TITLE: Luna® Token Security Policies

ABSTRACT: This document describes the security policies implemented by the Luna® Token and how the design of its firmware and hardware enforces these policies.

DOCUMENT NUMBER: CR-1356

ORIGINATOR: Terry Fletcher

DEPARTMENT: Systems Engineering

LOCATION OF ISSUE: Ottawa

DATE ORIGINATED: October 29, 2001

CHANGE LEVEL: 4

CHANGE DATE: September 27, 2002

SECURITY LEVEL: None

SUPERSESSION DATA:

TABLE OF CONTENTS

LIST OF TABLES

1. INTRODUCTION

1.1. Purpose

This document describes the security policies implemented by the Luna® PC Card (*Luna® Token*) and how the design of its firmware and hardware enforces these policies.

1.2. Scope

This document addresses the Luna Token's security policies.

1.3. Application

This document describes the security policies that apply to the following products:

1. Luna® CA^3
2. Luna® RA^3
3. Luna® RA
4. Luna® 2
5. Luna® DSM

1.4. Intended Audience

The intended audience for this document is the Chrysalis-ITS Engineering and Product Management Team, external agencies for validation or endorsement of the Luna Token products; selected industry partners; prospective customers; and potential users of Luna Token products who want to understand the security policies of the products for FIPS operations.

The reader of this document should be familiar with the PKCS#11 standard defined by RSA Laboratories.

1.5. History of Revision

Revision	Date	Description
1	October 29, 2001	Combined CA3, Luna2 and Luna RA security policies.
2	February 6,2002	Removed former Appendix B (Policy Bit Settings) as obsolete. Minor wording changes to sections 4.2.1, 4.2.2 & 5.2
3	April 16, 2002	Extended security policy description to include Luna DSM
4	September 27, 2002	Add bold formatting to DES, 3DES wrapping mechanisms, revised description of fixed and configurable policies.

1.6. References

Document No.	Revision	Author	Title
PKCS#11	V2.10	RSA Laboratories	PKCS#11: Cryptographic Token Interface Standard, December 1999
FIPS PUB 140-1		Information Technology Laboratory, National Institute of Standards and Technology	FIPS PUB 140-1: Security Requirements For Cryptographic Modules, 11 January 1994.

1.7. Glossary of Acronyms / Abbreviations

Term	Explanation
CAV	Cryptographic Algorithm Vector
CCM	Custom Command Module
CRC	Cyclic Redundancy Check
CSP	Critical Security Parameters
DAC	Discretionary Access Control
FPV	Fixed Policy Vector
I&A	Identification and Authentication
PED	PIN Entry Device
PIN	Personal Identification Number
SO	Security Officer
SRAM	Static Random Access Memory
TPV	Token Policy Vector
UAV	User's Authorization Vector

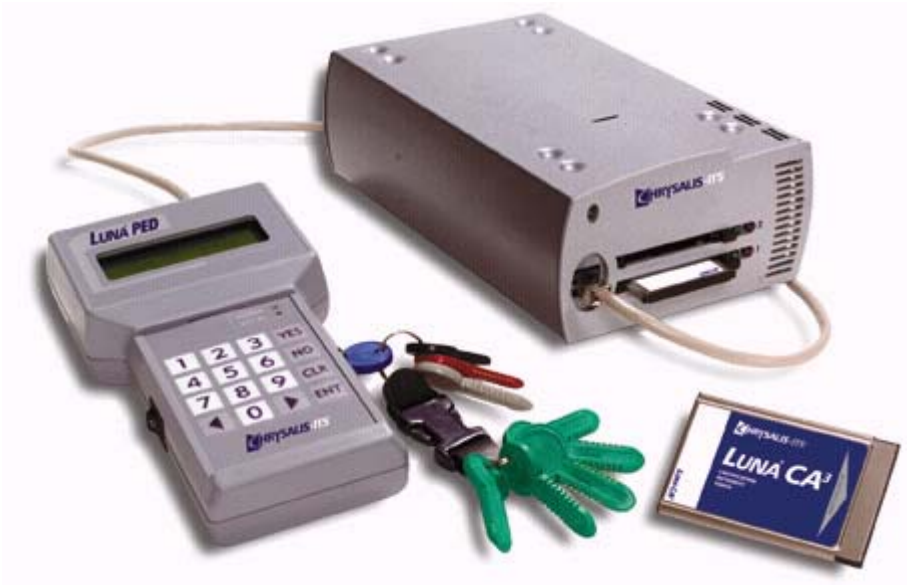
2. LUNA® TOKEN OVERVIEW

The Luna Token is a multi- chip standalone cryptographic device as defined in FIPS PUB 140-1. It securely stores data and keying material inside its cryptographic boundary. It also performs cryptographic operations on data provided by external applications using the keying material stored in the token. These abilities are defined as key management, object management, and cryptographic capability.

