FIPS 140-2 Security Policy

BlackBerry Cryptographic Java Module Versions 2.8 and 2.8.1

BlackBerry Security Certifications, Research In Motion

Published by BlackBerry Security

Table of Contents

List of Tables

List of Figures

1 Introduction

BlackBerry® is the leading wireless solution that allows users to stay connected to a full suite of applications, including email, phone, enterprise applications, the Internet, Short Message Service (SMS), and organizer information. The BlackBerry solution is an integrated package that includes innovative software, advanced BlackBerry wireless devices and wireless network service, providing a seamless solution. The BlackBerry® Enterprise Solution architecture is shown in the following figure.

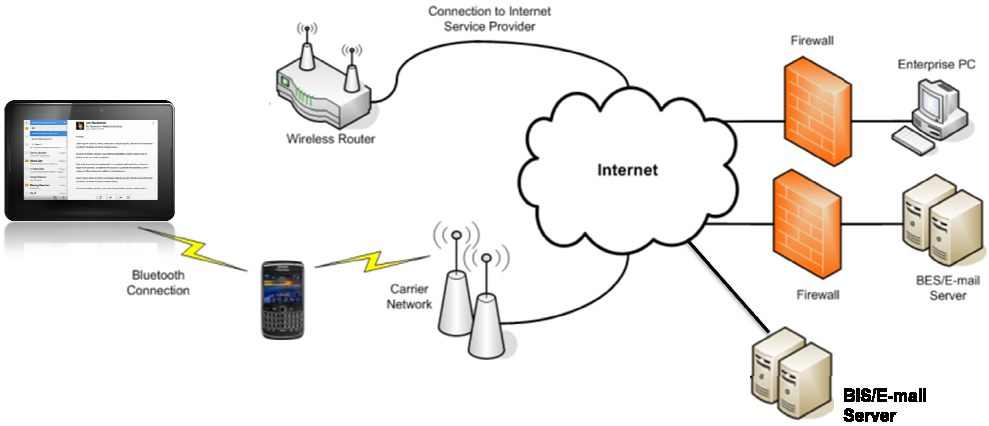


Figure 1. BlackBerry Solution Architecture

BlackBerry® PlayBook Administration Server software tightly integrates with Microsoft® Exchange, IBM® Lotus® Domino®, and Novell® GroupWise® while working with other existing enterprise systems to enable push-based access of wireless email and data. It allows users to securely send and receive email and information from enterprise data stores and applications.

BlackBerry® Desktop Software runs on your computer, allowing you to keep your computer, tablet, and smartphone data organized. It synchronizes the email and organizer information between your BlackBerry smartphone, tablet, and your computer.

For more information on the BlackBerry solution, visit http://www.blackberry.com/.

The BlackBerry Cryptographic Java Module, hereafter referred to as cryptographic module or module, is a software module that provides the following cryptographic services to the BlackBerry® PlayBook Administration Server.

- Data encryption and decryption
- Message digest and authentication code generation
- Random data generation
- Elliptic curve key pair generation
- Elliptic curve digital signature generation and verification
- Elliptic curve key agreement

The BlackBerry Cryptographic Java Module meets the requirements of the FIPS 140-2 Security Level 1 as shown in Table 1.

Section	Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services, and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	1
Cryptographic Module Security Policy	1

Table 1. Summary of achieved Security Levels per FIPS 140-2 Section

2 Cryptographic Module Specification

The BlackBerry Cryptographic Java Module is a multiple-chip stand-alone software cryptographic module that operates with the following components:

Commercially available general-purpose computer hardware

Commercially available Operating System (OS) that runs on the computer hardware

A commercially available Java Virtual Machine (JVM) that runs on the computer hardware and OS

2.1 Physical Specifications

The general-computer hardware component consists of the following devices:

