows XP SP2, Sun JRE 1.4.2/5.0/6.0, IBM JRE 1.4.2/5.0, JRockit 5.0/6.0
  - Windows XP Professional (64 -bit), Sun JRE 5.0/6.0
  - Windows 2003 Server (32-bit), Sun JRE 1.4.2/5.0/6.0, IBM JRE 1.4.2/5.0, JRockit 5.0/6.0
  - Windows 2003 Server (64-bit), Sun JRE 5.0/6.0, JRockit 5.0/6.0
  - Windows Vista® (32-bit), Sun JRE 1.4.2/5.0/6.0, IBM JRE 1.4.2/5.0, JRockit 5.0/6.0
  - Windows Vista (64-bit), Sun JRE 5.0/6.0, JRockit 5.0/6.0.

- Sun
  - Solaris™ 9, UltraSparc v8+ (32-bit), Sun JRE 1.4.2/5.0/6.0
  - Solaris 9, UltraSparc v9 (64-bit), Sun JRE 5.0/6.0)
  - Solaris 10, UltraSparc v8+ (32-bit), Sun JRE 1.4.2/5.0/6.0
  - Solaris 10, UltraSparc v9 (64-bit), Sun JRE 5.0, 6.0, IBM JRE 5.0, JRockit 5.0
  - Solaris 10, x86 (64-bit), Sun JRE 5.0, 6.0.
- Linux®
  - Red Hat® Enterprise Linux AS 3.0, x86 (32-bit), Sun JRE 1.4.2/5.0
  - Red Hat Enterprise Linux AS 3.0, x86 (64-bit), Sun JRE 6.0
  - Red Hat Enterprise Linux AS 4.0, x86 (32-bit), Sun JRE 1.4.2/5.0/6.0, IBM JRE 5.0, JRockit 5.0/6.0
  - Red Hat Enterprise Linux AS 4.0, x86 (64-bit), Sun JRE 5.0/6.0, JRockit 5.0/6.0
  - Red Hat Enterprise Linux AS 5.0, x86 (32-bit), Sun JRE 1.4.2/5.0/6.0, IBM JRE 5.0, JRockit 5.0/6.0
  - Red Hat Enterprise Linux AS 5.0, x86 (64-bit), Sun JRE 5.0/6.0, JRockit 5.0/6.0
  - Novell® SUSE® Linux Enterprise Server 9, x86 (32-bit), Sun JRE 1.4.2/5.0/6.0
  - Novell SUSE Linux Enterprise Server 9, x86 (64-bit), Sun JRE 5.0/6.0.
- HP
  - HP-UX 11.23, Itanium 2 (32-bit), HP JRE 1.4.2/5.0
  - HP-UX 11.23, Itanium 2 (64-bit), HP JRE 5.0
  - HP-UX 11.31, Itanium 2 (32/64-bit), HP JRE 5.0.
- IBM
  - AIX 5L™ v5.3, Power PC® (32-bit), IBM JRE 1.4.2/5.0
  - AIX 5L v5.3, Power PC (64-bit), IBM JRE 5.

For a resolution on the issue of multi-user modes, see the NIST document Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program.

2.3 Toolkit Interfaces

As a multi-chip standalone toolkit, the physical interface to Crypto-J consists of a keyboard, mouse, monitor, serial ports and network adapters.

The underlying logical interface to the toolkit is the API, documented in the *RSA BSAFE Crypto-J 4.0 Developer's Guide*. The Crypto-J toolkit provides for Control Input through the API calls. Data Input and Output are provided in the variables passed with API calls, and Status Output is provided in the returns and error codes documented for each call. This is shown diagrammatically in Figure 1 Crypto-J Jsafe Logical Diagram.