Before a Luna Token can be used to perform any cryptographic or key/object management functions, it must receive a valid operator identity (also known as a user number), as well as valid authentication data. These two inputs are processed by the token during a "LOGIN" command. When this command has completed successfully, the token allows the user to perform operations based on the policy settings defined for that token. For a Luna CA^3 or RA^3 token, the authentication data must be entered using a Luna PIN Entry Device (PED), which provides a trusted path to the token, separate from the host platform.

The token has the ability to distinguish between two categories of authenticated users: super-users and normal users. The super-user category is referred to as the Security Officer (SO) and the normal user category is referred to as the user. A token can have only one SO. The SO is allowed to perform all of the cryptographic, key and object management functions provided by the token, as well as a set of functions called the SO functions. These SO functions are available only to the SO, and they allow the SO to manage the token policy.

For a Luna Token, there is no limit on the number of users that can be created by the SO. All users are subjected to the same policy settings as those established by the SO. However, each user has his or her own authentication code initially assigned under control of the SO, which is used internally to protect the data the user owns.



The Luna CA^3 and RA^3 tokens protect critical security parameters in accordance with the requirements for FIPS 140-1 Level 3. In the case of these products, C*ritical Security Parameters (CSPs)* are defined to be the SO's and Token Users' authentication codes, cloning domain identifier and M of N secret shares. These CSPs can only be exchanged with a Luna CA^3 or RA^3 token through a data path separate from the host platform (also known as a trusted path). The trusted path is provided via a dedicated data port on the PC Card reader directly to the token. The user interface to this trusted path is a PIN Entry Device or *PED*. With the PED, a user can store a pseudo randomly generated authentication code, the cloning domain identifier or M of N shares on Datakey® serial memory keys (*Datakey device*). To enter any of these CSPs into the token, a user needs the token, a PED, a card reader capable of supporting communication between the token and the PED, **and** the Datakey devices containing the authentication data, cloning domain identifier or M of N shares.

The Luna 2, Luna RA and Luna DSM tokens meet the FIPS 140-1 Level 2 physical security requirements. The Luna CA^3 and RA^3 token meet the FIPS 140-1 Level 3 physical security requirements. The physical security policy is described in section 7.

3. SECURITY POLICY OVERVIEW

The security behaviour of the Luna® token is governed by the following security policies:

- Operational Policy
- Identification and Authentication Policy
- The Token Access Control (TAC) Policy
- Physical Security Policy

These policies complement each other to provide assurance that cryptographic material is securely managed throughout its life cycle and that access to other data and functions provided by the product is properly controlled. Configurable parameters that determine many of the variable aspects of a token's behaviour are specified by the higher level Operational Policy implemented through two sets of policy parameters: the Fixed Policy and the Configurable Policy. They are described in sections 4.1 and 4.2.

The Identification and Authentication policy is crucial for security enforcement and it is described in Section 5. The major security functional policy is the TAC policy. It is described in section 6,which also describes the supporting object re-use policy.

Security audit is not performed by the Luna products. It is assumed that audit is a function provided by the environment.

4. OPERATIONAL POLICY

The Luna® token employs the concept of Operational Policy Vectors to control the token's overall behaviour. The vectors control access to certain critical operations, such as token cloning, and establish security-critical information, such as the maximum number of failed login attempts. The Operational Policy is comprised of two sub-policies, each implemented through a policy vector: the Fixed Policy (Fixed Policy Vector) and the Configurable Policy (Token Policy Vector). The Fixed Policy is set as part of the manufacturing process and cannot be modified once set. The Configurable Policy is set by the Security Officer as part of token initialization and can be changed by a subsequent token initialization operation.

4.1. Fixed Policy

The fixed policy is determined by the settings of the Fixed Policy Vector (FPV). The FPV contains the settings necessary to enforce policy rules that apply across a wide range of possible token usage scenarios and environments. The settings are described in the following sections.

No Token User or SO can modify the FPV on a token. The FPV is put on the token when it is manufactured and remains in place until the token is destroyed or the firmware is erased. The integrity of the FPV is maintained through the same mechanism used to protect the executable code from being modified. This mechanism is a 32-bit Cyclic Redundancy Check (CRC) computation.

4.1.1. Number of SO Login Fails Allowed

This field defines the number of consecutive failed login attempts that can be made by the SO before the token erases the flash memory to prevent illegal access to its contents. This is set to three (3) for all Luna Tokens.

4.1.2. Secret Key Policy Bits - FPV

The following table defines the flags that identify the security policies that are followed for secret key objects.