1. CPU (microprocessor)
2. Memory

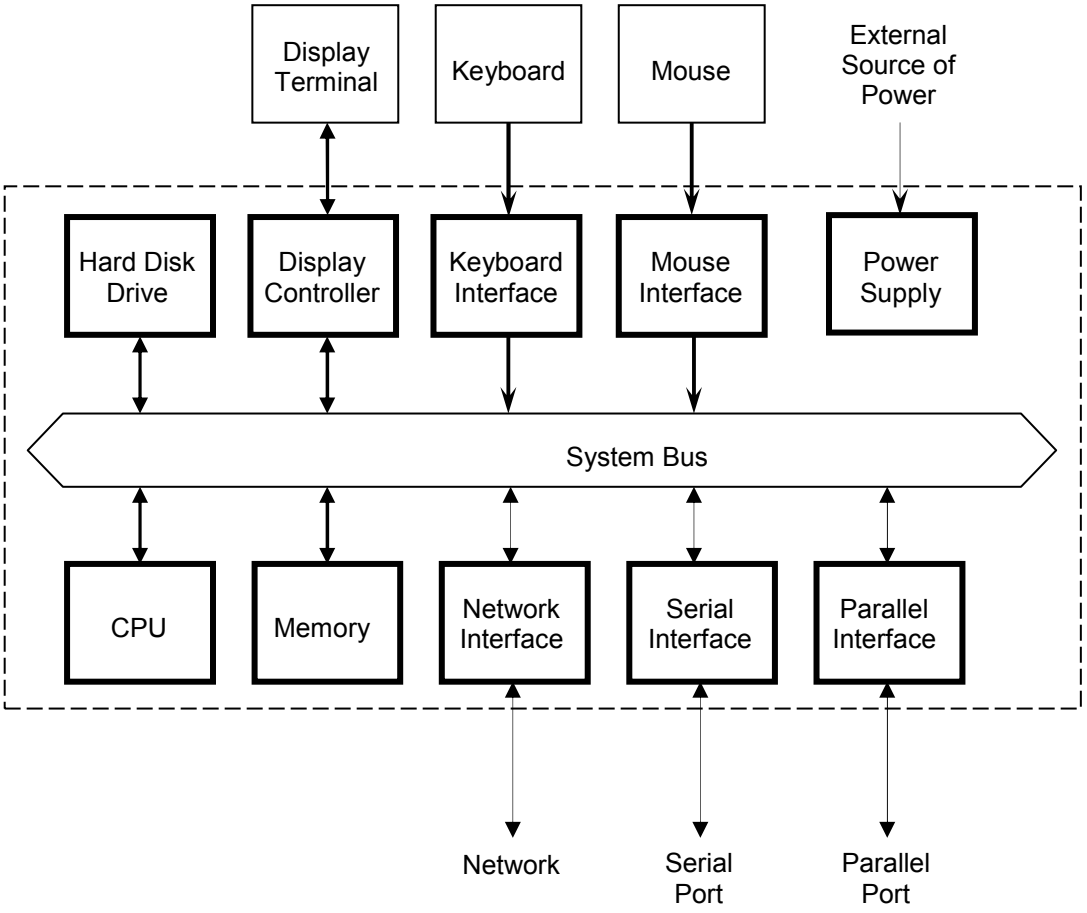
Working memory is located on the RAM and contains the following spaces (key storage is not deployed in this module):

i. Input/output buffer
ii. Plaintext/ciphertext buffer
iii. Control buffer

Program memory is also located on RAM

3. Hard Disk (or disks)
4. Display controller
5. Keyboard interface
6. Mouse interface
7. Network interface
8. Serial port
9. Parallel port
10. Power supply

Figure 2 illustrates the configuration of this component.



Key:

Cryptographic boundary

Flow of data, control input, and status output

Flow of control input

Flow of status output

Figure 1. Cryptographic Module Hardware Block Diagram

2.2 Computer Hardware, OS, and JVM

The BlackBerry Cryptographic Java Module is tested on the following representative combinations of computer hardware and OS, running the Java Runtime Environment (JRE) 1.5.0 and 1.6.0 by Sun Microsystems:

1. Solaris 10, 32-bit SPARC (Binary compatible to Solaris 9)
2. Solaris 10, 64-bit SPARC (Binary compatible to Solaris 9)
3. Red Hat Linux AS 5.5, 32-bit x86 (Binary compatible to AS 2.1/3.0/4.0/5.0)
4. Red Hat Linux AS 5.5, 64-bit x86 (Binary compatible to AS 4.0/5.0)
5. Windows Vista, 32-bit x86 (Binary compatible to Windows 98/2000/2003/XP)
6. Windows Vista, 64-bit x86 (Binary compatible to Windows 64-bit XP).
7. Windows 2008 Server, 64-bit x86

The module will run on the JREs 1.3.1, and 1.4.2, and on various hardware and OS such as,

1. Any other Solaris Platforms,
2. Any other Linux Platforms,
3. Any other Windows Platforms,
4. AIX Platforms, and
5. HP-UX Platforms,

while maintaining its compliance to the FIPS 140-2 Level 1 requirements. Thus, this validation is applicable to these JREs and platforms as well.

