Bluefly Processor

Security Policy

PRODUCT NAME:	**Bluefly Processor**
PROJECT NUMBER:	**MSW4000**
AUTHOR:	**Darren Krahn**
REVISION :	**1.12**
DOCUMENT REFERENCE :	**SP-MSW4000-01**
DOCUMENT TYPE:	**Security Policy**
DEPARTMENT:	**Secure Storage Products**
FIRMWARE VERSIONS :	**2.0, 2.1**
HARDWARE VERSIONS :	**3.0 (Part # 950 000 003 R)** **4.0 (Part # 950 000 004 R)**

Review and Approval

Larry Hamid	**CTO**		
Name	Title	Signature	Date
Marc Charbonneau	**Software Manager**		
Name	Title	Signature	Date
Name	Title	Signature	Date
Name	Title	Signature	Date
Name	Title	Signature	Date
Name	Title	Signature	Date
Name	Title	Signature	Date

The presence of the Directors name and date of approval certifies that the material contained within this revision of the document has been approved for release and will supersede all previous versions.

Representatives of appropriate Development Engineering directorates give this approval upon completion of a successful review.

1 Table of Contents

2 General

2.1 Revision History

Author	Date	Version	Description of Change
D. Krahn	Dec 15, 2008	1.0	Initial Draft
D. Krahn	Jan 21, 2009	1.1	Corrections
D. Krahn	Apr 21, 2009	1.2	Revised based on feedback.
D. Krahn	Jun 23, 2009	1.3	Revised based on feedback.
D. Krahn	Jun 26, 2009	1.4	Revised based on feedback.
D. Krahn	Jun 29, 2009	1.5	Revised based on feedback.
D. Krahn	Jul 2, 2009	1.6	Revised based on feedback.
D. Krahn	Jul 3, 2009	1.7	Revised based on feedback.
D. Krahn	Jul 6, 2009	1.8	Corrections.
D. Krahn	Feb 2, 2010	1.9	Revised based on CMVP comments.
D. Krahn	Feb 8, 2010	1.10	Revised based on feedback.
D. Krahn	Apr 13, 2010	1.11	Added FW and HW versions.
D. Krahn	Apr 27, 2010	1.12	Added algorithm validation numbers.

2.2 References

Reference	Title	Author
R1	FIPS 140-2	NIST

2.3 Terminology

Term	Definition
ASIC	Application Specific Integrated Circuit
Zero-Footprint Authentication	Authentication without running any host software.
UTF-8	8-bit Unicode Transformation Format. An 8-bit character encoding for Unicode.
CSP	Critical Security Parameter
PPSD	Personal Portable Security Device
USB	Universal Serial Bus

SATA	Serial Advanced Technology Attachment
SD	Secure Digital (card)
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver / Transmitter
LED	Light-Emitting Diode
HOTP	HMAC-based One Time Password
PRNG	Pseudo-Random Number Generator
TRNG	True Random Number Generator
TDES	Triple-DES
CRC	Cyclic Redundancy Check

3 Overview

3.1 Purpose

This document is the Cryptographic Module Security Policy for the Bluefly Processor. The purpose of a Security Policy is specified by FIPS 140-2 Appendix C.2

3.2 Scope

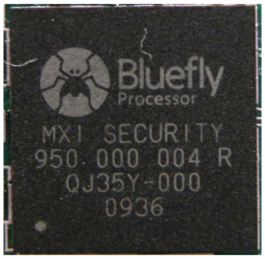
This document fulfills the requirements and expectations for a Cryptographic Module Security Policy specified by FIPS 140-2 Appendix C.

3.3 Target Audience

This document is intended for security reviewers as well as current and potential purchasers of products using the Bluefly Processor.

4 Introduction

The Bluefly Processor is a cryptographic and authentication engine for Personal Portable Security Devices (PPSDs). It provides secure storage, digital identity functions, and multifactor user authentication for USB-based peripherals. It is a single chip (ASIC) of dimensions 12 x 12 x 1.2 mm.



4.1 Security Levels

The following table lists the sections of FIPS 140-2 and targeted security level for each section.

FIPS 140-2 Security Requirement Section	***Target Level***
Cryptographic Module Specification	3
Cryptographic Module Ports and Interfaces	3
Roles, Services and Authentication	3
Finite State Model	3
Physical Security	3
Operational Environment	N/A
Cryptographic Key Management	3
EMI/EMC	3
Self-Tests	3
Design Assurance	3
Mitigation of Other Attacks	N/A

4.2 Cryptographic Algorithms

The following cryptographic algorithms are supported by the Bluefly Processor.