Table 4-1 Secret Key Policy Bits - FPV

Name	Description	Setting				
		2	DSM	RA	CA[3]	RA[3]
FPV_SECRET_KEY_SENSITIVE	This bit determines whether a secret key object must always be made sensitive or if it can be determined by the high-level application using the token. When this bit is set, all secret keys stored on the token are sensitive. The keys are encrypted when in the flash memory and they can be extracted outside of the token only in encrypted form using the LUNA_WRAP_KEY function. This bit must be set for FIPS-compliant tokens.	1	1	1	1	1
FPV_SECRET_KEY_NO_CREATE	This bit determines whether a secret key object can be created by an external application using the LUNA_CREATE_OBJECT call, instead of being generated by the token. When this bit is set, an external application cannot create a secret key on the token; it is not possible to enter a secret key in plain text form on the token. This bit must be set for FIPS-compliant tokens.	1	1	1	1	1

4.1.3. Private Key Policy Bits - FPV

The following table defines the flags that identify the security policies that are followed for private key objects.

Table 4-2 Private Key Policy Bits – FPV

Name	Description	Setting				
		2		RA	CA[3]	RA[3]
FPV_PRIVATE_KEY_SENSITIVE	This bit determines whether a private key object must always be made sensitive or if it can be determined by the high-level application using the token. When this bit is set, all private keys stored on the token must be flagged as sensitive whether or not the high-level application requested this flag when the keys were created. When this bit is set, all private keys are encrypted while stored in flash memory. This bit must be set for FIPS-compliant tokens.	1	1	1	1	1
FPV_PRIVATE_KEY_NO_CREATE	This bit determines whether a private key object can be created by an external application using the LUNA_CREATE_OBJECT call, instead of being generated by the token. When this bit is set, an external application cannot create a private key on the token; it is not possible to enter a private key in plain text form on the token. This bit must be set for FIPS-compliant tokens.	1	1	1	1	1

4.1.4. Token Security Policy Bits - FPV

The following table defines the flags that identify the security policies that dictate the behavior of the token in general.

Table 4-3 Token Security Policy Bits - FPV

Name	Description	Setting				
		2	DSM	RA	CA[3]	RA[3]
FPV_DOMESTIC_FLAG	This bit determines whether the token can be exported. When this bit is set, the token is configured for the domestic market and cannot be exported. This bit is verified internally every time a cryptographic function implying an encryption or a decryption is performed. If the bit is set, no restrictions exist on key sizes. If the bit is not set, a limitation of 56 bits is applied to any symmetric keys used for encryption or decryption, and a 512-bit limitation on asymmetric keys used for wrapping and unwrapping operations. Signature and verification operations are not restricted in terms of key lengths.	1	1	1	1	1
FPV_ENABLE_CLONING	This bit determines whether sensitive objects on the token can be "cloned" to another similarly enabled token. When this bit is set, cloning is enabled. For a LunaRA token, this bit is tested in conjunction with the FPV_RA_TOKEN bit to restrict the cloning of objects with the private key attribute set (i.e., CKO_PRIVATE_KEY).	0	1	1	1	1
FPV_USE_CAV	This bit is used by the firmware to determine if the CAV should be checked to validate the desired algorithm. Normally, this bit is clear, which assumes all algorithms are valid.	0	0	0	0	0
FPV_WRAPPING_TOKEN	This bit determines whether RSA private keys can be wrapped. When this bit is set, an RSA private key can be wrapped. **Note:** For Luna 2 and Luna CA[3] tokens this setting is disabled to ensure that a private key cannot be extracted from the token even in encrypted format.	0	0	1	0	1
FPV_USE_M_OF_N	This bit defines whether the token can perform M of N activation. When this bit is set, the token can be configured to perform M of N activation.	0	0	0	1	1
FPV_USE_RAW_RSA	This bit determines whether RAW RSA operations can be performed on the token. When this bit is set, RAW RSA operations are allowed. RAW RSA provides access to RSA to perform encrypt and decrypt operations on data without any padding.	1	1	1	1	1
FPV_SPECIAL_CLONING	This bit determines whether the token allows the factory-default Chrysalis-ITS key cloning certificate to be replaced. When this bit is set, customers can create their own key cloning certificate.	0	1	1	1	1
FPV_ENABLE_CCM	This bit determines whether a Custom Command Module (CCM) can be loaded onto the token. When this bit is set, a CCM can be loaded onto the token. This bit must be cleared (i.e., zero) for FIPS-compliant tokens.	0	0	0	0	0
FPV_CCM_PRESENT_ FWUPDATE	This bit determines whether a CCM must be present before a firmware update operation is allowed to proceed. When this bit is set, a CCM must be loaded on the token to perform a firmware update. Additionally, the CCM must implement the PreModuleUpdate function. This bit is for an OEM version of Luna2 and does not apply to the Chrysalis-ITS	0	0	0	0	0