2.3 Software Specifications

The BlackBerry Cryptographic Java Module provides services to the Java computer language users in the form of a Java archive (JAR). The same binary is used for all identified computer hardware and OS because the JVM underneath the BlackBerry Cryptographic Java Module will absorb the differences of the computer hardware and OS.

The interface into the BlackBerry Cryptographic Java Module is through Application Programmer's Interface (API) method calls. These method calls provide the interface to the cryptographic services, for which the parameters and return codes provide the control input and status output (see Figure 3).

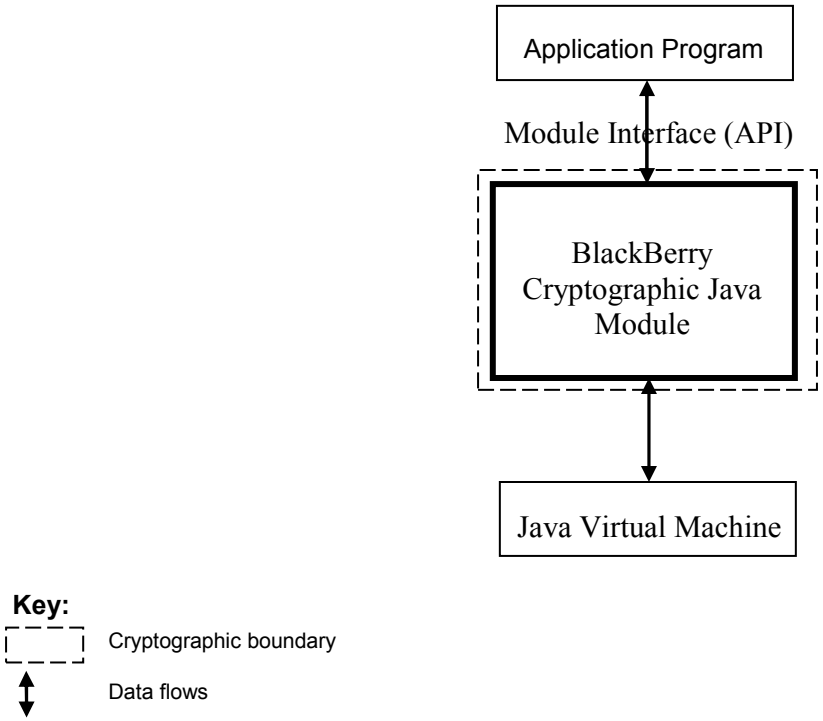


Figure 3: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The cryptographic module ports correspond to the physical ports of the GPC that is executing the module, and the module interfaces correspond to the module's logical interfaces. The following table describes the module ports and interfaces.

FIPS 140-2 interface	Module Interfaces	Module Ports
Data Input	API	Ethernet Port
Data Output	API	Ethernet Port
Control Input	API	Keyboard and Mouse
Status Output	Return Code	Display
Power Input	Initialization Function	The Power Supply is the power interface.
Maintenance	Not supported	Not supported

Table 2. Implementation of FIPS 140-2 Interfaces

4 Roles, Services, and Authentication

4.1 Roles and Services

The module supports User and Crypto Officer roles. The module does not support a maintenance role. The module does not support multiple or concurrent operators and is intended for use by a single operator, thus it always operates in a single-user mode.

Service	Crypto Officer	User
Initialization, etc.		
Initialization	x	x
Deinitialization	x	x
Self-tests	x	x
Show status	x	x
Symmetric Ciphers (AES and TDES)		
Key generation (Triple-DES only)	x	x
Encrypt	x	x
Decrypt	x	x
Key zeroization	x	x
Hash Algorithms and Message Authentication (SHA, HMAC)		
Hashing	x	x
Message authentication	x	x
Random Number Generation (pRNG)		
Instantiation	x	x
Request	x	x
CSP/key zeroization	x	x
Digital Signature (DSA, ECDSA, RSA)		
Key pair generation	x	x
Sign	x	x

Service	Crypto Officer	User
Verify	x	x
Key Zeroization	x	x
Key Agreement (Diffie-Hellman, Elliptic Curve Diffie-Hellman, ECMQV)		
Key pair generation	x	x
Shared secret generation	x	x
Key Zeroization	x	x
KeyWrapping (RSA)		
Key pair generation	x	x
Wrap	x	x
Unwrap	x	x
Key Zeroization	x	x

Table 3. Module Roles and Services

In order to operate the module securely, it is the Crypto Officer's and the User's responsibility to confine calls to those methods that have been FIPS 140-2 Approved. Thus, in the approved mode of operation, all roles shall confine themselves to calling FIPS Approved algorithms, as shown in Table 4.

