Rainbow Technologies CryptoSwift HSM Cryptographic Accelerator

FIPS 140-1 Non-Proprietary
Cryptographic Module Security Policy
Hardware P/N 107316
Firmware version 5.6.27
Ver 25
7/29/01
for
Level 3 Overall
Level 4 for Self-Test
Validation

1.0 Scope

This document describes the security policy for the CryptoSwift HSM cryptographic accelerator. It is to be used for the FIPS 140-1 validation process. The board is designed to attain a level 3 overall validation and a level 4 validation in the area of Self-Test.

The following table describes the compliance level for each section of the FIPS 140-1 specification:

Cryptographic Modules:	Level	**3**
Module Interfaces:	Level	**3**
Roles and Services:	Level	**3**
Finite State Machine Model:	Level	**3**
Physical Security:	Level	**3**
Software Security:	Level	**3**
Operating System Security:	Level	**N/a**
Cryptographic Key Management:	Level	**3**
Cryptographic Algorithms:	Level	**3**
EMI/EMC:	Level	**3**
Self-Tests:	Level	**4**

If changes are made to the design of the CryptoSwift HSM, this document should be updated to incorporate the changes and reviewed by an NVLAP-accredited CMT lab.

2.0 Applicable Documents

FIPS PUB 140-1 Federal Information Processing Standard, Security Requirements for Cryptographic Modules January, 11, 1994, U.S. Department of Commerce, National Institute of Standards and Technology

Derived Test Requirements for FIPS PUB 140-1, Security Requirements for Cryptographic Modules
FINAL, March 1995, Mitre for NIST Contract 50SBNIC6732

FIPS PUB 46-3 and FIPS PUB 81, for information on the Data Encryption Standard (DES), and Triple DES algorithm.
U.S. Department of Commerce, National Institute of Standards and Technology

FIPS PUB 180-1, Secure Hash Algorithm (SHA-1), U.S. Department of Commerce, National Institute of Standards and Technology

ANSI Standard X9.17-1995, Financial Institution Key Management (Wholesale), American Banking Association, X9 Financial Services, American National Standards Institute

PKCS #1 RSA Cryptography Standard, Version 2.0,
http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/index.html
RSA Security Inc.

3.0 Overview

The CryptoSwift HSM is a cryptographic module which is used to accelerate cryptographic processing for network based electronic commerce and other network based applications. The board has two modes. These are the non-FIPS140-1 mode and the FIPS140-1 mode. In the FIPS140-1 mode, the board can be used in servers to improve the performance associated with high rate signing operations. In the non-FIPS140-1 mode, the board can be used to accelerate RSA operations for SSL connections on web servers. Other uses are limited only by the creativity of applications developers who can write to standard API's such as Cryptoki (PKCS#11).

The CryptoSwift HSM is a PCI card. It has a serial port, a Universal Serial Bus (USB) port, and an LED. The board is shipped with four tokens. These tokens plug into the USB port. The first token is used for authenticating the Security Officer to the HSM. The second token is used to for authenticating the User. The third and forth tokens are called "code tokens." One of these is held (controlled) by the Security Officer. The other held by the User. The code keys are used to move key parts (also known as "key shares") between two HSM boards. Key parts transferred by this mechanism are combined within the destination boards so that a shared secret can exist on one or more boards without having existed in plaintext outside of a family of HSM boards. The shared secret is a Key-Wrapping-Key. When two or more boards contain the same Key-Wrapping-Key, they are said to be in the same family. The Key-Wrapping-Key is used to encrypt other keys. These encrypted keys can then be transmitted between boards over untrusted paths under the control of a Rainbow Technologies key management utility. This allows boards to share keys as would be appropriate for load distribution or redundancy needs.

The key wrapping key also makes it possible for keys to be stored in encrypted form on backup tapes or hard drives for archival purposes. The keys encrypted with the Key-Wrapping-Key need never exist in plaintext form outside of an HSM.

When an operator uses an HSM, he will be assisted by a key management utility. This utility will prompt the operator when it is time to plug a particular token into a particular HSM. A particular host system may contain one or more HSM's. So that there is no confusion, the key management utility will control an LED on each HSM to alert the operator to know where to insert a particular token.