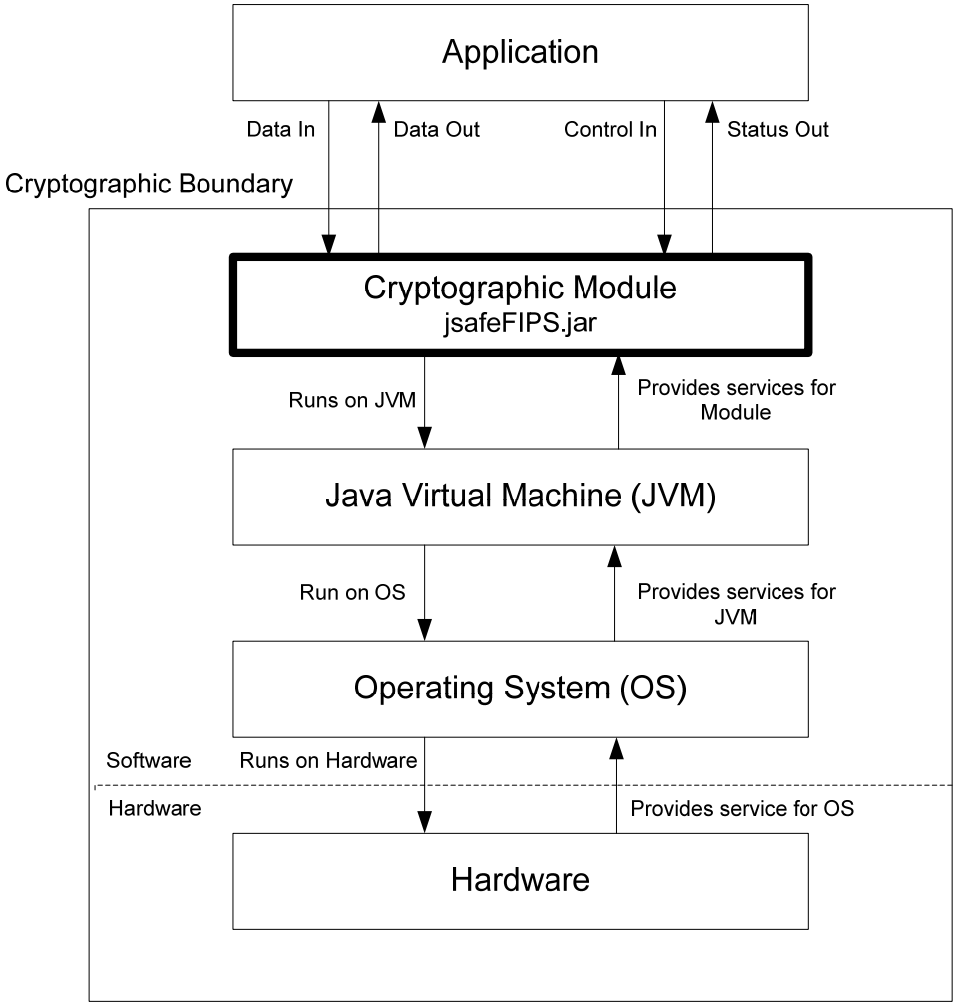

Figure 1 Crypto-J Jsafe Logical Diagram

2.4 Roles and Services

Crypto-J meets all FIPS140-2 Level 1 requirements for Roles and Services, implementing both a Crypto Officer role and a Crypto User role. As allowed by FIPS 140-2, Crypto-J does not require user identification or authentication for these roles.

The API for control of Crypto-J is through the `com.rsa.jsafe.CryptoJ` class.

2.4.1 Crypto Officer Role

An operator can assume the Crypto Officer role by invoking the `com.rsa.jsafe.CryptoJ.setRole()` method with the `CRYPTO_OFFICER_ROLE` argument.

Once in the Crypto Officer role, the operator can start the power-up self-tests on demand by calling the `com.rsa.jsafe.CryptoJ.runSelfTests()` method.