Algorithm	***FIPS Approved***	***Key Sizes (bits)***
AES (ECB, CBC, CFB-128, OFB, CTR, CCM, GCM)	Yes	128, 256
TDES (ECB, CBC, CFB-64, OFB, CTR)	Yes	56 x 3
SHA-1	Yes	N/A
SHA-224	Yes	N/A
SHA-256	Yes	N/A
SHA-384	Yes	N/A
SHA-512	Yes	N/A
MD5	No	N/A
HMAC (with any supported hash)	Yes (with Approved hash)	160+
CMAC-AES	Yes	128, 256
PRNG - X9.31 A.2.4 with AES	Yes	256
TRNG	No	N/A
RSA X9.31 Key Pair Generation	Yes	1024, 2048, 3072
RSA X9.31 Sign / Verify	Yes	1024, 2048, 3072
RSA PSS Sign / Verify	Yes	1024, 2048, 3072
RSA PKCS #1 v1.5 Sign / Verify	Yes	1024, 2048, 3072
RSA PKCS #1 v1.5 Encrypt / Decrypt	No	1024, 2048, 3072
RSA OAEP Encrypt / Decrypt	No	1024, 2048, 3072
RSA X.509 (raw) Encrypt / Decrypt	No	1024, 2048, 3072
DSA Key Pair Generation	No	1024, 2048, 3072
DSA Sign / Verify	Yes (1024-bit)	1024, 2048, 3072
dhEphem Key Agreement	Yes	L=2048, N=256

4.3 Algorithm Validation Numbers

The following table lists algorithm validation numbers for all FIPS Approved algorithms supported by the Bluefly Processor.

Algorithm	***Validation Numbers***
AES	1119, 1292, 1333, 1334
Triple DES	908, 932
SHA	1186, 1220
CMAC-AES	1292, 1334
HMAC	752, 782
PRNG - X9.31 A.2.4 with AES	720, 735
RSA	618, 646
DSA	417, 438
dhEphem Key Agreement	6, 7

5 FIPS Approved Mode of Operation

The Bluefly module can operate in two different and mutually exclusive modes: (1) Approved mode and (2) non-Approved mode. To operate the module securely the module must be kept in Approved mode. See section 6.2 for information on which services may be executed in Approved mode.

5.1 Entering and Leaving Approved Mode

The Bluefly module is not in Approved mode until it has been initialized. At minimum, a crypto officer must be enrolled in order for the module to be considered initialized.

Once initialized, the module always starts in Approved mode and does not leave Approved mode unless explicitly instructed to do so.

Note: There are two scenarios in which the module will leave Approved mode:

1. The operator utilizes the Set FIPS Mode service to instruct the module to leave Approved mode. The Set FIPS Mode service takes an input parameter which indicates whether to enter or leave Approved mode. When this operation completes the Approved mode indicator immediately conforms to the new operating mode.
2. The operator authenticates using an external authentication module (e.g. fingerprint) through the 'zero-footprint' authentication service. Because the strength of authentication of the external authentication module cannot be determined by the Bluefly module this method cannot be used on its own in Approved mode. When this authentication occurs the Approved mode indicator immediately changes to report that the module is not in Approved mode.

If the module is operating in the non-Approved mode, an operator may enter Approved mode by utilizing the Set FIPS Mode service or by powering down the module and powering back up. The Set FIPS Mode service takes an input parameter which indicates whether to enter or leave Approved mode.

5.2 Obtaining Approved Mode Indication

An operator may query the current mode of operation by utilizing the Get State service. This service responds with information about the current module state. Contained in this information is a Boolean value indicating whether or not the module is currently operating in Approved mode.

5.3 Approved Mode Enforcement

When the module is in Approved mode the following policies are enforced.

1. Only Approved services may be executed.

2. Cryptographic keys that may be used in non-Approved mode are not accessible. Such a key must be explicitly marked for use in a non-Approved mode and this marking cannot be removed.
3. Users that are capable of authenticating using a non-Approved authentication mechanism are not accessible. Such a user must be explicitly marked for use in this way and this marking cannot be removed.
4. Users that are not marked for non-Approved authentication (that is, they can authenticate in Approved mode) cannot configure a non-Approved authentication mechanism.