	labeled product.					
FPV_FORCE_RSA_BLINDING	This bit determines whether the token must perform blinding, which introduces a random element to the time needed to complete an RSA operation. Blinding defeats timing attacks on an RSA operation. If this bit is set, the token will always use RSA blinding (the TPV_FORCE_RSA_BLINDING bit will have no effect).	0	0	0	0	0
FPV_PIN_MUST_USE_SP	This bit determines if the serial communication port must be used to enter an authentication code. When this bit is set, an authentication code can only be entered through the serial communication port. When this bit is cleared, authentication codes are entered via the host computer.	0	0	0	1	1
FPV_MOFN_MUST_USE_SP	This bit determines if the serial communication port must be used to enter the M of N secret. When this bit is set, the M of N secret can only be entered through the serial communication port. When this bit is cleared, the M of N secret is entered via the host computer.	0	0	0	1	1
FPV_KCV_MUST_USE_SP	This bit determines if the serial communication port must be used to enter the key cloning domain identifier. When this bit is set, the key cloning domain identifier can only be entered through the serial communication port. When this bit is cleared, the key cloning domain identifier is entered via the host computer.	0	0	0	1	1
FPV_ALLOW_HA_RECOVERY	With this bit set, the High-availability recovery mode is enabled.	0	0	0	1	0

4.1.5. Product Identification Bits - FPV

The following table defines the flags that identify the configuration of the product.

Table 4-4 Product Identification Bits - FPV

Name	Description	Setting			
		2/DSM	RA	CA[3]	RA[3]
FPV_XP_TOKEN	This bit determines if the token is used in an XP style functionality, thus allowing the KCV to be set indirectly (allows XP tokens to get a CA[3]'s domain vector). This allows the token to be initialized as either a FIPS Level 2 or 3 device at Init Token time; all keys must be volatile.	0	0	0	0
FPV_LUNA_SSL_TOKEN		0	0	0	0
FPV_RA_TOKEN	This bit determines if the token has the RA functionality which includes private RSA key extraction. In addition to the cloning restriction of private key objects (see FPV_ENABLE_CLONING), special component key wrapping is associated with the LUNA_MECH_3DES_ECB mechanism.	0	1	0	1
FPV_XPPLUS_TOKEN	This bit indicates that the token is built upon XPplus-style hardware. XPplus hardware has: asymmetric math accelerators; extra RAM; non-volatile RAM; tamper detection mechanisms which trigger interrupts and wipe non-volatile RAM.	0	0	0	0

4.2. Configurable Policy

The configurable policy is determined by the settings of the Token Policy Vector (TPV). The TPV contains the settings necessary to enforce policy rules locally in an organization. For example, one bit in the TPV defines whether the token can perform a signature operation using a signing key generated by an outside process or if it must use an internally generated key for this function. The TPV can be modified by the token's SO. The TPV contents are used by the internal code to validate the operations performed by the token's user.

4.2.1. Number of User Login Fails Allowed

This field defines the number of consecutive failed login attempts that can be made by a user before the user gets locked out or the user's data is erased. This security feature prevents illegal access to the user's data and keys: it prevents an impostor from cracking the user's authentication code on the token. The default value, set at manufacturing time, is ten (10).

Whether the user is locked out or the data is erased depends upon the "User Zeroize" bit. If the User Zeroize bit is disabled, too many failed login attempts results in the User getting locked out. In this case, a user must make a request to the SO to regain access to the token. The SO also provides a new password for the user. If the User Zeroize bit is enabled, too many failed login attempts results in the User being deleted. In this case, the User's identity and private data (including key material) are erased from the token. The SO must create a new user in order to continue. The new user will have no association with the previous (deleted) user. The default setting of the User Zeroize bit is enabled.

4.2.2. Minimum/Maximum Authentication Code Length

These two fields define the minimum and maximum length restrictions for a USER's authentication data in a Level 2 validated token (e.g., Luna2 and Luna DSM). They have no effect on the length of the pseudo-random authentication data that is generated by Level 3 tokens (e.g., Luna CA^3) and stored on the PED Key; that length is fixed at forty-eight (48) bytes. For Level 3 tokens, the Security Officer may, when the user is created, require the user to enter a PIN value of up to sixteen (16) digits via the keypad on the PED in addition to the 48-byte secret generated by the token. The PIN value is combined, within the PED, with the token-generated secret to form the value that is stored on the PED Key for the user.

4.2.3. Local Policy Bits - TPV

The following table defines the flags that identify the security policies that dictate the behavior of the users on the token.

