Route1®

FIPS Cryptographic Module

Security Policy

Version 2.1.0.3
July 10, 2008

Securing the Digital World™

Contact Information

Route1 Inc.,
155 University Avenue, Suite 1920
Toronto, Ontario M5H 3B7
Canada

Telephone: 416-848-8391
Fax: 416-848-8394
Email: sales@route1.com

Trademarks

Distribution

Published July 11, 2008

Table of Contents

This page is intentionally blank.

1 Introduction

This is a non-proprietary Route1™ FIPS Cryptographic Module security policy. This security policy describes how the Cryptographic Module meets the security requirements of FIPS 140-2, and how to securely operate the Cryptographic Module in a FIPS-compliant manner. This policy was prepared as part of the FIPS 140-2 Level 1 validation of the Cryptographic Module.

FIPS 140-2 (Federal Information Processing Standards Publication 140-2 — *Security Requirements for Cryptographic Modules*) details the United States Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the NIST Web site at http://csrc.nist.gov/cryptval/.

2 References

This document deals only with operations and capabilities of the Route1 FIPS Cryptographic Module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the Cryptographic Module from the following resources:

- The Route1 Web site contains information on their full line of products and services at http://www.route1.com/.
- An overview of the RSA BSAFE™ Crypto-C Micro Edition module, from which this cryptographic module is derived, is located at http://www.rsa.com/node.asp?id=1204.
- For answers to technical or sales related questions please refer to the "Contact Us" link at http://www.route1.com/.

3 Document Organization

This document explains the Cryptographic Module's FIPS 140-2 relevant features and functionality. This section, Introduction, provides an overview and introduction to the Security Policy. Route1 FIPS Cryptographic Module on page 6 describes the Cryptographic Module and how it meets FIPS 140-2 requirements. Secure Operation of the Cryptographic Module on page 13 specifically addresses the required configuration for the FIPS mode of operation. Services on page 16 lists all of the functions by the Cryptographic Module. Acronyms and Definitions on page 23 lists the definitions for the acronyms used in this document.

4 Route1 FIPS Cryptographic Module

This section provides an overview of the Route1 FIPS Cryptographic Module. The following topics are discussed:

- Introduction
- Cryptographic Module
- Module Interfaces
- Roles and Services
- Cryptographic Key Management
- Cryptographic Algorithms
- Self-Test.

4.1 Introduction

Route1 is the trusted provider of security and identity management solutions that are redefining today's digital landscape. The Route1 FIPS Cryptographic Module is a software development toolkit that enables developers to incorporate cryptographic technologies into security-cognizant applications, devices and systems.

Cryptographic technologies are at the core of Route1's MobiNET™, a communications and service delivery platform focused on identity management and entitlement-based access to systems and resources. MobiNET™ services are delivered on a number of digital form factors, such as mobile phones, handheld devices and Route1 MobiKEY® – an ultra-portable, smart-card enabled USB device. Route1 MobiKEY® is one of the most powerful and easy-to-use secure computing solutions available today

Route1 MobiNET® provides organizations with the facility to:

- Expand and manage electronic services through a single identity and entitlement management platform.
- Secure distribution of multimedia content to wired and wireless networks.
- Enhance and secure collaboration.
- Assure the integrity of commercial and financial transactions.
- Consistently and accurately identify and manage individuals or entities accessing computing resources.
- Customize entitlements to computing resources based on the individual, context and business rules.
- Enable highly secure remote access to desktops, applications and systems.

- Implement licensed content protection, digital rights management and content distribution.
- Secure all forms of integrated communications including VoIP and email.
- Provide a defense against identity theft, phishing and pharming attacks, and unauthorized website access.

The features of the Cryptographic Module include the ability to optimize code for different processors and specific speed or size requirements. Assembly-level optimizations on key processors mean Cryptographic Module algorithms can be used at increased speeds on many wireless platforms.

Also, the Cryptographic Module has received FIPS 140-2 validation and offers a full set of cryptographic algorithms including public-key (asymmetric) algorithms, symmetric (secret key) block and stream ciphers, message digests, message authentication, and Pseudo Random Number Generator (PRNG) support. Developers can implement the full suite of algorithms through a single Application Programming Interface (API) or select a specific set of algorithms to reduce code size or meet performance requirements.