When the module is not in Approved mode the following policies are enforced.

1. Cryptographic keys that are not marked for use in non-Approved mode are not accessible.
2. Users that are not marked for non-Approved authentication (that is, they can authenticate in Approved mode) cannot configure a non-Approved authentication mechanism.
3. The module will not enter Approved mode while any user marked for non-Approved authentication is authenticated. The 'Lock' service must be executed for each session in this state before Approved mode can be entered.

5.3.1 Key and User Markings

A cryptographic key (OSYM, OPUB, OPRIV, OPUB_E, or OPRIV_D) may be marked by the operator at the time of creation for use in a non-Approved mode. This marking is an optional parameter that can be provided by the operator to the Inject Key and Generate Key services. It is stored with the key (per key) and can be queried by the operator using the Get Key Properties service. It cannot be modified after the initial creation.

A user may be marked by the operator for use with non-Approved authentication. This marking is a configuration attribute that is stored per user and can be set using the Set Configuration service. Once this attribute is set for a user, it cannot be reset. It can be queried by the operator using the Get Configuration service.

6 Ports and Interfaces

This section provides an overview of the ports and interfaces supported by the Bluefly Processor.

6.1 USB

This is the primary interface supported by the Bluefly and most of the control input, status output, and data transfers between the module and a host computer occur through this interface. The Bluefly registers with the USB host as a USB Mass Storage device.

6.2 Data Storage

The Bluefly supports three storage interfaces: SATA, SD and SPI. SPI is used only for retrieving and updating firmware while SATA or SD (not both) is used for the storage of other persistent data including user data and configuration data. During manufacturing each module will be configured with appropriate storage settings and these settings cannot be changed.

6.3 Status Pins

There are general purpose I/O pins that are used to communicate status output. It is intended that these pins are attached to LEDs that visually display the current status to the operator.

6.4 External Authentication Module

The Bluefly may be connected to an external authentication module via a serial port (UART). The external module provides control input as well as small amounts of data input / output to / from the Bluefly module.

7 Identification and Authentication Policy

All authentication mechanisms for the Bluefly module are identity based.

7.1 Initial Configuration

When a module has not yet been initialized (i.e. no operator has yet enrolled) the device is in a 'Not Initialized' state and allows administrative services to be used in order for the operator to enroll but no cryptographic services are available in this state.

7.2 Roles and Authentication

7.2.1 Crypto Officer Role

The Crypto Officer role is an administrative role. An operator in this role may configure module policies, create other users and crypto officers, and configure user policies for other users. A crypto officer has no access to the private data of other crypto officers or users.

7.2.2 User Role

The User role is a limited role for day-to-day use of the module. An operator in this role may use cryptographic services and access private data belonging to themselves but they have no access to private data belonging to other users and they may not modify policies.

7.2.3 Upgrade Role

The Upgrade role is a limited role which allows an operator to perform tasks like upgrading firmware (for future use, this does not affect currently running firmware), upgrading read-only data, or recycling the module.

7.2.4 Authentication Data

There are two types of authentication used by the Bluefly module: (1) Password, and (2) External. External authentication is provided by an external authentication module which may be a biometric module or some other means of authenticating a user. These two types of authentication may be combined for two-factor authentication. The Bluefly module cannot determine the strength of the authentication provided by the external authentication module. For this reason, the Bluefly will not allow a user (or crypto officer) to authenticate using only an external module while operating in Approved mode. External authentication actions may be performed in Approved mode but the information received from the external module cannot be considered authentication data. Furthermore, the Bluefly will not allow a user (or crypto officer) to authenticate using a password (or both password and an external module) while operating in Approved mode if the user is capable, or has at any time been capable, of authenticating using only an external module.

The following table lists all available roles and authentication data.

Role	*Authentication Type*	*Authentication Data*
Crypto Officer	Identity	Password OR (Password AND External).
User	Identity	Password OR (Password AND External).
Upgrade	Identity	Password.

7.3 Authentication Strength

The following table provides the strength for each authentication mechanism.

Authentication Mechanism	*Strength*
Password	At minimum, 4 UTF-8 characters are required. There are 1114112 possible UTF-8 characters. A failed password attempt causes the module to delay execution for at least 500ms. Thus, the probability of guessing the password in a single attempt is 1 in 169,560,726,483,748,519,936 and the probability of guessing the password in 1 minute is 1 in 1,413,006,054,031,237,666.
External	The strength of this authentication mechanism depends on the specific external module being used.
Password and External	This mechanism has, at minimum, the same strength as the 'Password' mechanism.