Table 4-5 Local Policy Bits - TPV

Name	Description	Default Setting			
		2/DSM	RA	CA^3	RA^3
TPV_USER_ZEROIZE	This bit determines whether the token can be zeroized by a normal user or if only the token's SO can zeroize the token. When this bit is set, it indicates that a valid token user can zeroize the token. When this bit is set, a user is zeroized after too many unsuccessful login attempts.	1	1	1	1
TPV_USER_FW_UPDATE	This bit determines whether the firmware can be updated by a normal user or if only the token's SO can update the firmware. When this bit is set, a normal user can perform the firmware update.	0	0	0	0
TPV_M_OF_N_ACTIVATION	This bit determines whether M of N activation is required for a user to gain access to the token. When this bit and the FPV_USE_M_OF_N bit in the FPV is set, the token is not activated until the required number of parts to a split secret have been entered.	0	0	0	0
TPV_KEY_ATTRIB_LOCK	This bit determines whether the flag attributes of a key can be modified once the key is a valid object on the token. When this bit is set, it indicates that the flag attributes of a key cannot be modified after they have been established. For example, if this bit is set and a DES key is created for encryption and decryption, these attributes cannot be changed to wrap and unwrap once the key exists on the token.	1	1	1	1
TPV_KEY_SINGLE_FUNCTION	This bit determines whether a key can be used to perform multiple types of operations (i.e., use a key for encrypting, signing, and wrapping). When this bit is set, it indicates that keys can be used only to perform single functions. For symmetric keys, a single function is considered to be a pair of related functions such as encryption/decryption, wrapping/unwrapping, or sign/verify	0	0	0	0
TPV_SIGNING_KEY_LOCAL	When performing a signing operation, the private key used may have been		0	0	0

	generated locally or provided by an external source. In most environments, it is preferable to have the signing/verifying key pair generated by the token and never extracted from it. However, in certain cases the signing keys are generated externally and loaded on the token for subsequent signature operations. When this bit is set, it indicates that externally generated keys cannot be used for signing operations performed by the token.	0			

4.2.4. Local Policy Bits – Extended TPV

The following table defines the flags that identify the security policies that dictate the behavior of the users on the token.

Table 4-6 Local Policy Bits – Extended TPV

Name	Description	Default Setting			
		2/DSM	RA	CA[3]	RA[3]
TPV_FORCE_RSA_BLINDING	This bit determines whether the token must perform blinding on RSA operations. If the FPV_FORCE_RSA_BLINDING bit is on, RSA blinding is performed on the token regardless of this TPV bit. However, if the FPV_FORCE_RSA_BLINDING bit is clear, the TPV_FORCE_RSA_BLINDING bit determines if the token will use RSA blinding. When the bit is set, blinding is performed.	1	1	1	1
TPV_DISABLE_CLONING_BY_USER	This bit determines whether a user or both a user and the token's SO are permitted to clone sensitive objects when the FPV_ENABLE_CLONING bit is set. If the FPV_ENABLE_CLONING bit is clear, no cloning is permitted and the TPV_DISABLE_CLONING_BY_USER bit has no effect regardless of its value. If the FPV_ENABLE_CLONING bit is set and the TPV_DISABLE_CLONING_BY_USER is clear, both a user and the token's SO are permitted to clone sensitive objects. If the FPV_ENABLE_CLONING bit is set and the TPV_DISABLE_CLONING_BY_USER is set, only the token's SO is permitted to clone sensitive objects.	0	0	0	0
TPV_XP_MUST_USE_SP	This bit determines whether the token implements XP-type functionality. When this bit is set: all objects are stored in volatile memory; token KCV may be cloned from a different token; the token is dual mode (i.e., whether the SP is required is defined at token initialization time); and, object headers are always modifiable, even in non-modifiable objects.	0	0	0	0
TPV_ALLOW_HA_RECOVERY	If enabled by the Fixed Policy Vector, with this bit set, High-availability recovery is allowed.	0	0	1	0

5. IDENTIFICATION AND AUTHENTICATION (I&A)

The Luna Token enforces an identity-based user authentication policy. Users are identified on the token by a user number, with the Security Officer having a special user number assigned. The Luna Token also supports three user roles: Public, Token User and Security Officer. Public users are unidentified and unauthenticated and may perform a limited set of functions, such as opening a session with a token and performing pre-defined diagnostics. A Public user cannot perform any cryptographic functions. The Security Officer (SO) is a privileged role whose primary purpose is to initially configure the token for operation and to perform security administration tasks such as user creation. The Token User is the normal operational role given to authenticated users on the token.

Note: Token users also have a text-based name associated with them. The name corresponding to a particular user number can be queried from the token.