4.2 Security Functions

The BlackBerry Cryptographic Java Module supports many cryptographic algorithms. The set of cryptographic algorithms supported by the BlackBerry Cryptographic Java Module is shown in Table 4.

Type	Algorithm	FIPS approved or allowed	Certificate number
Block Ciphers	DES (ECB, CBC, CFB64, OFB64)		
	TDES (TECB, TCBC, TCFB64, TOFB) [FIPS 46-3]	✕	# 964
	DESX (ECB, CBC, CFB64, OFB64)		
	AES (ECB, CBC, CFB128, OFB128, CTR, CCM, CMAC, GCM) [FIPS 197]	✕	# 1411
	ARC2 (ECB, CBC, CFB64, OFB64) [RFC 2268]		
Stream Cipher	ARC4		
Hash Functions	SHA-1 [FIPS 180-2]	✕	# 1281
	SHA-224 [FIPS 180-2]	✕	# 1281
	SHA-256 [FIPS 180-2]	✕	# 1281
	SHA-384 [FIPS 180-2]	✕	# 1281
	SHA-512 [FIPS 180-2]	✕	# 1281
	MD5 [RFC 1321]		
	MD4		
	MD2 [RFC 1115]		
Message Authentication	HMAC-SHA-1 [FIPS 198]	✕	# 832
	HMAC-SHA-224 [FIPS 198]	✕	# 832
	HMAC-SHA-256 [FIPS 198]	✕	# 832
	HMAC-SHA-384 [FIPS 198]	✕	# 832
	HMAC-SHA-512 [FIPS 198]	✕	# 832
	HMAC-MD5 [RFC 2104]		
pRNG	ANSI X9.62 RNG [ANSI X9.62]	✕	# 773

Type	Algorithm	FIPS approved or allowed	Certificate number
	DRBG [NIST SP 800-90]	✕	# 52
Digital Signature	DSA [FIPS 186-2]	✕	# 455
	ECDSA [FIPS 186-2, ANSI X9.62]	✕	# 179
	RSA PKCS1 v1.5 Signature[PKCS #1 v2.1]	✕	# 687
	RSA PSS [PKCS #1 v2.1]	✕	# 687
	ECQV		
Key Agreement	Diffie-Hellman [NIST SP 800-56A]	✕	# 8
	Elliptic Curve Diffie-Hellman [NIST SP 800-56A]	✕	# 8
	ECMQV [NIST SP 800-56A]	✕	# 8
Key Wrapping	RSA PKCS1 v1.5 Encryption [PKCS #1 v2.1]		
	RSA OAEP [NIST SP 800-56B]	✕	
	ECIES [ANSI X9.63]		

Table 4. Supported Algorithms and Standards

The Triple-DES, AES (ECB, CBC, CFB128, OFB128, CTR, CCM, GCM, and CMAC modes), SHS (SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512), HMAC-SHS (HMACSHA-1, HMAC-SHA-224, HMAC-HA256, HMAC-SHA-384, and HMAC-SHA-512), pRNG (ANSI X9.62, NIST SP 800-90), DSA, ECDSA, RSA PKCS #1 v1.5 Signature, and RSA PSS algorithms, and NIST SP 800-56A Key Establishment techniques (key agreement) Diffie-Hellman, Elliptic Curve Diffie-Hellman, and ECMQV have been independently tested. The BlackBerry Cryptographic Java Module also supports a NIST SP 800-56B Key Establishment technique (key wrapping), RSA OAEP. In order to operate the module in a FIPS Approved mode of operation only these FIPS Approved or allowed algorithms may be used.

DES, DESX, AES CCM* mode, ARC2, ARC4, MD5, MD4, MD2, and HMAC-MD5, ECQV, ECIES and RSA PKCS #1 v1.5 Encryption algorithm are supported as non FIPS Approved algorithms. In order to operate the module in a FIPS Approved mode of operation these algorithms must not be used.