The Crypto Officer can start the power-up self-tests manually at the command prompt by navigating to the directory containing the appropriate `.jar` file, and typing:

```
java -cp jsafeFIPS.jar com.rsa.jsafe.CryptoJ -testAll
```

Alternatively, the Crypto Officer can start the power-up self-tests programmatically:

```
com.rsa.jsafe.CryptoJ.runSelfTests();
```

Note: When Crypto-J is loaded and configured for FIPS140-2 use, the power-up self tests run automatically. If the power-up self tests are re-executed explicitly (`CryptoJ.runSelfTests`) only the KATs are executed again.

2.4.2 Crypto User Role

The Crypto User role is the default operating role. An operator can explicitly assume the Crypto User role by invoking the `com.rsa.jsafe.CryptoJ.setRole()` method with the `USER_ROLE` argument.

The Crypto-J API, its functions and capabilities are documented in the *RSA BSAFE Crypto-J 4.0 Developer's Guide*. A full list of services is also provided in "Services" page 22.

2.5 Cryptographic Key Management

2.5.1 Key Generation

The Crypto-J toolkit supports the generation of the DSA, RSA, and Diffie-Hellman (DH) and ECC public and private keys. The toolkit also employs a Federal Information Processing Standard 186-2, Digital Signature Standard (FIPS 186-2) Approved random number generator as well as a FIPS Approved Dual Elliptic Curve Deterministic Random Bit Generator (ECDRBG SP 800-90) for generating asymmetric and symmetric keys used in algorithms such as AES, Triple-DES, RSA, DSA, DH and ECC.

2.5.2 Key Storage

Crypto-J does not provide long-term cryptographic key storage. Storage of keys is the responsibility of the user of Crypto-J.

The following table shows how the storage of keys and Critical Security Parameters (CSPs) are handled. The Crypto User and Crypto Officer roles have equal and complete access to all keys and CSPs.

Table 1. Key and CSP Storage

Item	Storage
AES keys	In volatile memory only (plaintext)
Triple-DES keys	In volatile memory only (plaintext)
HMAC with SHA1 and SHA-2 keys (SHA224, SHA256, SHA384, SHA512)	In volatile memory only (plaintext)
ECDH public keys	In volatile memory only (plaintext)
ECDH private keys	In volatile memory only (plaintext)
DH public key	In volatile memory only (plaintext)
DH private key	In volatile memory only (plaintext)
RSA public key	In volatile memory only (plaintext)
RSA private key	In volatile memory only (plaintext)
DSA public key	In volatile memory only (plaintext)
DSA private key	In volatile memory only (plaintext)
PRNG seeds (FIPS 186-2)	In volatile memory only (plaintext)
PRNG Keys (FIPS 186-2)	In volatile memory only (plaintext)

Item	Storage
EC DRBG Entropy	In volatile memory only (plaintext)
EC DRBG S Value	In volatile memory only (plaintext)
EC DRBG init_seed	In volatile memory only (plaintext)
HMAC DRBG Entropy	In volatile memory only (plaintext)
HMAC DRBG V Value	In volatile memory only (plaintext)
HMAC DRBG Key	In volatile memory only (plaintext)
HMAC DRBG init_seed	In volatile memory only (plaintext)

2.5.3 Key Protection

All key data resides in internally allocated data structures and can only be output using the Crypto-J API. The operating system and the Java Runtime Environment (JRE) protect memory and process space from unauthorized access.

2.5.4 Key Zeroization

Users can ensure sensitive data is properly zeroized by making use of the `clearSensitiveData()` method for clearing sensitive data. The toolkit ensures that all ephemeral sensitive data is cleared within the toolkit. For more information about clearing sensitive data, see Clearing Sensitive Data in the *RSA BSAFE Crypto-J 4.0 Developer's Guide*.

2.6 Cryptographic Algorithms

Crypto-J meets FIPS 140-2 requirements by implementing algorithm enforcement, such that when operating in `FIPS140_MODE`, only FIPS 140-approved algorithms are available for use.

The following table lists the FIPS 140-approved algorithms provided by Crypto-J, when operating in `FIPS140_MODE`.