8 Access Control Policy

8.1 Critical Security Parameter Listing

The following table lists all critical security parameters (CSPs) and public keys used inside the Bluefly module. Each CSP / public key is given an identifier which may be used to identify it elsewhere in this document.

Note: The source of random numbers internal to the Bluefly is a pseudo-random number generator (PRNG) seeded with hardware generated entropy. There is one PRNG instance but two PRNG contexts (described under the SEED1 and SEED2 entries in the table below). Each context is used exclusively for its stated purpose; the operator cannot choose which context to use for a particular operation. When a symmetric key, asymmetric key, or other CSP is generated the PRNG with SEED1 is used to produce the key material and / or other required random parameters. When the operator executes the 'Generate Random' service, the PRNG with SEED2 is used to provide random data.

Note: The term 'injection' is used to describe the process by which an operator provides a key to the module (in contrast to the key being generated internally). Keys may be injected using the 'Inject Key' service. Also, a user-specific key (UK) may be injected using the 'Set Password' service. All key injection must occur via an encrypted channel; the module will reject any plain-text injection attempt.

The 'Details' column may include the following information:

- **Type** – The key type.
- **Size** – The key size.
- **Location** – Where the CSP resides. 'Internal' specifies that the CSP is stored inside the Bluefly chip. 'External' specifies that the CSP is encrypted and stored outside the chip. The key used for the encryption is specified along with reasoning for using the key.
- **Use** – How the CSP is used.
- **Source** – How the CSP is generated or injected into the module.
- **Output** – Under what circumstances the CSP is exported outside of the module.
- **Modification** – Under what circumstances the CSP is modified.
- **Zeroization** – Under what circumstances the CSP is destroyed. The term 'direct erasure' means the explicit erasure of the CSP from its storage location. The term 'indirect erasure' means the zeroization of the key used to encrypt the CSP.
- **Remarks** – Other information about this CSP.

CSP / Public Key	***Identifier***	***Details***

CSP / Public Key	***Identifier***	***Details***
Firmware Signing Public Key #1	FWS1	**Type**: RSA public key **Size**: 2048 bits **Location**: Internal **Use**: To verify firmware signatures. **Source**: Injected during manufacturing. **Output**: Get Configuration service. **Modification**: Roll Firmware Keys service. **Zeroization**: Direct erasure by the 'Zeroize' service.
Firmware Signing Public Key #2	FWS2	**Type**: RSA public key **Size**: 2048 bits **Location**: Internal **Use**: To verify firmware signatures. **Source**: Injected during manufacturing. **Output**: Get Configuration service. **Modification**: Roll Firmware Keys service. **Zeroization**: Direct erasure by the 'Zeroize' service.
Firmware Encryption Key #1	FWE1	**Type**: AES key **Size**: 256 bits **Location**: Internal **Use**: To decrypt firmware. **Source**: Injected during manufacturing. **Output**: Never. **Modification**: Never. **Zeroization**: Direct erasure by the 'Zeroize' service.
Firmware Encryption Key #2	FWE2	**Type**: AES key **Size**: 256 bits **Location**: Internal **Use**: To decrypt firmware. **Source**: Injected during manufacturing. **Output**: Never. **Modification**: Never. **Zeroization**: Direct erasure by the 'Zeroize' service.
Master Key	MK	**Type**: AES key **Size**: 256 bits **Location**: Internal **Use**: To encrypt keys and configuration for external storage. **Source**: Generated internally and programmed during manufacturing. **Output**: Never. **Modification**: Never. **Zeroization**: Direct erasure by the 'Zeroize' service.
Lifecycle Key	LK	**Type**: AES key **Size**: 256 bits **Location**: External (MK). MK is chosen because it resides inside the chip and is always available to the module. **Use**: To encrypt keys, configuration, and data for external