Note: When operating in a FIPS-approved manner, the set of algorithm implementations is not customizable.

4.2 Cryptographic Module

This Cryptographic Module is classified as a multi-chip standalone module for FIPS 140-2 purposes. As such, the module must be tested upon a particular operating system and computer platform. The cryptographic boundary thus includes the Cryptographic Module running on selected platforms running selected operating systems while configured in “single user” mode. The Cryptographic Module was validated as meeting all FIPS 140-2 Level 1 security requirements, including cryptographic key management and operating system requirements. The Cryptographic Module is packaged as a set of dynamically loaded modules or shared library files which contain the module’s entire executable code. Additionally, the Cryptographic Module relies on the physical security provided by the host PC in which it runs.

The Route1 FIPS Cryptographic Module toolkit was tested on the following platforms:

- Microsoft Windows 2003 Server, Service Pack 1 x86 (Intel Pentium 4)
  Binary executable built with Microsoft Visual Studio 2005.
  Compliance is maintained on platforms for which the binary executable remains unchanged including (but not limited to):
    - Microsoft Windows 2000 Professional, Service Pack 4 (32-bit)
    - Microsoft Windows XP, Service Pack 1 (32-bit)
    - Microsoft Windows XP, Service Pack 2 (32-bit).

Refer to the NIST document, *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*, for resolution on the issue of “Multi User” modes. This document is located at: http://csrc.nist.gov/cryptval/140-1/FIPS1402IG.pdf.

4.3 Module Interfaces

The Cryptographic Module is evaluated as a multi-chip, standalone module. The Cryptographic Module’s physical interfaces consist of the keyboard, mouse, monitor, CD-ROM drive, floppy drive, serial ports, USB ports, COM ports, and network adapter(s). However, the module sends/receives data entirely through the underlying logical interface, a C-language API documented in the Cryptographic Module API Reference.

The module 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 through the returns, exceptions, and error codes that are documented for each call.

4.4 Roles and Services

The Cryptographic Module meets all FIPS 140-2 Level 1 requirements for Roles and Services, implementing both a User (User) role and Officer (CO) role. As allowed by FIPS 140-2, the Cryptographic Module does not support user identification or authentication for these roles. Only one role may be active at a time and the Cryptographic Module does not allow concurrent operators.

Table 1. Cryptographic Module Roles and Services

Role	Services
Officer	The Officer has access to a superset of the services that are available to the User. The Officer role may also invoke the full set of self tests inside the module.
User	The User may perform general security functions. The User may also call specific FIPS 140 module functions.

4.4.1 Officer Role

An operator assuming the Officer role can call any of the module’s functions. The complete list of the functionality available to the Officer is outlined in Services on page 16.

4.4.2 User Role

An operator assuming the User role can utilize the entire Cryptographic Module API except for the `R_FIPS140_self_test_full()` method, which is reserved for the Officer. The Cryptographic Module API functions are documented in Services on page 16.

4.5 Cryptographic Key Management

4.5.1 Key Generation

The Cryptographic Module supports generation of DSA, RSA, Diffie-Hellman (DH) and ECC public and private keys. Furthermore, the module employs a FIPS 186-2 compliant random number generator as well as a Dual Elliptic Curve Deterministic Random Bit Generator for generating asymmetric and symmetric keys used in algorithms such as AES, TDES, RSA, DSA, Diffie-Hellman or ECC.

4.5.2 Key Storage

The Cryptographic Module does not provide long-term cryptographic key storage. If a User chooses to store keys, the User is responsible for storing keys exported from the module.

Volatile (short term) memory storage of cryptographic keys employed by the cryptographic module is handled in the following manner:

Note: The User and Officer roles have equal and complete access to all keys listed in Table 2.

Table 2. Cryptographic Module Key Storage

Item	Storage
AES keys	In volatile memory only (plaintext)
Triple DES keys	In volatile memory only (plaintext)
HMAC with SHA1 and SHA2 keys	In volatile memory only (plaintext)
Diffie-Hellman public key	In volatile memory only (plaintext)
Diffie-Hellman private key	In volatile memory only (plaintext)
ECC public key	In volatile memory only (plaintext)
ECC 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 and ECDRBG)	In volatile memory only (plaintext)

4.5.3 Key Access

An authorized operator of the module has access to all key data created during the module’s operation.