Table 2. Crypto-J FIPS-approved Algorithms

Algorithm	Validation Certificate
AES ECB, CBC, CFB (128), OFB (128), CTR – [128, 192, 256 bit key sizes]	Certificate #670
AES CCM	Certificate #670
AES CTR	Certificate #670
CBC, CFB (64bit) , ECB, OFB (64 bit) and Triple-DES	Certificate #615
Diffie-Hellman	Non-Approved (Allowed in FIPS mode)
DSA	Certificate #252
EC-Diffie-Hellman, EC-Diffie-Hellman with Cofactor	Non-Approved (Allowed in FIPS mode)
EC-DSA, EC-DSA-SHA1	Certificate #73
ECDRBG	Vendor Affirmed: SP 800-90
FIPS 186-2 PRNG (Change Notice 1-with and without the mod q step)	Certificate #390
RSA X9.31, PKCS#1 V.1.5, PKCS#1 V.2.1 (SHA256 - PSS)	Certificate #312
RSA encrypt/decrypt	Non-Approved (Allowed in FIPS mode for key transport)
SHA-1	Certificate #703
SHA-2 (SHA224, SHA256, SHA384, SHA512)	Certificate #703
HMAC-SHA1, SHA-2 (SHA224, SHA256, SHA384, SHA512)	Certificate #354
HMAC DRBG	Vendor Affirmed: SP 800-90

The following table lists the non-FIPS 140-approved algorithms provided by Crypto-J, when operating in `NON_FIPS140_MODE`.

Table 3. Crypto-J non-FIPS140-approved algorithms

Algorithm
DES
DESX
ECAES
AES-GCM
ECIES
MD2
MD5
PBE
Random Number Generators (ANSI X9.31, MD5Random and SHA1Random)
RC2® block cipher
RC4® stream cipher
RC5® block cipher
PBE with SHA1 and Triple-DES
RSA OAEP for key transport
Raw RSA encryption and decryption
RSA Keypair Generation MultiPrime (2 or 3 primes)
RIPEMD160
HMAC-MD5

2.7 Self-tests

Crypto-J performs power-up and conditional self-tests to ensure proper operation. If the power-up self-test fails, the toolkit is disabled and throws a `SecurityException`. The toolkit can only leave the disabled state by restarting the JVM. If the conditional self-test fails, the toolkit throws a `SecurityException` and aborts the operation. A conditional self-test failure does NOT disable the toolkit.

2.7.1 Power-up Self-tests

The following power-up self-tests are implemented in Crypto-J:

- FIPS 186-2 PRNG KATs
- AES KAT
- DES KAT
- TDES KAT
- SHA-1 KAT
- SHA-224 KAT
- SHA-256 KAT
- SHA-384 KAT
- SHA-512 KAT
- HMAC SHA-1 KAT
- HMAC SHA-224 KAT
- HMAC SHA-256 KAT
- HMAC SHA-384 KAT
- HMAC SHA-512 KAT
- HMAC DRBG KAT
- EC DRBG KAT
- EC DSA KAT
- Software/firmware integrity check
- DSA, RSA, ECDSA pair-wise consistency test (power-up).

Power-up self-tests are executed automatically when Crypto-J is loaded into memory.

2.7.2 Conditional Self-tests

Crypto-J performs two conditional self-tests:

- Pair-wise consistency tests each time the toolkit generates a DSA, RSA or EC public/private key pair.
- Continuous RNG (CRNG) test each time the toolkit produces random data, as per the FIPS 186-2 standard. The CRNG test is performed on all approved and non-approved RNGs.

2.7.3 Critical Functions Test

In addition to the power-up self tests KATs, the following are also performed:

- When operating in `FIPS140_SSL_MODE`, a KAT is performed for MD5 and HMAC-MD5.
- When operating in `FIPS140_ECC_MODE`, a KAT is performed for ECAES and ECIES.
- When operating in `FIPS140_SSL_ECC_MODE`, a KAT is performed for MD5, HMAC-MD5, ECAES and ECIES.