CSP / Public Key	Identifier	Details
		storage. Because MK cannot be modified and LK can, LK allows for a quick and cryptographically enforced recycle operation. **Source**: Generated. **Output**: Only for encrypted storage (see Location). **Modification**: Never. **Zeroization**: Direct erasure upon recycle.
Password-Based Value(s) #1	PBV1	**Type**: Binary value **Size**: 256 bits **Location**: Not stored. **Use**: This value is used only to obscure UK. **Source**: Derived from user's password using PKCS #5 PBKDF2 with HMAC-SHA-256. **Output**: Never. **Modification**: Change Password service. **Zeroization**: Not applicable (since the key is not stored). **Remarks**: This key differs from PBV2 only by the salt value that is passed as input to the key derivation function.
Password-Based Value(s) #2	PBV2	**Type**: Binary value **Size**: 256 bits **Location**: External (LK). LK is chosen because it is always available and is destroyed upon recycle. **Use**: This value is used only to perform password comparisons during authentication. **Source**: Derived from user's password using PKCS #5 PBKDF2 with HMAC-SHA-256. **Output**: Only for encrypted storage (see Location). **Modification**: Change Password service. **Zeroization**: Direct erasure upon recycle or user deletion. **Remarks**: This key differs from PBV1 only by the salt value that is passed as input to the key derivation function.
User Key(s)	UK	**Type**: AES key **Size**: 256 bits **Location**: External (LK). LK is chosen because it is always available to the firmware and is destroyed upon recycle. When password authentication is used, UK is first obscured with PBV1 and then LK. PBV1 is chosen so that UK is only accessible after user authentication. When an external authentication module is used, this key is encrypted with LK and then included in the payload sent to the external module. **Use**: Each user enrolled for authentication has one of these keys. It is used to encrypt private user keys and data. **Source**: Generated or optionally injected during a SetPassword operation. **Output**: Only for encrypted storage (see Location). **Modification**: A different key may be injected during a SetPassword operation but this will effectively cause all

CSP / Public Key	***Identifier***	***Details***
		user-specific keys and data to be destroyed. **Zeroization**: Direct erasure upon recycle, user deletion, or when the data destruction policy is set and the user exceeds their maximum failed authentication attempts.
Partition Key(s)	PK	**Type**: AES key **Size**: 256 bits **Location**: External (LK or UK). LK is chosen because it is always available to the firmware and is destroyed upon recycle. If the partition is a private partition each partition owner encrypts PK with their own UK instead of using LK. **Use**: Each partition has one of these keys. It is used only to encrypt the partition data on that specific partition. **Source**: Generated. **Output**: Only for encrypted storage (see Location). **Modification**: Never. **Zeroization**: Direct erasure upon partition deletion or when all partition owners are removed as owners. Indirect erasure upon recycle or when all partition owners have their UK destroyed.
Operator Symmetric Key(s)	OSYM	**Type**: AES or TDES keys **Size**: AES: 128, 256 bits; TDES: 3 x 56 bits **Location**: External (LK or UK). LK is chosen when the key is shared between all users because it is always available to the firmware and is destroyed upon recycle. UK is chosen when the key is private to a particular user because UK is only available once the user authenticates. **Use**: An operator may use these keys for cryptographic operations. **Source**: Generated (Generate Key service) or injected (Inject Key service). **Output**: Only for encrypted storage (see Location). **Modification**: Never. **Zeroization**: Direct erasure upon explicit deletion by the operator. Indirect erasure upon recycle or when the key is private and the owner's UK is destroyed.
Operator Asymmetric Key(s) for Signing	OPUB OPRIV	**Type**: RSA or DSA keys **Size**: 1024, 2048, or 3072 bits **Location**: External (LK or UK). LK is chosen when the key is shared between all users because it is always available to the firmware and is destroyed upon recycle. UK is chosen when the key is private to a particular user because UK is only available once the user authenticates. **Use**: An operator may use these keys for generating and verifying digital signatures. **Source**: Generated (Generate Key service) or injected (Inject Key service). An injected key may consist of a public key only or a complete key pair. **Output (OPUB)**: Get Public Key service.