4.5.4 Key Protection/Zeroization

All key data resides in internally allocated data structures and can be output only using the module's defined API. The operating system protects memory and process space from unauthorized access. The operator should follow the steps outlined in the Cryptographic Module Developer's Guide to ensure sensitive data is protected by zeroizing the data from memory when it is no longer needed.

4.6 Cryptographic Algorithms

The Cryptographic Module supports a wide variety of cryptographic algorithms. FIPS 140-2 requires that FIPS-approved algorithms be used whenever there is an applicable FIPS standard. The following table lists the FIPS approved algorithms supported by the Cryptographic Module.

Table 3. Cryptographic Module FIPS-Approved Algorithms

Algorithm	Validation Certificate
AES ECB, CBC, CFB (128), OFB (128), CTR – [128, 192, 256 bit key sizes]	Cert. 673
AES CCM	Cert. 673
3DES ECB, CBC, CFB (64bit) , and OFB (64 bit)	Cert. 618
DSA	Cert. 254
EC-DSA, EC-DSA-SHA1	Cert. 74
FIPS 186-2 PRNG (Change Notice 1-with and without the mod q step)	Cert. 392
RSA X9.31, PKCS#1 V.1.5, PKCS#1 V.2.1 (SHA256 - PSS)	Cert. 314
SHA-1	Cert. 706
SHA-224, 256, 384, 512	Cert. 706
HMAC-SHA1, SHA224, SHA256, SHA384, SHA512	Cert. 357

Table 4. Cryptographic Module Non-FIPS Approved Algorithms Allowed for Use in FIPS mode

Algorithm	Use
Diffie-Hellman	Non-Approved (Allowed in FIPS mode for key transport)
EC-Diffie-Hellman	Non-Approved (Allowed in FIPS mode for key transport)
RSA encrypt/decrypt	Non-Approved (Allowed in FIPS mode for key transport)

Table 5. Cryptographic Module Non-FIPS Approved Algorithms

Algorithm
DES
MD2

MD5
HMAC MD5
DES40
RC2
RC4
RC5
ECAES
RSA PKCS #1 V.2.0 (SHA256 - OAEP)
ANSI X9.82/NIST SP 800-90 Dual EC Deterministic Random Number Generator (ECDRBG) (non-compliant)

For more information on using the Cryptographic Module in a FIPS compliant manner refer to Secure Operation of the Cryptographic Module on page 13.

4.7 Self-Test

The Cryptographic Module performs a number of power-up and conditional self-tests to ensure proper operation.

4.7.1 Power-Up Self-Tests

The power-up self-tests implemented in the Cryptographic Module are:

- AES Known Answer Tests (KATs)
- AES CCM Known Answer Tests (KATs)
- TDES KATs
- DES KATs
- SHA-1 KATs
- SHA-256 KATs
- SHA-384 KATs
- SHA-512 KATs
- HMAC SHA-1 KATs
- HMAC SHA-224 KATs
- HMAC SHA-256 KATs
- HMAC SHA-384 KATs
- HMAC SHA-512 KATs

- RSA Sign/verify test
- DSA Sign/verify test
- DH conditional test
- ECDSA Sign/verify test
- PRNG (FIPS 186-2 and ECDRBG) KATs
- Software integrity test.

Power-up self-tests are executed automatically when the module is loaded into memory.

4.7.2 Conditional Self-Tests

The Cryptographic Module performs two conditional self-tests: a pair-wise consistency test each time the module generates a DSA, DH, RSA, or EC public/private key pair, and a continuous random number generator test each time the module produces random data per its FIPS 186-2 random number generator.

4.7.3 Critical Functions Test

When operating in `R_FIPS140_MODE_FIPS140_SSL`, a known answer test is performed for MD5 and HMAC-MD5.

When operating in `R_FIPS140_MODE_FIPS140_ECC`, a known answer test is performed for ECAES and ECDRBG.

When operating in `R_FIPS140_MODE_FIPS140_SSL_ECC`, a known answer test is performed for MD5, HMAC-MD5, ECAES and ECDRBG.