2.7.4 Mitigation of Other Attacks

RSA key operations implement blinding by default, providing a defense against timing attacks. Blinding is implemented through blinding modes, for which the following options are available:

- Blinding mode off
- Blinding mode with no update, where the blinding value is squared for each operation
- Blinding mode with full update, where a new blinding value is used for each operation.

3 Secure Operation of Crypto-J

Crypto-J does not require any special configuration to operate in conformance with FIPS 140-2 requirements. The following guidance must be followed, however, to achieve a FIPS140 mode of operation.

3.1 Crypto Officer Guidance

The Crypto Officer is responsible for installing the toolkit. Installation instructions are provided in the *RSA BSAFE Crypto-J 4.0 Installation Guide*.

3.2 Crypto User Guidance

The Crypto User must only use algorithms approved for use in a FIPS140 mode of operation, as listed in Table 2 Crypto-J FIPS-approved Algorithms on page 13. The requirements for using the approved algorithms in a FIPS140 mode of operation are as follows:

- The bit-length for a DSA key pair must be 1024 bits.
- Random Number Generators must be seeded with values of at least 160 bits in length.
- HMAC-DRBG random data requests must be less than 2^{19} bits in length.
- Bit lengths for an RSA[1] key pair must be between 1024 and 4096 bits in multiples of 512.
- Bit lengths for the Diffie-Hellman[2] key agreement must be between 1024 and 2048 bits. Diffie-Hellman shared secret provides between 80 bits and 112 bits of encryption strength.
- Bit lengths for an HMAC key must be one half of the block size.
- If RSA key generation is requested in FIPS140 mode, the toolkit always uses the FIPS140-approved RSA X9.31 key-generation procedure. Key wrapping methodology provides between 80 and 150 bits of encryption strength.
- EC key pairs must have domain parameters from the set of NIST-recommended named curves (P192, P224, P256, P384, P521, B163, B233, B283, B409, B571, K163, K233, K283, K409, and K571). The domain parameters can be specified by name or can be explicitly defined. The module limits possible curves for Dual EC DRBG to P-256, P-384, and P-521 in accordance with SP 800-90.

[1] When used for transporting keys and using the minimum allowed modulus size, the minimum strength of encryption provided is 80 bits.

[2] Using the minimum allowed modulus size, the minimum strength of encryption provided is 80 bits.

- EC Diffie-Hellman primitives must use curve domain parameters from the set of NIST-recommended named curves listed above. The domain parameters can be specified by name, or can be explicitly defined. Using the NIST-recommended curves, the computed Diffie-Hellman shared secret provides between 80 bits and 256 bits of encryption strength.
- When using an Approved RNG to generate keys, the RNG's requested security strength must be at least as great as the security strength of the key being generated.

Crypto-J users should take care to zeroize CSPs when they are no longer needed. For more information on clearing sensitive data, see "Clearing Sensitive Data" in the *RSA BSAFE Crypto-J 4.0 Developer's Guide*.

3.3 Role Changes

If a user of Crypto-J needs to operate the toolkit in different roles, then the user must ensure that all instantiated cryptographic objects are destroyed before changing from the Crypto User role to the Crypto Officer role, or unexpected results could occur.

3.4 Modes of Operation

There are five modes of operation:

- `FIPS140_MODE`
- `FIPS140_SSL_MODE`
- `NON_FIPS140_MODE`
- `FIPS140_ECC_MODE`
- `FIPS_SSL_ECC_MODE.`

The following table lists the values that can be used in the `setMode()` method to change the Crypto-J mode of operation, and the algorithms available in that mode.