CSP / Public Key	***Identifier***	***Details***
		Output (OPRIV): Only for encrypted storage (see Location). **Modification**: Never. **Zeroization**: Direct erasure upon explicit deletion by the operator. Indirect erasure upon recycle or when the key is private and the owner's UK is destroyed.
Operator Asymmetric Key(s) for Encryption	OPUB_E OPRIV_D	**Type**: RSA keys **Size**: 1024, 2048, or 3072 bits **Location**: External (LK or UK). LK is chosen when the key is shared between all users because it is always available to the firmware and is destroyed upon recycle. UK is chosen when the key is private to a particular user because UK is only available once the user authenticates. **Use**: An operator may use these keys for performing asymmetric encryption and decryption operations. **Source**: Generated (Generate Key service) or injected (Inject Key service). An injected key may consist of a public key only or the complete key pair. **Output (OPUB_E)**: Get Public Key service. **Output (OPRIV_D)**: Only for encrypted storage (see Location). **Modification**: Never. **Zeroization**: Direct erasure upon explicit deletion by the operator. Indirect erasure upon recycle or when the key is private and the owner's UK is destroyed.
Operator Password	PWD	**Location**: Not stored (volatile RAM only). **Use**: An operator provides a password to the module for authentication. **Source**: Always chosen by the operator and injected as part of an authentication operation. **Output**:Never. **Modification**: Not applicable (since it is not stored). **Zeroization**: This value resides in volatile RAM and will be physically destroyed when the module powers down. The RAM used to store this value is released to the system upon completion of authentication and may be reused. **Remarks**: PBV1 and PBV2 are derived from PWD.
PRNG Seed (Internal)	SEED1	**Size**: 512 bits **Location**: Not stored (volatile RAM only). **Use**: This value is generated upon power-up and is used as the PRNG seed and seed key. This PRNG context is used to generate cryptographic keys and CSPs. **Source**: Hardware generated entropy. **Output**: Never. **Modification**: The seed key portion is never modified. The seed portion is modified during every PRNG iteration using this context. **Zeroization**: This value resides in volatile RAM and will be

CSP / Public Key	*Identifier*	*Details*
		physically destroyed when the module powers down. **Remarks**: See the description of PRNG at the beginning of this section.
PRNG Seed (External)	SEED2	**Size**: 512 bits **Location**: Not stored (volatile RAM only). **Use**: This value is used as the PRNG seed and seed key upon power-up. This PRNG context is used to generate random numbers for the 'Generate Random' service. **Source**: Hardware generated entropy or provided by the operator. **Output**: Never. **Modification**: The seed key and seed portions are modified by the Set Random service. The seed portion is modified during every PRNG iteration using this context. **Zeroization**: This value resides in volatile RAM and will be physically destroyed when the module powers down. **Remarks**: See the description of PRNG at the beginning of this section.
Secure Channel Session Key(s)	SCSK	**Type**: AES key **Size**: 256 bits **Location**: Not stored (volatile RAM only). **Use**: This ephemeral key is the product of the key agreement scheme employed to negotiate a secure channel. This key encrypts / decrypts all communication sent / received via the secure channel. Each secure channel instance has a different key. **Source**: Key agreement scheme. **Output**:Never. **Modification**: Never. **Zeroization**: This key is zeroized when a secure channel is disconnected. Also, because this key resides in volatile RAM it is physically destroyed when the module powers down if the associated secure channel was never closed.

8.2 Service Listing

The following table lists all available services along with their class and access information. Service classes are used for categorizing services according to their access permissions. The following services classes are used:

1. Public – Services in this class are always accessible, even without authentication.
2. User – Services in this class require user or crypto officer authentication.
3. Administrative – Services in this class require crypto officer authentication or an uninitialized module.

4. Enrollment – Services in this class require user or crypto officer authentication or an uninitialized module. Users may only operate these services on themselves.

5. Upgrade – Services in this class require upgrade authentication.

For each service the access to CSPs is the same for all roles that are able to access the service. For example, both crypto officers and users may access the 'Encrypt' service providing 'Execute' access to 'OSYM' keys. This access to OSYM is the same for the crypto officer and the user.

Some of the services are self-explanatory but the following descriptions provide more information on those that are not:

- USB Mass Storage (Bypass) – This service allows operators to access public (not-encrypted) partitions via the USB Mass Storage standard interface. When available, the partition will be automatically mounted by most modern operating systems. There are three independent actions that need to occur in order to enable this bypass. 1) The partition must be assigned a logical unit number (LUN). 2) The partition must be configured to be read/write (not read-only). 3) The partition must be configured to public (not associated with a user).

- USB Mass Storage – This service allows operators to access private and read-only (encrypted) partitions via the USB Mass Storage standard interface. When available, the partition will be automatically mounted by most modern operating systems.

- Roll Firmware Keys – This service allows an operator to modify FWS1 or FWS2 provided that the operator is a crypto officer and that the request is digitally signed and can be verified using both the existing FWS1 and the existing FWS2.