4.7.4 Mitigation of Other Attacks

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

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

5 Secure Operation of the Cryptographic Module

This section provides an overview of how to securely operate the Cryptographic Module in order to be in compliance with the FIPS 140-2 standards.

5.1 Approved DSA and RSA Modulus Sizes

In the FIPS approved mode, the DSA key-pair modulus sizes must be 1024 bits, and the RSA modulus size can range from 1024 to 4096 bits.

5.2 Modes of Operation

There are six modes of operation:

1. `R_FIPS140_MODE_FIPS140`
2. `R_FIPS140_MODE_FIPS140_ECC`
3. `R_FIPS140_MODE_FIPS140_SSL`
4. `R_FIPS140_MODE_FIPS140_SSL_ECC`
5. `R_FIPS140_MODE_NON_FIPS140`
6. `R_FIPS140_MODE_DISABLED`.

`R_FIPS140_MODE_DISABLED` indicates that the FIPS140 library is disabled, usually due to an internal or caller's usage error. The other modes vary in the set of algorithms available and which random number generator (FIPS 186-2 or ECDRBG) is the default.

Cryptographic keys must not be shared between `R_FIPS140_MODE_FIPS140`/`R_FIPS140_MODE_FIPS140_SSL` and `R_FIPS140_MODE_FIPS140_ECC` /`R_FIPS140_MODE_FIPS140_SSL_ECC`/`R_FIPS140_MODE_DISABLED`/`R_FIPS140_MODE_NON_FIPS140`.

Table 6. Cryptographic Module Modes of Operation

Cryptographic Module Mode	Description
R_FIPS140_MODE_FIPS140 FIPS Approved	This mode provides the cryptographic algorithms listed in Table 3 Cryptographic Module FIPS-Approved Algorithms and Table 4 Cryptographic Module Non-FIPS Approved Algorithms Allowed for Use in FIPS Mode above. The default Random Number Generator (RNG) is the FIPS 186-2 Pseudo Random Number Generator (PRNG).
R_FIPS140_MODE_FIPS140_ECC Not FIPS Approved	This mode provides the same algorithms as R_FIPS140_MODE_FIPS140, however the RNG in this mode is the Dual Elliptic Curve Deterministic Random Bit Generator (DRGB)

Cryptographic Module Mode	Description
	(ANSI X9.82 Part 3).
R_FIPS140_MODE_FIPS140_SSL FIPS Approved if used with TLS protocol implementations	This mode provides the MD5 message digest in addition to the algorithms available in R_FIPS140_MODE_FIPS140. This mode can be used in the context of the key establishment phase in the TLSv1 and TLSv1.1 protocol (see Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, 7.1 Acceptable Key Establishment Protocols). The implementation guidance disallows the use of the SSLv2 and SSLv3 versions. Cipher suites that include non-FIPS approved algorithms are unavailable. This mode allows implementations of the TLS protocol to operate the Cryptographic Module in a FIPS 140-2 compliant manner with the FIPS 140-2 approved FIPS 186-2 PRNG as the default.
R_FIPS140_MODE_FIPS140_SSL_ECC Not FIPS Approved	This mode provides the same algorithms as R_FIPS140_MODE_FIPS140_SSL with the addition of Elliptic Curve Cryptography (ECC). In particular the random number generator in this mode is the Dual Elliptic Curve (EC) Deterministic Random Bit Generator (DRBG) (ANSI X9.82 Part 3). The same restrictions with respect to protocol versions and cipher suites as in R_FIPS140_MODE_FIPS140_SSL apply.
R_FIPS140_MODE_NON_FIPS140 Not FIPS Approved	This mode allows users to operate the Cryptographic Module without any cryptographic algorithm restrictions.
R_FIPS140_MODE_DISABLED Not FIPS Approved	This mode indicates that the FIPS 140 library is disabled. No future transition into other modes is permitted.

5.3 Operating the Cryptographic Module

The Cryptographic Module operates in `R_FIPS140_MODE_FIPS140` by default if the Cryptographic Module is initialized with the `PRODUCT_DEFAULT_RESOURCE_LIST()`.The current mode of the cryptographic module can be determined with a call to `R_FIPS140_get_mode`(). The mode of the cryptographic module can be changed by using the function `R_FIPS140_set_mode`() with an information identifier from Table 5 Cryptographic Module Modes of Operation above.