1. The HSM can detect attempts to penetrate its cryptographic envelope. If it detects a tamper attempt, the HSM will erase all of the critical security parameters that it contains.

The HSM is controlled via its PCI interface. Commands are entered via the PCI bus, and status is read from the PCI bus. Also, both plaintext and encrypted data is transmitted over the PCI interface. The serial port is disabled in the production version of the CryptoSwift HSM. A primary function of the HSM is to securely generate, store, and use private keys (particularly for signing operations).

4.0 Capabilities

The CryptoSwift HSM is capable of performing a wide variety of cryptographic calculations including DES, SHA-1, DSA, 3DES, RSA exponentiation, RC4 and HMAC. When in the FIPS 140-1 mode, the board can perform DES, 3DES, RSA Signatures, RSA Signature Verifications and SHA-1 functions. When in the non-FIPS 140-1 mode, the board can also perform the RSA exponentiation, RC4, MD5, HMAC (SHA-1 and MD5) and DSA.

The RSA signature and verification implementation is compliant with the PKCS #1 standard.

The following table describes how each cryptographic algorithm is used by our module while operating in the FIPS 140-1 Mode:

Algorithm	**How it is used by the HSM module**	**Used in FIPS 140-1 Mode?**
DES	The module provides services for encryption/decryption. As currently implemented, the plaintext key must be inputted via the PCI interface. Therefore, this algorithm is not accessible in the FIPS 140-1 Mode. The self-tests perform a known answer test on this algorithm in FIPS 140-1 Mode.	No
3DES	Used to generate Pseudo-random numbers using the X9.17 Appendix C PRNG algorithm for the purposes of key generation of RSA and 3DES keys. Encryption/decryption of every key stored in persistence storage within the module using the Master Key. Wrapping (encryption) of Private RSA Keys using the Key-Wrapping-Key for archival purposes. Unwrapping (decryption) of Private RSA Keys using the Key-Wrapping-Key for the purpose of restoring an archived key. Note: The 3DES Encrypt and Decrypt services are not available for this algorithm in FIPS mode because keys are entered in plaintext.	Yes
RSA Signature/Verification	Generation and verification of digital signatures using the RSA algorithm, in accordance with the PKCS #1 specification. Keys pairs of modulus size in the range 192 through 1024 bits, in 64 bit increments. Note: The message digest operation of the digital signature and verification function is performed outside of the cryptographic boundary for performance reasons. After the digest is computed outside the module, the module formats and pads the message digest according to the PKCS #1 standard and then uses the RSA algorithm to compute the digital signature.	Yes
SHA-1	Hashing of host-provided data. Hashing for the purpose of verifying the RSA digital signature of a firmware image.	Yes

	Hashing a 3DES key for the purpose of checking its integrity after it is split and then the corresponding shares combined.	
MD5	The module provides services to compute an MD5 message digest. As this algorithm is not FIPS-approved, the corresponding services are not available in the FIPS 140-1 Mode.	No
HMAC (SHA-1)	The module provides a service to compute HMAC using SHA-1. As currently implemented, the service requires the MAC key to be inputted unencrypted via the PCI interface, and therefore this service is not available in the FIPS 140-1 Mode.	No
HMAC (MD5)	The module provides a service to compute HMAC using MD5. Since MD5 is not a FIPS-approved algorithm, this service is not available in the FIPS 140-1 Mode.	No
RC4	The module provides services for encryption/decryption with RC4. Since RC4 is not a FIPS-approved algorithm, the corresponding services are not available in the FIPS 140-1 Mode.	No
DSA	The module provides services for generating and verifying DSA signatures. As currently implemented, the private key for signature generation must be inputted via the PCI interface. Therefore, this algorithm is not available in the FIPS 140-1 Mode. Keys pairs of modulus size in the range 512 through 1024 bits, in 64 bit increments.	No

5.0 Physical Security

The board is designed to detect tampering attempts and will zeroize critical security parameters under a variety of prescribed circumstances. These circumstances include penetration of the module's cryptographic envelope. The cryptographic envelope consists of an opaque tamper resistant lid and circuit board, and will provide clear visual evidence of tampering. The lid and circuit board are joined to form a contiguous perimeter. This perimeter encloses module components responsible for the creation, storage and processing of critical security parameters. The boundary contains intricate serpentine patterns that are used to detect tamper attempts associated with a breach of the cryptographic envelope by drilling, sawing or removal of the tamper lid.