Table 5 summarizes the keys and critical security parameters (CSPs) used in the FIPS mode.

Algorithm	Key and CSP	Key Size	Strength	Access
AES	Key	128-256 bits	128-256 bits	Use
Triple-DES	Key	112 bits	112bits	Create, Read, Use
HMAC	Key	160-512 bits	80-256 bits	Use
pRNG	Seed key, seed	160-512 bits	80-256 bits	Use
DSA	Key pair	1024-15360 bits	80-256 bits	Create, Read, Use
ECDSA	Key pair	160-571 bits	80-256 bits	Create, Read, Use
RSA Signature	Key pair	1024-15360 bits	80-256 bits	Create, Read, Use
Diffie-Hellman	Static/Ephemeral key pair	1024-15360 bits	80-256 bits	Create, Read, Use
Elliptic Curve Diffie-Hellman	Static/Ephemeral key pair	160-571 bits	80-256 bits	Create, Read, Use
ECMQV	Static/Ephemeral key pair	160-571 bits	80-256 bits	Create, Read, Use
RSA Key wrapping	Key pair	1024-15360 bits	80-256 bits	Create, Read, Use

Table 5. Key and CSP, Key Size, Security Strength, and Access in FIPS mode

4.3 Operator Authentication

The BlackBerry Cryptographic Java Module does not deploy an authentication mechanism. The operator implicitly selects the roles of Crypto Officer and User.

5 Finite State Model

The Finite State Model (FSM) contains the following states:

- Installed/Uninitialized
- Initialized
- Self-Test
- Idle
- Crypto Officer/User
- Error

The following list provides the important features of the state transition:

1. When the module is installed by the Crypto Officer, the module is in the Installed/Uninitialized state.
2. When the initialization command is applied to the module, that is, the module is loaded on the memory, turning to the Initialization state. Then, the module transits to the Self-Test state and automatically runs the power-up tests. While in the Self-Test state, all data output through the data output interface is prohibited. On success, the module enters idle; on failure the module enters the Error state and the module is disabled. From the Error state, the Crypto Officer might need to reinstall the module to attempt correction.
3. From the Idle state, which is entered only if self-tests have succeeded, the module can transit to the Crypto Officer/User state when an API method is called.
4. When the API method has completed successfully, the state transits back to Idle.
5. If the conditional test (continuous RNG test or pair-wise consistency test) fails, the state transits to the Error state and the module is disabled.
6. When the on-demand self-test is executed, the module enters the Self-Test state. On success, the module enters the Idle state; on failure the module enters the Error state and the module is disabled.
7. When the deinitialization command is executed, the module returns to the Installed/Uninitialized state.

6 Physical Security

Physical Security is not applicable to the BlackBerry Cryptographic Java module at FIPS 140-2 Level 1, as it is a software module only.

7 Operational Environment

The BlackBerry Cryptographic Java Module is to run in a single-user operational environment where each user application runs in a virtually separated, independent space.

Note: Modern operating systems, such as UNIX, Linux, and Windows, provide such operational environments.

8 Cryptographic Key Management

The BlackBerry Cryptographic Java Module provides the underlying functions to support FIPS 140-2 Level 1 key management. The User will select FIPS approved algorithms and will handle keys with appropriate care to build up a system that complies with FIPS 140-2. The Crypto Officer and User are responsible for selecting FIPS 140-2 validated algorithms (see Table 4).

8.1 Key Generation

The BlackBerry Cryptographic Java Module provides FIPS 140-2 compliant key generation. The underlying random number generation uses a FIPS Approved method, ANSI X9.62 RNG or DRBG.

8.2 Key Establishment

The BlackBerry Cryptographic Java Module provides the following FIPS allowed key establishment techniques [5]:

1. Diffie-Hellman
2. Elliptic Curve Diffie-Hellman
3. ECMQV
4. RSA OAEP

The Elliptic Curve Diffie-Hellman and ECMQV key agreement technique implementations support elliptic curve sizes from 160 bits to 571 bits that provide between 80 and 256 bits of security strength. The Diffie-Hellman key agreement technique implementation supports modulus sizes from 512 bits to 15360 bits that provide between 56 and 256 bits of security strength, where 1024 bits and above must be used to provide minimum of 80 bits of security in the FIPS Mode. The RSA OAEP key wrapping implementation supports modulus sizes from 512 bits to 15360 bits that provide between 56 and 256 bits of security strength, where 1024 bits and above must be used to provide minimum of 80 bits of security in the FIPS Mode.