Table 4. Values in setMode to Change the Mode of Operation

Value in setMode()	Algorithms Available
`CryptoJ.FIPS140_MODE` FIPS 140-2 approved.	Provides the cryptographic algorithms listed in Crypto-J FIPS-approved Algorithms on page 12 The default random number generator is the FIPS 186-2 PRNG. This is the Crypto-J default mode on start up.
`CryptoJ.FIPS140_SSL_MODE` FIPS 140-2 approved if used with TLS protocol implementations.	Provides the same algorithms as `CryptoJ.FIPS140_MODE`, plus the MD5 message digest. This mode can be used in the context of the key establishment phase in the TLSv1, TLSv1.1 and TLSv1.2 protocols. For more information, see section 7.1 Acceptable Key Establishment Protocols in "Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program". The implementation guidance disallows the use of the SSLv2 and SSLv3 versions. Cipher suites that include non-FIPS 140-2-approved algorithms are unavailable. This mode allows implementations of the TLS protocol to operate Crypto-J in a FIPS 140-2-compliant manner with the FIPS 186-2 PRNG as the default.
`CryptoJ.NON_FIPS140_MODE` Not FIPS 140-2 approved.	Allows users to operate Crypto-J without any cryptographic algorithm restrictions.
`CryptoJ.FIPS140_ECC_MODE` Not FIPS 140-2 approved.	Provides the same algorithms as `CryptoJ.FIPS140_MODE`, plus ECAES and ECIES. The random number generator in this mode is the Dual ECDRBG.
`CryptoJ.FIPS140_SSL_ECC_MODE` Not FIPS 140-2 approved.	Provides the same algorithms as `CryptoJ.FIPS140_SSL_MODE`, plus ECAES and ECIES. The random number generator in this mode is the Dual ECDRBG. The same restrictions with respect to protocol versions and cipher suites as in `CryptoJ.FIPS140_SSL_MODE` apply.

Cryptographic algorithms can be created in different modes using the `com.rsa.jsafe.FIPS140Context` object. For more information about operating in FIPS 140 mode, see the *RSA BSAFE Crypto-J 4.0 Developers Guide*.

3.5 Operating the Cryptographic Module

The Cryptographic Module operates in `FIPS140_MODE` by default for the FIPS 140 Crypto-J toolkit variant. The initial mode can be configured through the use of the `fips140initialmode` property. See the *RSA BSAFE Crypto-J 4.0 Installation Guide* for details on how to set this property. If the property is not set, then the default mode `FIPS140_MODE` is used. The current mode of the cryptographic module can be determined with a call to the `CryptoJ.getMode()` method. When changing the mode of operation to a FIPS Approved mode, the initial mode must be configured using the `fips140initialmode` property. The module must then be reloaded to insure all power-up self tests have been properly executed. The mode of the cryptographic module can be changed by using the function `CryptoJ.setMode()` with an information identifier from Table 4 Values in setMode to Change the Mode of Operation.

Note: The `CryptoJ.setMode()` function can only be used when changing to a non-FIPS Approved mode.

After setting the cryptographic module into a FIPS approved mode, the Cryptographic Module ensures that only the FIPS approved algorithms listed in "Services" on page 22 are available to operators. To disable FIPS mode, call `the CryptoJ.setMode()` method with the mode identifier `NON_FIPS140_MODE`. The Service `CryptoJ.runSelfTests()` is restricted to operation by the Crypto Officer.

3.6 Startup Self Tests

Crypto-J offers the ability to configure when KATs are executed. To operate the Crypto-J module in a FIPS 140-2 mode the Crypto Officer must set the `com.rsa.cryptoj.jsafe.kat.strategy` property to `on.load`. For the correct configuration settings, see the *RSA BSAFE Crypto-J 4.0 Installation Guide*.

With the KAT strategy set to `on.load`, all KATs are executed on toolkit start-up, which occurs on first use. If any KAT fails, the toolkit is disabled.

3.7 Random Number Generator

In FIPS 140-2 modes, Crypto-J provides a default RNG. For the `FIPS140_MODE` and `FIPS140_SSL_MODE`, Crypto-J provides a FIPS 186-2 Pseudo Random Number Generator (PRNG) and uses this PRNG internally by default in all operations that require the generation of random numbers. For the `FIPS140_ECC_MODE` and `FIPS140_SSL_ECC_MODE`, Crypto-J implements an ECDRBG internally by default.