6.0 Module Interfaces

6.1 USB (Universal Serial Bus) Interface

This is the trusted interface of the HSM. It is used for communicating with iKey1000 tokens.
Four tokens are shipped with each HSM. One will contain a pin used to authenticate the Security Officer. One will contain a pin used to authenticate the User. One will contain a key-part to be controlled by the Security Officer. One will contain a key-part to be controlled by the user. No secrets, key-parts or critical security parameters are contained within any of the tokens or within the HSM when these items are shipped from Rainbow Technologies.

6.2 Status LED (Light Emitting Diode) Interface

The LED can be in four possible states. These are off, green, orange and red.
The meaning associated with each LED state is as follows:

LED State	Meaning
off	power off
green	board is on but idle
orange	board is in the self-test state or performing a crypto function
red	board is in the error state

The true state of the HSM, will be obtainable from the status register which is read by the host over the PCI interface.

6.3 Serial Interface

The serial interface is disabled in the production version of the CryptoSwift HSM board.

6.4 PCI Interface

This interface is used to provide data and commands to the CryptoSwift HSM board. It is also used to read data and status from the CryptoSwift HSM.

6.5 Backup Battery Interface

The Backup Battery Interface is used to provide backup power to the HSM. This gives the HSM the capability to maintain and protect secrets should PCI power become unavailable. The battery is continuously monitored by the HSM for a voltage low condition. This makes it possible to alert an operator. The operator may then replace the battery. This can be done without loss of critical security parameters as long as the battery is replaced when PCI power is present. If the battery is removed while PCI power is absent, all critical security parameters contained within the HSM will be erased.

6.6 PCI Power Interface

The PCI Power Interface will provide the power necessary to perform all other CryptoSwift HSM functions.

7.0 Components

7.1 Bulk Crypto

This component performs cryptographic hashing and symmetric cryptographic operations.

7.2 Power Management and Tamper Detect

This component monitors battery voltage and the security envelope to detect conditions that will result in the zeroization of critical security parameters. Battery voltage is also monitored to determine when it is necessary to replace the battery.

7.3 FastMap Processor

This component contains a processor and internal SRAM. The processor executes the software that initially resides in Flash memory and is eventually loaded into the external SRAM (external to the FastMap Processor yet still within the cryptographic boundary). The FastMap Processor also contains large accumulators and a random number generator. The accumulators are necessary for the acceleration of public key cryptographic operations. The random number generator generates truly random numb ers through a stochastic process. The output of this random number generator is used only for seeding the FIPS-approved ANSI X9.17 Appendix C pseudo-random number generator (PRNG). The output of the PRNG is used for generating 3DES and RSA keys, as well as outputting random numbers requested via the Generate Random Number service.

7.4 Flash

This component is non-volatile memory. The contents of Flash will maintain its state after PCI power and Battery power have been removed. The Flash contains the firmware that controls processing within the CryptoSwift HSM. It also contains public keys and other information that are not considered dangerous if exposed (certificates, public keys, encrypted data, encrypted keys and hash values used for authentication).

7.5 SRAM

SRAM is Static Random Access Memory. This memory will be used to store plaintext data, ciphertext data, symmetric keys, asymmetric keys, intermediate values, and firmware after it has been loaded from Flash.

7.6 Real Time Clock/Battery Powe red RAM (RTC/BBRAM)

This component is used to store values that are to be retained when PCI power is removed. This includes the master key (MK) that can be used to decrypt encrypted private keys and symmetric keys stored in Flash. The RTC is used to provide input to the key generation process so that it is consistent with FIPS 140-1 key generation requirements.

7.7 Programmable Logic Device (PLD)

This component embodies all additional logic necessary to interface components contained within the security envelope.