After setting the cryptographic module into a FIPS Approved mode, the Cryptographic Module enforces that only the FIPS approved algorithms listed in Services on page 16 are available to operators. To disable FIPS mode, call `R_FIPS140_set_mode`() with the mode identifier `R_FIPS140_MODE_NON_FIPS140`.

The following Services are restricted to operation by the Officer:

- `R_FIPS140_self_tests_full()`

The user of the Cryptographic Module shall link with the static library for their platform which will load the cryptographic module's shared or dynamic link master and provider libraries at runtime.

5.4 Startup Self Tests

The Cryptographic Module offers the ability to configure when power up self tests are executed. To operate the Cryptographic Module in a FIPS 140-2 compliant manner the default shipped configuration, which executes the self tests when the module is first loaded, must be used.

5.5 Random Number Generator

The Route1 FIPS Cryptographic Module provides a FIPS 186-2 Pseudo Random Number Generator and uses this PRNG internally in all operations that require the generation of random numbers.

6 Services

The Cryptographic Module provides the following services. For details of the operation of each of these services see the Developers Guide.

Table 7. Route1 FIPS Cryptographic Module Services

Function	Function
BIO_append_filename	R_FIPS140_load_module
BIO_clear_flags	R_FIPS140_MODE_from_string
BIO_clear_retry_flags	R_FIPS140_MODE_to_string
BIO_copy_next_retry	R_FIPS140_new
BIO_debug_cb	R_FIPS140_RESULT_from_string
BIO_dump	R_FIPS140_RESULT_to_string
BIO_dump_format	R_FIPS140_ROLE_from_string
BIO_dup_chain	R_FIPS140_ROLE_to_string
BIO_f_buffer	R_FIPS140_self_tests_full
BIO_f_null	R_FIPS140_self_tests_short
BIO_find_type	R_FIPS140_set_info
BIO_flags_to_string	R_FIPS140_set_interface_version
BIO_flush	R_FIPS140_set_mode
BIO_free	R_FIPS140_set_role
BIO_free_all	R_FIPS140_STATE_from_string
BIO_get_cb	R_FIPS140_STATE_to_string
BIO_get_cb_arg	R_FIPS140_unload_module
BIO_get_close	R_FORMAT_from_string
BIO_get_flags	R_FORMAT_to_string
BIO_get_fp	R_free
BIO_get_retry_BIO	R_get_mem_functions

Function	Function
BIO_get_retry_flags	R_HW_CTX_build_device_handle_list
BIO_get_retry_reason	R_HW_CTX_free
BIO_gets	R_HW_CTX_get_device_handle_list
BIO_method_name	R_HW_CTX_get_device_handle_list_count
BIO_method_type	R_HW_CTX_get_device_handle_list_handle
BIO_new	R_HW_CTX_get_info
BIO_new_file	R_HW_CTX_iterate_devices
BIO_new_fp	R_HW_CTX_new
BIO_new_mem	R_HW_CTX_probe_devices
BIO_open_file	R_HW_CTX_set_info
BIO_pop	R_HW_DEV_get_device_driver_id
BIO_print_hex	R_HW_DEV_get_device_name
BIO_printf	R_HW_DEV_get_device_number
BIO_push	R_HW_DEV_get_info
BIO_puts	R_HW_DEV_is_equal
BIO_read	R_HW_DEV_set_info
BIO_read_filename	R_HW_DRIVER_free
BIO_reference_inc	R_HW_DRIVER_get_info
BIO_reset	R_HW_DRIVER_load_devices
BIO_retry_type	R_HW_DRIVER_new
BIO_rw_filename	R_HW_DRIVER_probe_devices
BIO_s_file	R_HW_DRIVER_set_info
BIO_s_mem	R_HW_OBJ_dup
BIO_s_null	R_HW_OBJ_free
BIO_seek	R_HW_OBJ_get_info
BIO_set_bio_cb	R_HW_OBJ_init