5.1. Login

For a user to assume either the Token User or Security Officer role and perform cryptographic functions and other operations beyond those allowed for a Public user, the user must be identified and authenticated. For a Token User, the user number and valid authentication data (e.g., a password or the data stored on a Datakey device) must be provided to the token before access to private data and token services can be granted. For the SO, only valid authentication data is required. When M of N Activation has been enabled, as required by local security policy, no user can assume either the Token User or the Security Officer role before the M of N activation has been completed.

5.2. High Availability Recovery Support

Luna CA^3 provides an indirect login mechanism to support a capability for high availability recovery between two Luna CA^3 devices. It permits a backup Luna CA^3 device to share a common authentication state with the primary device within a given domain in order to allow automatic recovery of operations via the backup device in the event of a primary device failure.

5.3. Authentication Data and Trusted Path

Authentication data can be provided in multiple forms. For the Luna 2, Luna RA and Luna DSM tokens, it is in the form of a user-generated password or Personal Identification Number (PIN) that is normally entered via the keyboard of the host computer.

For the Luna® CA^3 and RA^3 tokens, the primary form is data that is randomly generated by the token, stored on a PED key and entered by the user for authentication. These tokens require that authentication data and M of N shares be entered using a trusted path separate from the host IT environment. In addition, when the user is initially created by the SO, the user may also be required to enter a separate user-generated authentication secret, of up to sixteen (16) digits, via the trusted path in addition to the randomly generated data.

Level 2 tokens enforce minimum and maximum lengths on the user-generated secrets based on the settings of the Configurable Policy. For Level 3 tokens the authentication data length is always forty-eight (48) bytes.

5.4. Limits on Login Failures

The Luna® tokens also enforce a maximum login attempts policy. The policy differs for an SO authentication data search and a Token User authentication data search.

In the case of a Token User PIN search:

- If “y” consecutive user logon attempts fail (“y” is defined by the SO in the Configurable Policy) the token flags the event in the user’s account data and handles the event as described in section 4.2.1.

In the case of an SO PIN search:

- If three (3) consecutive SO logon attempts fail, the token is zeroized. It must be re-initialized to return it to operation.

5.5. M of N Activation (Luna CA^3 and Luna RA^3)

Luna CA^3 supports a token activation feature called *M of N*. The concept of the M of N activation capability provides protection of a secret by "splitting" it into "N" pieces, where any "M" of these pieces must be reassembled to reconstruct the original secret. The Luna CA^3 feature is based on Shamir’s threshold scheme. This scheme allows a secret value to be shared by "n" external recipients without risking any compromise to the secret.

Initialization for M of N activation must be performed when there are no permanent sensitive objects stored on the token. Otherwise, there is a risk of corrupting objects stored in flash prior to generating the M of N set or after “deactivating” the feature. Management of M of N activation is an operational issue for the SO.

The M of N secret splits may be shared by more than one token within a domain. This capability must be invoked at the time of initialization. This same capability may be used to duplicate the M of N secret splits for backup purposes.

Note that for M of N activation to be effective, the values of M and N should be two or greater. Consider, for example, a 1 of 1 share to be of little value in securing activation of a token.

6. TOKEN ACCESS CONTROL (TAC)

The Token Access Control (TAC) policy applies to all objects on the Token, in particular to private key and secret key objects, and covers the following operations:

- Create
- Read
- Copy
- Modify
- Destroy
- Generate
- Derive
- Wrap
- Unwrap
- Use (encrypt, decrypt, sign, verify)
- Clone

The policy is summarized by the following statements:

A user may perform an allowed operation on an object if one of the following two conditions holds:

1. The object is a “Public” object, i.e., the PRIVATE attribute is FALSE, or
2. The user owns the object.

Allowed operations are those permitted by the Fixed and Configurable Policy settings and the values of the attributes shown in Table 6-1.

The token does not allow for any granularity of ownership other than that of private or public (i.e., a data object cannot be owned by two users and restricted from other users). Also, ownership of an object implies full access rights to the object and those access rights cannot be individually assigned by the owner to other users (e.g., User1, as the owner of an object, cannot give read access to it to User2). Table 6-1 lists object attributes used in making TAC policy enforcement decisions, the values each can have and the impact of each attribute value on policy enforcement. These attributes may only be set and/or modified, within the restrictions established by the Fixed and Configurable Policies, by the SO and Token User.