7.8 USB (Universal Serial Bus) Controller

This component allows the board to communicate with an iKey. The iKey is used to store a Personal Identification Number PIN that allows for user authentication, or to store key parts for moving keys from one HSM to another HSM.

7.9 Universal Asynchronous Receiver Transmitter (UART)

This component is disabled in the production version of the CryptoSwift HSM board.

7.10 33MHz Clock

This circuitry generates a square wave to provide the primary system clock and to synchronize the various components of the CryptoSwift HSM with the operation of the FastMap chip.

8.0 Definition of Security Relevant Data Items

The following are the security relevant data items contained in this module:

Master Key (MK) = The 3DES3KEY key which encrypts all non-volatile critical security parameters that are stored within the module (in the flash). The master key is stored in the BBRAM, and is destroyed when power is removed from both the PCI interface and the battery, and by the tamper detection circuitry whenever tampering is detected. The master key is randomly generated when the board is initialized (the Security Officer role is created).

Security Officer role PIN (SOPIN) = The SO role PIN is generated randomly when the board is initialized. It is written to an iKey token via the trusted USB interface. Please refer to section 9.2 below for a description of how this PIN is used for authentication.

User Role PIN (UserPIN) = The User Role PIN is generated randomly when the SO invokes the Create User service. It is written to an iKey token via the trusted USB interface. Please refer to section 9.2 below for a description of how this PIN is used for authentication..

Key-Wrapping-Key (KWK) = A 3DES3KEY key created by either the SO or User role for the purpose of wrapping private RSA keys. The Key-Wrapping-Key may be randomly generated using the Generate Key service, or may be entered into the module using the Combine Key service, which combines two key shares entered via the trusted USB interface. In the non-FIPS 140-1 mode, the Key-Wrapping-Key may also be created via the Derive Key service.

PRNG3DES Key (PRNGKey)= This 3DES2Key is used for seeding the X9.17 Pseudo-random Number Generator (PRNG). The PRNG 3DES Key is generated randomly using the hardware random number generator (RNG) within the FastMap processor. This key is generated every time a random number is needed for key generation or as a direct request via the Generate Random Number service. The PRNG 3DES EDE Key is destroyed after each PRNG is generated.

RSA Public and Private Key Pair (SPK, VPK)= This RSA key pair is generated by either the SO or User role for the purpose generating RSA digital signatures via the RSA Sign service, or for verifying the same via the RSA Verify service. A key pair which is designated by the user who created it cannot be used for any other purpose such as key exchanges or encryption/decryption of data. The user may specify via Boolean attributes whether the private key may be used for Signature Generation and/or Data Decryption, and whether the public key may be used for Signature Verification and/or Data Encryption. Hence, a given key pair may be used for both signatures/verifications as well as data encryption/decryption. In FIPS 140-1 Mode, data encryption/decryption is not available.

RSA Encryption/Decryption Public and Private Key Pair (EPK, DPK)= This key pair is generated by either the SO or User role for the purpose of encrypting and decrypting data. When creating this key pair, the user may specify via Boolean attributes whether the private key may be used for Signature Generation and/or Data Decryption, and whether the public key may be used for Signature Verification and/or Data Encryption. Hence, a given key pair may be used for both signatures/verifications as well as data encryption/decryption. Note that in the

FIPS 140-1 Mode, although Encryption/Decryption key pairs may be generated, the RSA Encrypt and RSA Decrypt services are not available ,and therefore, such keys are not useable in this mode.

Key-Wrapping-Key Share (KWKShare) = Key share obtained by splitting the KWK into two shares with the Split Key service. Two corresponding shares may be combined with the Combine Key service to enter the KWK into the module.

9.0 Roles & Services

9.1 Roles

The CryptoSwift HSM supports two roles. These are the User role and the Security Officer role.
Each role has a username and an iKey ID that are selectable by the security officer.

The module must be handled in a secure manner prior to initialization because authentication is not required to initialize the module.

Cryptographic keys and user-defined data which is created by a specific authenticated user cannot be deleted or modified by another user, regardless of the role. For example, a specific user of the User role may not delete or modify keys or data created by a different user of either the User or SO roles.

The SO and User roles cannot operate simultaneously. Only one authenticated user is allowed at a time.