Function	Function
BIO_set_cb	R_HW_OBJ_new
BIO_set_cb_arg	R_HW_OBJ_set_info
BIO_set_close	R_HW_SEARCH_eof
BIO_set_flags	R_HW_SEARCH_free
BIO_set_fp	R_HW_SEARCH_get_locate_count
BIO_should_io_special	R_HW_SEARCH_locate
BIO_should_read	R_HW_SEARCH_new
BIO_should_retry	R_HW_SEARCH_next
BIO_should_write	R_HW_SEARCH_set_browse
BIO_tell	R_LIB_CTX_free
BIO_write	R_LIB_CTX_get_detail_string
BIO_write_filename	R_LIB_CTX_get_error_string
PRODUCT_DEFAULT_RESOURCE_LIST	R_LIB_CTX_get_function_string
PRODUCT_FIPS140_ECC_SWITCH_RESOURCE_LIST	R_LIB_CTX_get_info
PRODUCT_FIPS140_SSL_ECC_SWITCH_RESOURCE_LIST	R_LIB_CTX_get_reason_string
PRODUCT_FIPS140_SSL_SWITCH_RESOURCE_LIST	R_LIB_CTX_new
PRODUCT_FIPS140_SWITCH_RESOURCE_LIST	R_LIB_CTX_set_info
PRODUCT_LIBRARY_FREE	R_lock_ctrl
PRODUCT_LIBRARY_INFO	R_lock_get_cb
PRODUCT_LIBRARY_INFO_TYPE_FROM_STRING	R_lock_get_name
PRODUCT_LIBRARY_INFO_TYPE_TO_STRING	R_lock_num
PRODUCT_LIBRARY_NEW	R_lock_r
PRODUCT_LIBRARY_VERSION	R_lock_set_c
PRODUCT_NON_FIPS140_SWITCH_RESOURCE_LIST	R_lock_w
R_CR_asym_decrypt	R_locked_add
R_CR_asym_decrypt_init	R_locked_add_get_cb

Function	Function
R_CR_asym_encrypt	R_locked_add_set_cb
R_CR_asym_encrypt_init	R_lockid_new
R_CR_CTX_alg_supported	R_lockids_free
R_CR_CTX_free	R_malloc
R_CR_CTX_get_info	R_PKEY_cmp
R_CR_CTX_ids_from_sig_id	R_PKEY_CTX_free
R_CR_CTX_ids_to_sig_id	R_PKEY_CTX_get_info
R_CR_CTX_new	R_PKEY_CTX_get_LIB_CTX
R_CR_CTX_set_info	R_PKEY_CTX_new
R_CR_decrypt	R_PKEY_CTX_set_info
R_CR_decrypt_final	R_PKEY_decode_pkcs8
R_CR_decrypt_init	R_PKEY_delete_device
R_CR_decrypt_update	R_PKEY_encode_pkcs8
R_CR_DEFINE_CUSTOM_CIPHER_LIST	R_PKEY_FORMAT_from_string
R_CR_DEFINE_CUSTOM_METHOD_TABLE	R_PKEY_FORMAT_to_string
R_CR_derive_key	R_PKEY_free
R_CR_digest	R_PKEY_from_binary
R_CR_digest_final	R_PKEY_from_bio
R_CR_digest_init	R_PKEY_from_file
R_CR_digest_update	R_PKEY_from_public_key_binary
R_CR_dup	R_PKEY_get_handle
R_CR_encrypt	R_PKEY_get_info
R_CR_encrypt_final	R_PKEY_get_num_bits
R_CR_encrypt_init	R_PKEY_get_num_primes
R_CR_encrypt_update	R_PKEY_get_PKEY_CTX
R_CR_free	R_PKEY_get_private_handle