Users in all modes (Approved or non-Approved) can select either the FIPS 186-2, ECDRBG or HMAC DRBG when creating a RNG object and setting this object against the operation requiring random number generation (for example key generation). However, whenever DSA is used, the RNG used internally will always be the FIPS 186-2 Change Notice 1 Option 1 with mod Q PRNG.

4 Services

Crypto-J meets all FIPS 140-2 Level 1 requirements for Roles and Services, implementing both a Crypto Officer role and a Crypto User role. As allowed by FIPS 140-2, Crypto-J does not require user identification or authentication for these roles.

The following table lists the services provided by Crypto-J in terms of the toolkit interface. For more information on each function, see the *RSA BSAFE Crypto-J 4.0 Developer's Guide*.

Table 5. Service for Crypto-J (jsafeFIPS.jar)

Service	Service	Service
CryptoJ.getFIPS140Context	CryptoJ.isInECCMode	JSAFE_Obfuscator
CryptoJ.getSeeder	CryptoJ.notInFIPS140Mode	JSAFE_Object
CryptoJ.getDefaultRandom	CryptoJ.fips186RandomClearQ	JSAFE_Parameters
CryptoJ.isFIPS140Compliant	CryptoJ.fips186RandomSetQ	JSAFE_PrivateKey
CryptoJ.getState	CryptoJ.setDiscardFirstBlockForCRNG	JSAFE_PublicKey
CryptoJ.getMode	CryptoJ.disableLibrary	JSAFE_Recode
CryptoJ.setMode	JSAFE_AsymmetricCipher	JSAFE_SecretKey
CryptoJ.getRole	JSAFE_Key	JSAFE_SecureRandom
CryptoJ.setRole	JSAFE_KeyAgree	JSAFE_Session
CryptoJ.selfTestPassed	JSAFE_KeyAttributes	JSAFE_SessionSpec
CryptoJ.runSelfTests*	JSAFE_KeyPair	JSAFE_Signature
CryptoJ.isInFIPS140Mode	JSAFE_MAC	JSAFE_SymmetricCipher
CryptoJ.isInSSLMode	JSAFE_MessageDigest	JSAFE_VerifyPQG

*Only available in the Crypto Officer role.

5 Acronyms

The following table lists the acronyms, abbreviations and common terms used with Crypto-J and their definitions.

Table 6. Acronyms used with Crypto-J

Acronym	Definition
AES	Advanced Encryption Standard. A fast block cipher with a 128-bit block, and keys of lengths 128, 192 and 256 bits. This will replace DES as the US symmetric encryption standard.
ANSI X9.31	Pseudo-random number generation constructed using the SHA-1 digest algorithm and the ANSI X9.31-1998 standard. Refer to the RSA Laboratories X9.31 FAQ page for more information on the X9.31 standard.
API	Application Programming Interface.
Attack	Either a successful or unsuccessful attempt at breaking part or all of a cryptosystem. Attack types include an algebraic attack, birthday attack, brute force attack, chosen ciphertext attack, chosen plaintext attack, differential cryptanalysis, known plaintext attack, linear cryptanalysis, and middleperson attack.
CBC	Cipher Block Chaining. A mode of encryption in which each ciphertext depends upon all previous ciphertexts. Changing the Initialization Vector (IV) alters the ciphertext produced by successive encryptions of an identical plaintext.
CFB	Cipher Feedback. A mode of encryption that produces a stream of ciphertext bits rather than a succession of blocks. In other respects, it has similar properties to the CBC mode of operation.
CRNG	Continuous Random Number Generation.
CSP	Critical Security Parameters.
DES	Data Encryption Standard. A symmetric encryption algorithm with a 56-bit key.
Diffie-Hellman	The Diffie-Hellman asymmetric key exchange algorithm. There are many variants, but typically two entities exchange some public information (for example, public keys or random values) and combines them with their own private keys to generate a shared session key. As private keys are not transmitted, eavesdroppers are not privy to all of the information that composes the session key.
DRBG	Deterministic Random Bit Generator.
DSA	Digital Signature Algorithm. An asymmetric algorithm for creating digital signatures.
EC	Elliptic Curve.
ECAES	Elliptic Curve Asymmetric Encryption Scheme.