9.1.1 User

The User role can perform cryptographic operations using private keys which are encrypted and stored in flash. The User role cannot create a user.

9.1.2 Security Officer

The Security Officer role can also perform cryptographic operations using private keys which are encrypted and stored in flash. Additionally, the Security Officer may create a user, update the HSM firmware, or command the CryptoSwift HSM to "uninitialize."

9.2 Authentication

The CryptoSwift HSM uses identity-based authentication to allow subjects to assume one of the two roles. Usernames are transmitted to the CryptoSwift HSM over the PCI interface to identify the user. A corresponding personal identification number (SOPIN or UserPIN as described in section 8.0 above) is inputted to the HSM from an iKey token over the trusted USB interface. This PIN is hashed and compared with a hash value which is stored in flash and associated with the user's name on the HSM. If the two hash values match, the user is authenticated and assigned a role that is associated with the user's name. To increase security in case the iKey token is compromised, an iKey ID is used to unlock the plaintext PIN that is stored in the iKey. This plaintext iKey ID is inputted into the module in plaintext as part of the Login service. The module provides a SHA-1 of this iKey ID to the iKey token in order to unlock the PIN. Since the iKey ID does not authenticate the user to the module, but rather unlocks the plaintext PIN from the iKey, the iKey ID is not an SRDI.

9.3 Initialization

The CryptoSwift HSM is shipped in an un-initialized state. At this point, it contains no private or secret keys. The Security Officer initializes the board. Performing this function generates an internally stored master key, and generates a random PIN, which is stored in the Security Officer's iKey token. Initialization also creates the Security Officer account and associates the SHA-1 hash of the random PIN with the Security Officer account.

9.4 User Creation

Once the board has been initialized, the Security Officer can create a User account. Creating the User account generates a random PIN, which is stored in the User's iKey token. The SHA-1 hash of this random PIN is associated with the User account.

9.5 Services

The following table describes which services can be performed by which role, and the SRDI(s) which each service accesses.

Service	FIPS140-1 Level 3 Mode			Non- FIPS140-1 Mode			
	Not authentica-ted	**User Role**	**SO Role**	**Not authentica-ted**	**User Role**	**SO Role**	**SRDIs Accessed**
Modular Exponentiation using CRT (note 3)	YES	YES	YES	YES	YES	YES	None
Modular Exponentiation (note 3)	YES	YES	YES	YES	YES	YES	None
RSA Encrypt (note 8)	NO	NO	NO	NO	YES	YES	EPK (use)
RSA Decrypt (note 8)	NO	NO	NO	NO	YES	YES	DPK (use)
Digital Signature Standard Sign (note 1)	NO	NO	NO	YES	YES	YES	None.
Digital Signature Standard Verification (note 1)	NO	NO	NO	YES	YES	YES	None
Self-test	YES	YES	YES	YES	YES	YES	None
Firmware Update	NO	NO	YES	NO	NO	YES	None.
Generate Random Number	YES	YES	YES	YES	YES	YES	PRNGKey (create, destroy)
Get Configuration	YES	YES	YES	YES	YES	YES	None.
Get Status	YES	YES	YES	YES	YES	YES	None.
Verify Firmware Image	NO	NO	YES	NO	NO	YES	None.

SHA1 Hash	NO	YES	YES	YES	YES	YES	None.
SHA1 HMAC (note 1)	NO	NO	NO	YES	YES	YES	None.
MD5 Hash	NO	NO	NO	YES	YES	YES	None.
MD5 HMAC (note 1)	NO	NO	NO	YES	YES	YES	None.
DES Encrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
DES Decrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
Triple DES Encrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
Triple DES Decrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
RC4 Encrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
RC4 Decrypt (note 1)	NO	NO	NO	YES	YES	YES	None.
Encrypt SHA1 Hash (DES) (note 1)	NO	NO	NO	YES	YES	YES	None.
Decrypt SHA1 Hash (DES) (note 1)	NO	NO	NO	YES	YES	YES	None.
Encrypt SHA1 Hash (3DES) (note 1)	NO	NO	NO	YES	YES	YES	None.
Decrypt SHA1 Hash (3DES) (note 1)	NO	NO	NO	YES	YES	YES	None.
Encrypt MD5 Hash (RC4) (note 1)	NO	NO	NO	YES	YES	YES	None.
Decrypt MD5 Hash (RC4) (note 1)	NO	NO	NO	YES	YES	YES	None.
Generate and Return RSA Key Pair (note 4)	NO	NO	NO	YES	YES	YES	None.
Generate and Store RSA Key Pair	NO	YES	YES	NO	YES	YES	PRNGKey (create and destroy), and create either or both of the following pairs: (SPK, VPK) or (EPK, DPK)
Store Public Object (Public RSA Key, user data object)	NO	YES	YES	NO	YES	YES	Enter and store: EPK or VPK