Table 6-1: Object Attributes Used in TAC Policy

Attribute	Values	Impact
PRIVATE	TRUE – Object is private to (owned by) the user identified as the Access Owner when the object is created	Object is only accessible to subjects (sessions) bound to the user identity that owns the object.
	FALSE – Object is not private to one user identity	Object is accessible to all subjects.
SENSITIVE	TRUE – Attribute values representing plaintext key material are not permitted to exist (value encrypted)	Key material is stored in encrypted form. For all FIPS-compliant products, this attribute is always TRUE.
	FALSE – Attribute values representing plaintext key material are permitted to exist	Plaintext key material is stored with the object and is accessible to all subjects otherwise permitted access to the object.
MODIFIABLE	TRUE – The object's attribute values may be modified	The object is "writeable" and its attribute values can be changed during a copy operation.
	FALSE – The object's values may not be modified	The object can only be read and only duplicate copies can be made.
EXTRACTABLE	TRUE – Key material stored with the object may be extracted from the token using the Wrap operation	The ability to extract a key permits sharing with other crypto modules and archiving of key material.
	FALSE – Key material stored with the object may not be extracted from the token	Keys must never leave the token. Private Keys in the CA^3 and Luna 2 are always treated as if this attribute is FALSE.

6.1.1. Object Re-use

The access control policy is supported by an object re-use policy. The object re-use policy requires that the resources allocated to an object be cleared of their information content before they are re-allocated to a different object.

7. PHYSICAL SECURITY POLICY

The Luna Token hardware shall be constructed in such a way that it will resist attempts to open the card or otherwise get at its internal circuitry and it will provide evidence of physical tampering by ensuring that any such attempt will leave the card in a state in which it is either inoperative or it is at least damaged to an extent that it cannot be restored to its previous condition.

8. NOTES

Datakey is a registered trademark of Datakey, Inc.

APPENDIX A. Cryptographic Algorithms Support

FIPS-approved algorithms are shown in bold lettering.

Encrypt/Decrypt:

- **DES-ECB**
- **DES-CBC**
- **3-DES-ECB**
- **3-DES-CBC**
- RC2-ECB
- RC2-CBC
- RC4
- RC5-ECB
- RC5-CBC
- CAST-ECB
- CAST-CBC
- CAST3-ECB
- CAST3-CBC
- CAST5-ECB
- CAST5-CBC
- RSA X-509

Digest:

- MD2
- MD5
- **SHA-1**

Sign/Verify:

- **RSA -1024**
- **RSA -2048**
- **DSA**
- **DES-MAC**
- **3-DES-MAC**
- RC2-MAC
- RC5-MAC
- CAST-MAC
- CAST3-MAC
- CAST5-MAC
- SSL3-MD5-MAC
- SSL3-SHA1-MAC
- HMAC-SHA1
- HMAC-MD5

Generate Key:

- **DES**
- **double length DES**
- **triple length DES**
- RC2
- RC4
- RC5
- CAST
- CAST3
- CAST5
- PBE-MD2-DES
- PBE-MD5-DES
- PBE-MD5-CAST
- PBE-MD5-CAST3
- PBE-SHA-1-CAST5
- GENERIC-SECRET
- SSL PRE-MASTER

Generate Key Pair:

- **RSA-1024**
- **RSA-2048**
- **DSA-1024**
- DH-1024

Wrap Symmetric Key Using Symmetric Algorithm:

- **DES-ECB**
- **3-DES-ECB**
- RC2-ECB
- CAST-ECB
- CAST3-ECB
- CAST5-ECB

Wrap Symmetric Key Using Asymmetric Algorithm:

- RSA-1024
- RSA-2048

Wrap Asymmetric Key Using Symmetric Algorithm:

- 3-DES-CBC[1]

Unwrap Symmetric Key With Symmetric Algorithm:

- DES-ECB
- 3-DES-ECB
- RC2-ECB
- CAST-ECB
- CAST3-ECB
- CAST5-ECB

Unwrap Symmetric Key With Asymmetric Algorithm:

- RSA-1024
- RSA-2048

Unwrap Asymmetric Key With Symmetric Algorithm:

- DES-CBC
- 3-DES-CBC
- CAST-CBC
- CAST3-CBC
- CAST5-CBC

Derive Key Value:

- DH-1024
- concatenate Base & Key
- concatenate Base & Data
- concatenate Data & Base
- XOR Base and Data
- Extract Key from Key
- MD2 Derivation
- MD5 Derivation
- SHA-1 Derivation
- SSL3-Master
- SSL3-Key & MAC

[1] Although this is a mechanism that is supported by the base firmware, the FPV settings for CA3 and XPPlus prevent wrapping of asymmetric private keys.