- Raw Sector Access – This service allows an operator to read / write specific sectors within a partition independent of the USB Mass Storage service. This service gives access to partitions that are not currently available through the USB Mass Storage service. All applicable access restrictions are enforced for this service (e.g. an administrator may not use this service to read / write sectors of a user's private partition). This service may act as a bypass subject to the following two independent actions. 1) The operator must specify the 'Raw Sector Access' command identifier. 2) The operator must specify a flag indicating that encryption / decryption is not to be performed.

- Object Management – This service allows an operator to manage data objects in the various stores (read / write / delete). The available stores include a public store and a private store for each user. Objects in the public store are always accessible and are not encrypted. Managing objects in the public store is a bypass operation and is subject to the following two independent actions. 1) The operator must specify the 'Object Management' command identifier. 2) The operator must specify the public store identifier.

- Get Configuration – This service allows an operator to query public information from module including configuration, policy, and diagnostic information.

- Set Configuration – This service allows an operator to modify configuration and policy information in the module.

- Get State – This service allows an operator to query information about the current authentication and operating state of the module. This service allows an operator to query the Approved mode indicator.

- Lock – This service allows an operator to 'lock' or 'logout' an authenticated session.

- Secure Channel – This service allows an operator establish a secure session with the module. All communication within the secure session will be encrypted.

- Get Key Properties – This service allows an operator to query public information about cryptographic keys including size, type, and purpose.

- Online Notification – This service resets offline counter and timestamp information. It is intended to be used as part of an offline policy strategy.

- Seed Random – This service allows an operator to seed the PRNG used to generate random numbers for the 'Generate Random' service. It does not affect the PRNG that is used to generate cryptographic keys or other CSPs.

- Move Partition – This service moves a partition within external physical storage. It can be used for defragmentation when a module is managing multiple partitions.

- Generate One-Time-Password – This service generates HOTP compliant one-time passwords. It is not considered a cryptographic service and so neither its input parameters nor its output parameters are considered CSPs.

- Set Password – This service enrolls a password for a user. If a password already exists for the user, the operator must first authenticate (as the user) before this operation is allowed. A user key (UK) may be injected using this service; the injected value will overwrite any current value of UK for the user. If a password does not already exist or if UK is provided explicitly by the operator then the operator may be authenticated as an administrator instead of as the user. This service can only be executed in an encrypted session.

- Inject Key – This service allows an operator to provide the module with a key that is stored and can later be used for cryptographic operations. As with a generated key, an injected key can never be exported. This service can only be executed in an encrypted session.

- Recycle – This service returns a device to its initial state. It erases all users, objects, keys, and data partitions on the device. All configuration values are returned to factory defaults. The operator may choose (using a service parameter) whether to keep read-only and public partitions.

Public partitions and objects are deallocated and have all reference to them removed but their data, which is not encrypted, is not erased and will remain in external storage until it is overwritten through normal use.

- Zeroize – This service performs a permanent zeroization. In addition to the erasure performed by a 'Recycle' operation that keeps no partitions, this service zeroizes all values stored permanently inside the module. Once erased, these values cannot be written again.

Service	***Service Class***	***Keys / CSPs***	***Access***
USB Mass Storage (Bypass)	Public		
USB Mass Storage	User	PK	Execute
Roll Firmware Keys	Administrative	FWS1, or FWS2	Write
Set Password	Enrollment	PBV2	Write
Change Password	Enrollment	PBV2	Write
Inject Key	User	OSYM, OPUB, OPRIV, OPUB_E, or OPRIV_D	Write
Encrypt, Decrypt AES-ECB, AES-CBC, AES-CTR, AES-CFB, AES-OFB, AES-CCM, AES-GCM, TDES-ECB, TDES-CBC, TDES-CTR, TDES-CFB, TDES-OFB	User	OSYM	Execute
Sign (RSA, DSA)	User	OPRIV	Execute
Verify (RSA, DSA)	User	OPUB	Execute
Decrypt (RSA)*	User	OPRIV_D	Execute
Encrypt (RSA)*	User	OPUB_E	Execute
MAC (HMAC, CMAC)	User	OSYM	Execute
MAC Key (HMAC, CMAC)	User	(1) OSYM (2) OSYM, OPRIV, OPUB, OPRIV_D, or OPUB_E	(1) Execute (2) Hash
Hash Key (MD5*, SHA-x)	User	OSYM, OPRIV, OPUB, OPRIV_D, or OPUB_E	Hash
Generate Key [Pair]	User	SEED1 OSYM, OPUB, OPRIV, OPUB_E, OPUB_D	Execute Generate
Get Public Key	Public	OPUB or OPUB_E	Read
Raw Sector Access**	Public	PK	Execute
Object Management**	Public	LK, UK	Execute
Get Configuration	Public	FWS1, FWS2	Read
Get State	Public		