Store Vendor-Defined Data Object	YES	YES	YES	YES	YES	YES	None.
Store Private Object (Private RSA Key) (note 4)	NO	NO	NO	NO	YES	YES	Enter and Store: SPK or DPK
Get Public Object (RSA public key, user-defined data object)	NO	YES	YES	NO	YES	YES	Read: SPK or DPK
Get Vendor-Defined Data Object	YES	YES	YES	YES	YES	YES	None.
Get Object Information by Object ID	YES	YES	YES	YES	YES	YES	None.
Get Object Count	YES	YES	YES	YES	YES	YES	None.
Get Object Information by Index	YES	YES	YES	YES	YES	YES	None.
Get RSA Key Information by ID (modulus, exponent)	NO	YES	YES	NO	YES	YES	Read: VPK or EPK
Get RSA Key Information by Index (modulus, exponent)	NO	YES	YES	NO	YES	YES	Read: VPK or DPK
Change Object ID	NO	YES	YES	NO	YES	YES	None.
Delete Object	NO	YES	YES	NO	YES	YES	Destroy selected key: KWK, SPK, VPK, EPK, DPK.
Delete All Objects	NO	YES	YES	NO	YES	YES	Destroy all keys: KWK, SPK, VPK, EPK, DPK
Initialize Card	YES	NO	NO	YES	NO	NO	MK (create), SOPIN (create and write to trusted path)
Uninitialize Card (note 7)	NO	NO	YES	NO	NO	YES	Destroy all of the following: MK, SOPIN, UserPIN, KWK, SPK, VPK, EPK, DPK
User Login/Change PIN (note 5)	YES	NO	NO	YES	NO	NO	UserPIN (read from trusted interface)
Create User	NO	NO	YES	NO	NO	YES	UserPIN (create, write to trusted

							interface)
User Logout	NO	YES	YES	NO	YES	YES	None.
Derive Key (note 2)	NO	NO	NO	NO	NO	YES	KWK (create)
Wrap Key (note 4)	NO	YES	YES	NO	YES	YES	KWK (use), Wrap: SPK, DPK
Unwrap Key (note 4)	NO	YES	YES	NO	YES	YES	KWK (use), Unwrap: SPK, DPK
Modify Object	NO	YES	YES	NO	YES	YES	None.
RSA Sign (note 4)	NO	YES	YES	NO	YES	YES	SPK (use)
RSA Verify	NO	YES	YES	NO	YES	YES	VPK (use)
Generate Key (note 6)	NO	YES	YES	NO	YES	YES	KWK (create)
Split Key	NO	YES	YES	NO	YES	YES	KWK (split), PRNGKey (create, destroy), Two KWKShares (created and written to trusted interface)
Combine Key	NO	YES	YES	NO	YES	YES	KWK (created), two KWKShares (read from trusted interface)
Set LED State	YES	YES	YES	YES	YES	YES	None.

Note 1 = The key for these commands is inputted via the PCI bus (data input interface)
Note 2 = This is a PKCS 12 method for deriving a 3DES key from a password, salt and iteration count..
Note 3 = The Exponentiation Using CRT and Exponentiation functions are generic math functions; all parameters are inputted via the PCI interface (data input interface).