It is the Users responsibility to ensure that the appropriate key establishment techniques are applied to the appropriate keys.

8.3 Key Entry and Output

Secret (security sensitive) keys must be imported into or exported from the BlackBerry Cryptographic Java Module in encrypted form using a FIPS Approved algorithm when crossing the module's physical boundary.

8.4 Key Storage

The BlackBerry Cryptographic Java Module is a low-level cryptographic toolkit, so it does not provide key storage.

8.5 Zeroization of Keys

The BlackBerry Cryptographic Java Module provides zeroizable interfaces which implement zeroization methods. Zeroization of all keys and CSPs are performed in the finalizing methods of the objects; JVM executes the finalizing methods every time it operates garbage collection.

9 Self-tests

9.1 Power-up Tests

9.1.1 Tests Upon Power-up

Self-tests are initiated automatically by the module at start-up. The following tests are applied:

1. Known answer tests (KATs):
   - KATs are performed on Triple-DES, AES, SHA (via HMAC-SHS), HMAC-SHS, RNG, RSA Signature Algorithm, Diffie-Hellman, Elliptic Curve Diffie-Hellman, ECMQV, and KDF (via key agreement). For DSA and ECDSA, Pair-wise Consistency Test is used.
2. Software Integrity Test:
   - The software integrity test deploys ECDSA signature validation to verify the integrity of the module.

9.1.2 On-demand Self-tests

The Crypto Officer or User can invoke on-demand self-tests by invoking a function, which is described in the Crypto Officer And User guide in Appendix C of this document.

9.2 Conditional Tests

The Continuous RNG Test is executed on all RNG generated data, examining the first 160 bits of each requested random generation for repetition. This ensures that the RNG is not stuck at any constant value.

Also, upon each generation of a RSA, DSA, or ECDSA key pair, the generated key pair is tested of their correctness by generating a signature and verifying the signature on a given message as a Pair-wise Consistency Test. Upon generation or reception of Diffie-Hellman, Elliptic Curve Diffie-Hellman, or ECMQV key pair, the key pair is tested of their correctness by checking shared secret matching of two key agreement parties as a Pair-wise Consistency Test..

9.3 Failure of Self-tests

Failure of the Self-Tests places the cryptographic module in the Error state, wherein no cryptographic operations can be performed. The module is disabled. Additionally, the cryptographic module will throw a Java exception to the caller.

10 Design Assurance

10.1 Configuration Management

A configuration management system for the cryptographic module is employed and has been described in a document that was submitted to the testing laboratory. It uses the Concurrent Versioning System (CVS) or Subversion (SVN) to track the configurations.

10.2 Delivery and Operation

Please refer to Section A.1 of Crypto Officer And User Guide in Appendix A to review the steps necessary for the secure installation and initialization of the cryptographic module.

10.3 Development

Detailed design information and procedures have been described in documentation that was submitted to the testing laboratory. The source code is fully annotated with comments, and it was also submitted to the testing laboratory.

10.4 Guidance Documents

The Crypto Officer Guide and User Guide is provided in Appendix C of this document. This appendix outlines the operations for Crypto Officer and User to verify the security of the module.

11 Mitigation of other attacks

The BlackBerry Cryptographic Java Module implements mitigation of the following attacks:

- Timing attack on RSA
- Attack on biased private key of DSA

11.1 Timing Attack on RSA

When employing Montgomery computations, timing effects allow an attacker to tell when the base of exponentiation is near the secret modulus. This leaks information concerning the secret modulus.

In order to mitigate this attack, the following is executed: The bases of exponentiation are randomized by a novel technique that requires no inversion to remove.

Note: remote timing attacks are practical: http://crypto.stanford.edu/~dabo/papers/ssl-timing.pdf.

11.2 Attack on Biased Private Key of DSA

The standard for choosing ephemeral values in DSA signature introduce a slight bias. Means to exploit these biases were presented to ANSI by D. Bleichenbacher. In order to mitigate this attack, the following is executed: The bias in the RNG is reduced to levels which are far below the Bleichenbacher attack threshold.