APPENDIX B. Session And Login States Required For Luna Commands

Command To Module	No Session Open	Session Open, No Login	SO Logged On	User Logged On
Token Main Module Commands				
LUNA_ZEROIZE	√			
LUNA_INIT_TOKEN			√	
LUNA_GET	√			
LUNA_GET_USV			√	
LUNA_SET_TPV			√	
LUNA_FW_UPDATE			√	
LUNA_CONFIGURE_SP	√			
Session Manager Commands				
LUNA_OPEN_ACCESS	√			
LUNA_CLEAN_ACCESS	√			
LUNA_CLOSE_ACCESS	√			
LUNA_GET_ALL_ACCESSES	√			
LUNA_OPEN_SESSION	√			
LUNA_CLOSE_SESSION		√		
LUNA_CLOSE_ALL_SESSIONS	√			
LUNA_GET_SESSION_INFO		√		
LUNA_EXTRACT_CONTEXTS		√		
LUNA_INSERT_CONTEXTS		√		
User Module Commands				
LUNA_GET_USER_LIST		√		
LUNA_GET_USER_NAME		√		
LUNA_LOGIN		√		
LUNA_LOGOUT				√
LUNA_SET_PIN				√
LUNA_INIT_PIN			√	
LUNA_CREATE_USER			√	
LUNA_DELETE_USER			√	
Object Management Module				
LUNA_CREATE_OBJECT		√		
LUNA_COPY_OBJECT		√		
LUNA_DESTROY_OBJECT		√		
LUNA_GET_OBJECT_SIZE		√		
LUNA_GET_ATTRIBUTE_VALUE		√		
LUNA_GET_ATTRIBUTE_SIZE		√		
LUNA_MODIFY_OBJECT		√		
LUNA_FIND_OBJECTS		√		
Random Number Generator Module				
LUNA_GET_RANDOM		√		
LUNA_SEED_RANDOM		√		
Key Management Module				
LUNA_GENERATE_KEY				√
LUNA_GENERATE_KEY_W_VALUE				√
LUNA_GENERATE_KEY_PAIR				√
LUNA_WRAP_KEY				√
LUNA_UNWRAP_KEY				√
LUNA_UNWRAP_KEY_W_VALUE				√
LUNA_DERIVE_KEY				√
LUNA_DERIVE_KEY_W_VALUE				√
LUNA_MFG_LOAD				√
Cryptographic Algorithm Module				
LUNA_ENCRYPT_INIT				√
LUNA_ENCRYPT_INIT_W_VALUE				√
LUNA_ENCRYPT_INIT				√
LUNA_ENCRYPT_INIT_W_VALUE				√

Command To Module	No Session Open	Session Open, No Login	SO Logged On	User Logged On
LUNA_ENCRYPT				√
LUNA_ENCRYPT_FIFO				√
LUNA_ENCRYPT_END				√
LUNA_DECRYPT_INIT				√
LUNA_DECRYPT_INIT_W_VALUE				√
LUNA_DECRYPT				√
LUNA_DECRYPT_FIFO				√
LUNA_DECRYPT_END				√
LUNA_DECRYPT_RAW_RSA				√
LUNA_DIGEST_INIT		√		
LUNA_DIGEST		√		
LUNA_DIGEST_FIFO		√		
LUNA_DIGEST_KEY				√
LUNA_DIGEST_KEY_VALUE				√
LUNA_DIGEST_END		√		
LUNA_SIGN_INIT				√
LUNA_SIGN_INIT_W_VALUE				√
LUNA_SIGN				√
LUNA_SIGN_FIFO				√
LUNA_SIGN_END				√
LUNA_SIGN_SINGLEPART				√
LUNA_SIGN_UPDATE_KEY				√
LUNA_SIGN_FINAL_DERIVE_KEY				√
LUNA_VERIFY_INIT				√
LUNA_VERIFY_INIT_W_VALUE				√
LUNA_VERIFY				√
LUNA_VERIFY_FIFO				√
LUNA_VERIFY_END				√
LUNA_VERIFY_SINGLEPART				√
LUNA_GET_MECH_LIST	√			
LUNA_GET_MECH_INFO	√			
LUNA_SELF_TEST	√			
LUNA_SET_UP_MASKING_KEY	√			
LUNA_CLONE_AS_SOURCE				√
LUNA_CLONE_AS_TARGET_INIT				√
LUNA_CLONE_AS_TARGET				√
LUNA_GEN_TKN_KEYS			√	
LUNA_LOAD_CERT			√	
LUNA_GEN_KCV				√
LUNA_LOAD_CUSTOMER_VERIFICATION_KEY			√	
LUNA_M_OF_N_GENERATE			√	
LUNA_M_OF_N_ACTIVATE				√
LUNA_M_OF_N_MODIFY			√	
Special Packet Processing Commands				
LUNA_IPSEC_INIT_NO_USER	√			
LUNA_IPSEC_PROCESS_PACKET	√			
LUNA_IPSEC_END	√			
LUNA_GEN_CRC32	√			
LUNA_SCP_TEST	√			