Function	Function
R_CR_generate_key	R_PKEY_get_public_handle
R_CR_generate_key_init	R_PKEY_get_purpose
R_CR_generate_parameter	R_PKEY_get_type
R_CR_generate_parameter_init	R_PKEY_iterate_fields
R_CR_get_crypto_provider_name	R_PKEY_METHOD_free
R_CR_get_default_imp_method	R_PKEY_METHOD_get_flag
R_CR_get_default_method	R_PKEY_METHOD_get_name
R_CR_get_default_signature_map	R_PKEY_METHOD_get_type
R_CR_get_detail	R_PKEY_new
R_CR_get_detail_string	R_PKEY_PASSWORD_TYPE_from_string
R_CR_get_detail_string_table	R_PKEY_PASSWORD_TYPE_to_string
R_CR_get_device_handle	R_PKEY_pk_method
R_CR_get_error	R_PKEY_print
R_CR_get_error_string	R_PKEY_public_cmp
R_CR_get_file	R_PKEY_public_to_bio
R_CR_get_function	R_PKEY_public_to_file
R_CR_get_function_string	R_PKEY_read_device
R_CR_get_function_string_table	R_PKEY_reference_inc
R_CR_get_info	R_PKEY_rsa_blinding_lib_start
R_CR_get_line	R_PKEY_rsa_no_blinding_lib_start
R_CR_get_reason	R_PKEY_set_handle
R_CR_get_reason_string	R_PKEY_set_info
R_CR_get_reason_string_table	R_PKEY_set_private_handle
R_CR_ID_from_string	R_PKEY_set_public_handle
R_CR_ID_sign_to_string	R_PKEY_set_purpose
R_CR_ID_to_string	R_PKEY_to_binary

Function	Function
R_CR_key_exchange_init	R_PKEY_to_bio
R_CR_key_exchange_phase_1	R_PKEY_to_file
R_CR_key_exchange_phase_2	R_PKEY_to_public_key_binary
R_CR_mac	R_PKEY_TYPE_from_string
R_CR_mac_final	R_PKEY_TYPE_to_string
R_CR_mac_init	R_PKEY_write_device
R_CR_mac_update	R_realloc
R_CR_new	R_remalloc
R_CR_random_bytes	R_RES_LIST_get_item
R_CR_random_seed	R_RES_LIST_get_resource
R_CR_RES_CRYPTO_CUSTOM_METHOD	R_RES_LIST_set_item
R_CR_set_info	R_RES_LIST_set_resource
R_CR_sign	R_set_mem_functions
R_CR_sign_final	R_SKEY_delete_device
R_CR_sign_init	R_SKEY_free
R_CR_sign_update	R_SKEY_get_handle
R_CR_SUB_from_string	R_SKEY_get_info
R_CR_SUB_to_string	R_SKEY_new
R_CR_TYPE_from_string	R_SKEY_read_device
R_CR_TYPE_to_string	R_SKEY_set_handle
R_CR_verify	R_SKEY_set_info
R_CR_verify_final	R_SKEY_write_device
R_CR_verify_init	R_thread_id
R_CR_verify_mac	R_thread_id_get_cb
R_CR_verify_mac_final	R_thread_id_set_cb
R_CR_verify_mac_init	R_TIME_cmp

Function	Function
R_CR_verify_mac_update	R_TIME_CTX_free
R_CR_verify_update	R_TIME_CTX_new
R_ERROR_EXIT_CODE	R_TIME_dup
R_FIPS140_free	R_TIME_export
R_FIPS140_get_default	R_TIME_free
R_FIPS140_get_failure_reason	R_TIME_get_time_mi_method
R_FIPS140_get_failure_reason_string	R_TIME_get_utc_time_method
R_FIPS140_get_info	R_TIME_import
R_FIPS140_get_interface_version	R_TIME_new
R_FIPS140_get_mode	R_TIME_offset
R_FIPS140_get_role	R_TIME_time
R_FIPS140_get_supported_interfaces	R_unlock_r
R_FIPS140_library_free	R_unlock_w
R_FIPS140_library_init	

7 Acronyms and Definitions

The following table gives an explanation of the terms and acronyms used throughout this document.

Table 8. Route1 FIPS Cryptographic Module Services

Term	Description
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.
API	Application Programming Interface
Attack	Either a successful or unsuccessful attempt at breaking part or all of a cryptosystem. Various 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.
DES	Data Encryption Standard. A symmetric encryption algorithm with a 56-bit key. See also Triple DES.
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 combine 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.
DSA	Digital Signature Algorithm. An asymmetric algorithm for creating digital signatures.
EC	Elliptic Curve
ECC	Elliptic Curve Cryptography
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
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.
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

Term	Description
	standards.
OS	Operating System
PC	Personal Computer
PDA	Personal Digital Assistant
PPC	PowerPC
privacy	The state or quality of being secluded from the view and/or presence of others.
private key	The secret key in public key cryptography. Primarily used for decryption but also used for encryption with digital signatures.
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.
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.
Triple DES	A variant of DES which uses three 56-bit keys.