Change Notice 1 of FIPS 186-2 is published to mitigate this attack: http://csrc.nist.gov/CryptoToolkit/tkdigsigs.html.

Appendix A Acronyms

Introduction

This appendix lists the acronyms that are used in this document.

Acronyms

Acronym	Full term
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	application programming interface
CAT	compare answer test
CBC	cipher block chaining
CSP	critical security parameter
CVS	Concurrent Versioning System
DEMA	differential electromagnetic analysis
DES	Data Encryption Standard
DPA	differential power analysis
EC	Elliptic curve
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
ECMQV	Elliptic Curve Menezes-Qu-Vanstone
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
HMAC	keyed-hash message authentication code
IEEE	Institute of Electrical and Electronics Engineers
KAT	known answer test
LCD	liquid crystal display
LED	light-emitting diode
OS	operating system

Acronym	Full term
PIM	personal information management
PIN	personal identification number
PKCS	Public Key Cryptography Standard
PUB	Publication
RIM	Research In Motion
RNG	random number generator
RSA	Rivest Shamir Adleman
SEMA	simple electromagnetic analysis
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SMS	Short Message Service
SPA	simple power analysis
SVN	Subversion
URL	Uniform Resource Locator
USB	Universal Serial Bus

Appendix B References

Introduction

This appendix lists the references that were used for this project. The references in this appendix are listed in the order that they are cited in this document. Uncited references are placed at the end of the list in alphabetical order by title.

References

1. *NIST Security Requirements For Cryptographic Modules, FIPS PUB 140-2, December 3, 2002.*
2. *NIST Security Requirements For Cryptographic Modules, Annex A: Approved Security Functions for FIPS PUB 140-2, January 27, 2010.*
3. *NIST Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2, June 14, 2007.*
4. *NIST Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2, July 21, 2009.*
5. *NIST Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2, October 8, 2009.*
6. *NIST Derived Test Requirements for FIPS 140-2, Draft, March 24, 2004.*
7. *NIST Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, June 15, 2010.*
8. *NIST Frequently Asked Questions for the Cryptographic Module Validation Program, December 4, 2007.*

Appendix C Crypto Officer and User Guide

Installation

In order to carry out a secure installation of the BlackBerry Cryptographic Java Module, the Crypto Officer must follow the procedure described in this section.

Installing

The Crypto Officer is responsible for the installation of the BlackBerry Cryptographic Java Module. Only the Crypto Officer is allowed to install the product.

Note: Place the cryptographic module, EccpressoFIPS.jar, in CLASSPATH or as an installed extension.

Uninstalling

Remove the jar file, `EccpressoFIPS.jar`, from the computer hardware.

Commands

Initialization

`FIPSManager.getInstance().activateFIPSMode()`

This method runs a series of Self-Tests on the module. These tests examine the integrity of the shared object, and the correct operation of the cryptographic algorithms. If these tests are successful, the module will be enabled.

Deinitialization

`FIPSManager.getInstance().deactivateFIPSMode()`

This method de-initializes the module..

Self-tests

`FIPSManager.getInstance().runSelfTests()`

This method runs a series of Self-Tests, and returns if the tests are successful, otherwise, an exception is thrown. These tests examine the integrity of the shared object, and the correct operation of the cryptographic algorithms. If these tests fail, the module will be disabled. Section A.3 of this document describes how to recover from the disabled state..

Show Status

Status can be found by calling `FIPSManager.getInstance().isFIPSMode()` and `FIPSManager.getInstance().requestCryptoOperation()`. If both methods return true, the module is in the Idle state.

When Module is Disabled

When BlackBerry Cryptographic Java Module becomes disabled, attempt to bring the module back to the Installed state by calling the deinitialization method, and then to initialize the module using the initialization method. If the initialization is successful, the module is recovered. If this attempt fails, uninstall the module and re-install it. If the module is initialized successfully by this reinstallation, the recovery is successful. If this recovery attempt fails, it indicates a fatal error. Please contact Support immediately.

Document and Contact Information

Version	Date	Description
1.0	January 06, 2012	Document creation
1.1	June 1, 2012	Updates based on Lab Comments
1.2	June 14, 2012	Added reference to version 2.8.1

Contact	Corporate office
Security Certifications team certifications@rim.com (519) 888-7465 ext. 72921	Research In Motion 295 Phillip Street Waterloo, Ontario Canada N2L 3W8 www.rim.com: www.blackberry.com