Note 4 = When operating in the FIPS140-1 mode, it is not possible for secret keys, private keys or critical security parameters to cross the PCI bus without being wrapped (encrypted) using the Key-Wrapping-Key.
Note 5 = User Login is the process that takes the board from an unauthenticated state to the authenticated state. Only one user may be authenticated at a particular time. Consequently, the User Login process cannot be started from the authenticated state. Nonetheless, the User Login process cannot be completed successfully without authentication.
Note 6 = This command is used for generating the key-wrapping-key.
Note 7 = When the board is in the zeroized state, it is possible to for an unauthenticated user to uninitialize the board.
Note 8 = These operations must access stored cryptographic keys. The keys may not be inutted ivia the PCI interface.

10.0 Key Management

10.1 Key Generation

Random number generation for key generation is accomplished using the algorithm described by appendix C of ANSI standard X9.17. This algorithm will use a seed value V (from appendix C) that is generated by the random number generator contained in the FastMap chip. Using this algorithm ensures that the keys generated will be consistent with the requirements of FIPS 140-1. Performing the key generation in this manner will ensure that the generated keys will be random and that the process used for their construction will be compatible with FIPS 140-1 requirements. Continuous random number testing is performed on the output of the hardware RNG (in the Fastmap chip) as well as on the output of the FIPS-approved ANSI X9.17 PRNG which is seeded by the RNG. For both continuous tests, the block size of 64 bits.

10.2 Key Storage

Private keys, symmetric keys and other critical security parameters will be stored in plaintext within the security envelope in RAM. Private and symmetric keys may also be stored in Flash, but only when first 3DES3KEY encrypted with the Master Key (MK) of the board. BBRAM is used to store the Master Key.

10.3 Key Entry and Output

When in the FIPS 140-1 mode, private keys and symmetric keys can only cross the cryptographic boundary when 3DES3KEY encrypted with a Key-Wrapping-Key. The Key-Wrapping-Key is generated when the "Generate Key" command is received by the HSM. The command that is used to encrypt and output a private or symmetric key is the "Wrap Key" command. The command that is used to enter and decrypt a private or symmetric key is the "Unwrap Key" command.

10.4 Key Distribution

To distribute a Key-Wrapping-Key between devices, it is split into two parts. The two parts, when exclusively ORed together, generate the Key-Wrapping-Key. The key splitting occurs when the "Write Key Split" command is first issued by the Security Officer. This command will cause one of the key parts to be written to an iKey controlled by the Security Officer. The second key part is written to an iKey controlled by the User. The Security Officer must logout and the User must login before the second "Write Key Split" can be performed. The two iKey tokens used for carrying key parts are labeled with the word "CODE".

The two key parts are then physically carried by separate trusted individuals to another device. If this device is also an HSM, the two parts may loaded into it using the "Read Key Split" command. Similarly, this command must be issued twice, once for the Security Officer and once for the User. Separate authentications are required for each "Read Key Split" command. After the second "Read Key Split" command has been successfully completed the destination device will contain the same Key-Wrapping-Key as the originating device.

Once two or more devices that contain the same Key-Wrapping-Key, they are said to be in the same family. Devices in the same family may share other secrets. Secrets are moved between devices under the control of a Rainbow Technologies key management utility. The key management utility runs on the host, and uses "Wrap Key" and "Unwrap" commands to move wrapped keys between devices in the same family.

10.5 Key Destruction

Critical security parameters including plaintext private keys, symmetric keys and intermediate values will be zeroized according to various conditions as described in figure 2. It is also possible for the security officer to command the board to un-initialize, which causes the data stored in RAM, FLASH and BBRAM to be erased.

Tamper Detected	Voltage Applied		Storage		
	Battery	PCI	BRAM	RAM and Other	Flash
NO	YES	YES	Retained	Retained	Retained
NO	YES	NO	Retained	Erased	Retained
NO	NO	YES	Retained	Retained	Retained
NO	NO	NO	Erased	Erased	Retained
YES	YES	YES	Erased	Erased	Retained
YES	YES	NO	Erased	Erased	Retained
YES	NO	YES	Erased	Erased	Retained
YES	NO	NO	Erased	Erased	Retained

Figure 2: Key Destruction

10.6 Key Archiving

Under the control of the Rainbow Technologies key management utility, it is also possible to archive keys. This may be done so that keys may be stored on backup media such as tape or hard drives.
The Rainbow Technologies key management utility utilizes the “Wrap Key” command to perform key archival. All archived keys are 3DES3KEY encrypted. Keys may only be archived and restored between devices in the same family.