Acronym	Definition
ECB	Electronic Code Book. A mode of encryption in which identical plaintexts are encrypted to identical ciphertexts, given the same key.
ECC	Elliptic Curve Cryptography.
ECDH	Elliptic Curve Diffie-Hellman.
ECDHC	Elliptic Curve Diffie-Hellman with Components.
ECDSA	Elliptic Curve Digital Signature Algorithm.
ECIES	Elliptic Curve Integrated Encryption Scheme.
Encryption	The transformation of plaintext into an apparently less readable form (called ciphertext) through a mathematical process. The ciphertext may be read by anyone who has the key that decrypts (undoes the encryption) the ciphertext.
FIPS	Federal Information Processing Standards.
HMAC	Keyed-Hashing for Message Authentication Code.
IV	Initialization Vector. Used as a seed value for an encryption operation.
JCE	Java Cryptography Extension.
JVM	Java Virtual Machine.
KAT	Known Answer Test.
Key	A string of bits used in cryptography, allowing people to encrypt and decrypt data. Can be used to perform other mathematical operations as well. Given a cipher, a key determines the mapping of the plaintext to the ciphertext. Various types of keys include: distributed key, private key, public key, secret key, session key, shared key, subkey, symmetric key, and weak key.
MD5	A secure hash algorithm created by Ron Rivest. MD5 hashes an arbitrary-length input into a 16-byte digest.
MD5Random	Pseudo-random number generation constructed using the MD5 digest algorithm.
NIST	National Institute of Standards and Technology. A division of the US Department of Commerce (formerly known as the NBS) which produces security and cryptography-related standards.
OFB	Output Feedback. A mode of encryption in which the cipher is decoupled from its ciphertext.
OS	Operating System.
PC	Personal Computer.
private key	The secret key in public key cryptography. Primarily used for decryption but also used for encryption with digital signatures.

Acronym	Definition
PRNG	Pseudo-random Number Generator.
RC2	Block cipher developed by Ron Rivest as an alternative to the DES. It has a block size of 64 bits and a variable key size. It is a legacy cipher and RC5 should be used in preference.
RC4	Symmetric algorithm designed by Ron Rivest using variable length keys (usually 40 bit or 128 bit).
RC5	Block cipher designed by Ron Rivest. It is parameterizable in its word size, key length and number of rounds. Typical use involves a block size of 64 bits, a key size of 128 bits and either 16 or 20 iterations of its round function.
RNG	Random Number Generator.
RSA	Public key (asymmetric) algorithm providing the ability to encrypt data and create and verify digital signatures. RSA stands for Rivest, Shamir, and Adleman, the developers of the RSA public key cryptosystem.
SHA	Secure Hash Algorithm. An algorithm which creates a unique hash value for each possible input. SHA takes an arbitrary input which is hashed into a 160-bit digest.
SHA-1	A revision to SHA to correct a weakness. It produces 160-bit digests. SHA-1 takes an arbitrary input which is hashed into a 20-byte digest.
SHA1Random	Pseudo-random number generation constructed using the SHA-1 digest algorithm.
SHA-2	The NIST-mandated successor to SHA-1, to complement the Advanced Encryption Standard. It is a family of hash algorithms (SHA-256, SHA-384 and SHA-512) which produce digests of 256, 384 and 512 bits respectively.

6 Contacting RSA

The RSA Web site contains the latest news, security bulletins and information about coming events.

The RSA BSAFE Web site contains product information.

The RSA Laboratories Web site contains frequently asked questions.

6.1 Support and Service

If you have any questions or require additional information, see RSA Support or RSA SecurCare Online.

6.2 Feedback

We welcome your feedback on RSA documentation. Please email userdocs@rsa.com.