Service	***Service Class***	***Keys / CSPs***	***Access***
Authenticate	Public		
Authenticate (Zero-Footprint)	Public		
Lock	Public		
Self-Test	Public	FWS1, FWS2	Execute
Recycle	Upgrade	LK, PK, UK, OSYM, OPUB, OPRIV, OPUB_E, OPUB_D, PBV2	Zeroize
Reset	Public		
Partition Read-Only Mode	Upgrade		
Secure Channel	Public		
Delete Key	User	OSYM, OPUB, OPRIV, OPUB_E, or OPRIV_D	Zeroize
Get Key Properties	Public		
Online Notification**	Public		
Set FIPS Mode**	Public		
Generate Random	User		
Seed Random	User		
Hash Data	User		
Generate One-Time-Password	User		
Set Configuration	Administrative	PK, UK, PBV2	Zeroize
Move Partition	Administrative		
Update Firmware	Upgrade		
Enroll / Delete Fingerprint	Enrollment	UK	Zeroize
Zeroize	Upgrade	FWS1, FWS2, FWE1, FWE2, MK, LK, PK, UK, OSYM, OPUB, OPRIV, OPUB_E, OPUB_D, PBV2	Zeroize

* This service or algorithm is not Approved and so has no access to Approved keys or CSPs and cannot be executed in Approved mode.

** This service has more granular access checks based on configuration, state or service parameters but the service itself is ‘Public’.

8.3 Roles and Access Control

The following table lists the services available for each role. In addition to the roles, two other table rows are provided for informational purposes: 'None' indicates services that require no authentication and 'Not Initialized' indicates services that are available when no operator has yet enrolled.

Role	***Authorized Services***
Crypto Officer	Only services in the ‘Public’, ‘Enrollment’, ‘User’ or ‘Administrative’ classes.
User	Only services in the 'Public', 'Enrollment', or 'User' classes.
Upgrade	Only services in the ‘Upgrade’ class.
None	Only services in the 'Public' class.
Uninitialized	Only services in the 'Public', 'Enrollment', or 'Administrative' classes.

9 Physical Security Policy

The Bluefly Processor has a single-chip module embodiment and is targeted at Security Level 3.

9.1 Tamper Resistance

The Bluefly ASIC enclosure is tamper resistant. Any attempt to open or invade the enclosure will, with extremely high probability, permanently damage the module.

9.2 Inspection

Inspection by the operator is not required.

10 Self Tests

This section provides an overview of the self tests performed by the Bluefly Processor.

10.1 Power-up Self Tests

The following tests are performed on power-up:

- Firmware digital signature verification.
- Firmware version check.
- PRNG known answer test.
- AES known answer test (encryption and decryption).
- Triple-DES known answer test (encryption and decryption).
- HMAC known answer tests with all Approved hashes.
- CMAC known answer test.
- RSA signature generation and verification known answer tests.
- DSA signature generation and verification known answer tests.

10.2 Conditional Self Tests

The following tests are performed under specific conditions:

- Each PRNG and TRNG output block is compared to the previous output block to ensure they are not the same.
- RSA and DSA key pairs are tested after generation using a pair-wise consistency test.
- Diffie-Hellman public keys used during the negotiation of secure channels are tested for validity with the associated domain parameters.
- A firmware digital signature is verified when firmware is loaded from outside the module.

10.3 Bypass Self Tests

The following tests are performed to verify bypass operations.

- When the partition table controlling bypass operations is modified, a CRC is performed on the table to ensure integrity. The table can only be modified if the test passes.
- When the partition table controlling bypass operations is modified, the flags controlling the USB driver data path (encrypted / plain) for each logical unit are verified.

- Before the logic which controls bypass decisions for the Object Management service is executed the first time, the logic is exercised in a test mode to ensure the correct data paths will be followed in all cases.
- Before the logic which controls bypass decisions for the Raw Sector Access service is executed the first time, the logic is exercised in a test mode to ensure the correct data paths will be followed in all cases.

11 Mitigation of Other Attacks

The Bluefly does not protect against attacks involving power or timing analysis, fault induction, or electromagnetic emissions.