11.0 Modes

The CryptoSwift HSM has two operating modes. These are the FIPS140-1 mode and the non-FIPS140-1 mode. Before the HSM is initialized with the "Initialize Card" command, it is in the non-FIPS140-1 mode. This command has an input parameter that specifies the mode of the card after initialization. Once initialized, the board remains in one of the two modes. If one wishes to change the operating mode of the card, the card must first be uninitialized using the "Uninitialize Card" command. Then, the card can be initialized with a different operating mode. Uninitializing the card removes all secrets from the card.

11.1 FIPS 140-1 Mode

In the FIPS 140-1 mode, the board may only perform FIPS approved algorithms.
These are as follows:

DES
3DES **
SHA-1
RSA Sign
RSA Verify

See the table in services section to identify the conditions necessary for performing various HSM commands in the FIPS140-1 mode.

No plaintext private or symmetric keys can cross the cryptographic boundary when the HSM is in the FIPS140-1 mode.

**The 3DES algorithm is used to secure private or symmetric keys stored in flash and for the key wrapping and unwrapping functions.

11.2 Non-FIPS 140-1 Mode

In the non-FIPS140-1 mode, the user has greater flexibility in the types of algorithms that can be performed and the manner that keys are handled. For example, in the non-FIPS140-1 mode, the board can perform all the functions of the FIPS140-1 mode plus other functions like MD5 and RC4. In the non-FIPS140-1 mode, keys may cross the cryptographic boundary in plaintext form for certain operations (e.g. DES, RSA CRT exponentiation). It is still possible to store keys on the board so that they cannot be extracted. These non-extractable keys will be erased if a tamper attempt is detected. See the table in services section to identify the conditions necessary for performing various HSM commands in the non-FIPS140-1 mode.

12.0 Self-Tests

The following table describes all of the cryptographic self-tests performed by the CryptoSwift HSM module. The following abbreviation is used:
KAT = Known Answer Test

Self-Test	FIPS 140-1 Mode	Non-FIPS 140-1 Mode	When performed
RSA Encrypt/Decrypt and Sign/Verify KATs	Yes	Yes	Power-up, Self-Test Service (on-demand)
DES KAT	Yes	Yes	Power-up, Self-Test Service (on-demand)
3DES KAT	Yes	Yes	Power-up, Self-Test Service (on-demand)
SHA-1 KAT	Yes	Yes	Power-up, Self-Test Service (on-demand)
DSA KAT	No	Yes	Power-up, Self-Test Service (on-demand)
MD5 KAT	No	Yes	Power-up, Self-Test Service (on-demand)
RC4 KAT	No	Yes	Power-up, Self-Test Service (on-demand)
RSA Key Generation Pairwise Consistency Test	Yes	Yes	Generate And Store RSA Key Pair Service , Generate And Return RSA Key Pair Service
Statistical Random Number Generator Tests (Monobit, Poker, Runs, Long Run)	Yes	Yes	Power-up, Self-Test Service (on-demand)
Continuous Random Number Generator Test	Yes	Yes	Whenever a pseudo-random number is generated: key generation, Generate Random Number Service
Firmware RSA Signature Verification Test	Yes	Yes	Power-up, Self-Test Service (on-demand), Firmware Update, Verify Firmware Image Service

13.0 Conclusion

The CryptoSwift HSM provides FIPS 140-1 Level 3 cryptographic processing, acceleration and security for RSA signing and verifying functions. In the non-FIPS140-1 mode, it can also bulk data cryptographic algorithms for PKI certificate server, firewall and web server equipment. It is suitable for use in applications requiring up to 200 public key transactions per second where protecting critical security parameters is a high priority. Industries requiring this high level of performance and security include (but are not limited to) banking, telecommunications, e-commerce, and medical services. In the area of self-test, the CryptoSwift HSM provides capabilities consistent with FIPS 140-1 Level 4.