TITLE: Security Target for Luna® CA^3 Version 3.97

ABSTRACT: The Security Target documents the security functional and assurance requirements for the CC EAL 4+ evaluation of the CA^3 product.

DOCUMENT NUMBER: CR-0246

ORIGINATOR: Terry Fletcher

DEPARTMENT: Technology Group

LOCATION OF ISSUE: Ottawa

DATE ORIGINATED: 27 November 2000

CHANGE LEVEL: 12

CHANGE DATE: 01 November 2002

SECURITY LEVEL:

SUPERSESSION DATA: CR-0246, 11

Document Approvals			
Name	**Position**	**Date**	**Signature**
Terry Fletcher	Senior Security Engineer		

Revision History

Revision	Date	Description
Original Draft	14 July 2000	Initial version for review and comment.
1	18 August 2000	Incorporated changes and corrections from peer review
2	7 September	Revised security policies, updated TOE Summary Spec, added formatting for assignments, selections, iterations, refinements
3	16 October 2000	Updated TOE Summary Spec, references, rationale. Added requirement for separate authentication for key use.
4	31 October 2000	Updated ST to incorporate comments from external review. Removed requirement for separate authentication for key use.
5 V 2.00	20 November 2000	Completed updates. Aligned TSS with SFRs more explicitly. Amalgamated Security Policies into one. Completed Assurance Measures table.
6	6 December 2000	Minor revisions to incorporate review comments.
7 V3	7 February 2001	Revisions completed to address EORs.
8 V4	22 May 2001	Revisions based on comments from consultancy.
9 V5	19 June 2001	Revisions completed to address EORs.
10 V6	4 July 2001	Minor revisions to incorporate review comments.
11 V7	24 September 2001	Change version of CA^3 to 3.97. Removed “User Zeroize” from SO configurable parameters.
12 V8	6 March 2002	Minor wording changes to prepare for evaluation under Cdn scheme. Numbered individual requirements and updated refs accordingly, moved Table 4-1 to Table 8-1, added new Table 8-4 to show necessity of SFRs, added new section 5.2 for SARs.
13 V9	27 June 2002	Revisions to address ORs.
14 V10	16 September 2002	Revisions to address ORs.
15 V11	08 October 2002	Revisions to address ORs.
16 V12	01 November 2002	Removal of Company Confidential text.

TABLE OF CONTENTS

List of Figures

List of Tables

1 ST Introduction

1.1 ST Identification

Title: Security Target for Luna® CA^3 Firmware Version 3.97, Software Versions 8.0 and 8.1
Assurance level: EAL 4-augmented by ALC_FLR.2
Keywords: Commercial-off-the-shelf (COTS), hardware security module, certification authority, certificate issuing and management system, key management, cryptographic services, key generation, key protection, digital certificate management, public-key infrastructure, digital signature, encryption, confidentiality, integrity, networked information systems, baseline information protection.

1.2 ST Overview

The Luna® CA^3 is used to satisfy the most demanding requirements for trustworthy cryptographic processing within a Public Key Infrastructure. A Public Key Infrastructure (PKI) provides the means to ensure the integrity and authenticity of private and public keys used to communicate securely within an organization and between organizations. It consists of a number of critical components, such as a Certification Authority (CA), one or more Registration Authorities, possibly a Validation Authority or Online Certificate Status Protocol responder, high-assurance directory services and end-entity components for the performance of encryption and digital signature operations. Within a PKI, the CA is the most trusted component and is responsible for generating and signing the public key certificates and certificate revocation lists required to ensure the integrity and authenticity of private and public keys. As such, the CA's private signature key must be afforded the maximum protection possible. The Luna® CA^3 is normally used to provide such protection by acting as the hardware security module for the generation, protection and secure usage of the private key of a CA or Certification Service Provider (CSP) to sign public key certificates within a Public Key Infrastructure.

This Security Target (ST) describes the combination of security functionality and high-performance cryptography provided by the Luna® CA^3.

The Target of Evaluation (TOE) includes the following:

- two of the Chrysalis-ITS ® Luna® CA^3 devices, each in a PC Card form factor, (referred to as tokens), firmware version 3.97,
- a Chrysalis-ITS ® dual-slot Luna® Dock PC Card Reader,
- a Luna® Pin Entry Device (PED) and PED Keys,
- Enabler (product configuration) software versions 8.0 and 8.1, and
- Cryptographic API software versions 8.0 and 8.1.

The Luna® CA^3 provides functions for user identification and authentication, access control, data protection, data encryption, message digesting, digital signature, cryptographic material management and secure backup of mission-critical cryptographic material. Cryptographic operations are performed within the FIPS 140-1 (Level 3)-validated cryptographic module (Luna® CA^3 token).

The Luna® CA^3 is typically used in conjunction with a host computer and PKI application level software. Such elements with which the Luna® CA^3 interfaces are not included within the TOE and are not described within this Security Target. It may, however, function as one component in

a larger Certificate Issuing and Management System (CIMS) that is the target of a separate evaluation.

1.3 CC Conformance Claim

This TOE is conformant with:

1) CC Version 2.1 Part 2- extended. The following non Part 2 Security Functional Requirement is included to meet a specific requirement of the TOE:
- FDP_LUNA_BKP.1 (Luna Backup)

2) CC Version 2.1 Part 3-EAL 4 augmented. The EAL 4 package has been augmented by the addition of the following Part 3 requirement:
- ALC_FLR.2 (Flaw Reporting Procedures).

2 TOE Description

2.1 Overview

The TOE is depicted in the illustration at Figure 2-1.

Figure 2-1 Luna® CA[3]

The TOE provides a physically and logically protected component for the performance of cryptographic functions for key generation, key storage, encryption and decryption, digital signature and verification used by application systems that provide cryptographic support functions such as a Certificate Issuing and Management System (CIMS). It includes processors, read-only and random-access memory, and firmware packaged in a tamper-resistant form along with a small amount of host platform-specific communications support software.

The boundary of the TOE described in this ST encompasses the following:

1. Two identically configured printed circuit boards embedded in tamper resistant Type II PC Card carrier packages, known as Luna® CA[3] tokens. Each PC Card package hosts volatile and non-volatile memory, a microprocessor, with its associated firmware, data, control and key transfer signal paths, input/output controller, power management and a local oscillator.
2. A specially constructed PC Card reader, known as Luna® Dock.
3. The PIN Entry Device (PED), which is housed in a separate physical enclosure and, through a physically and electrically separate data port connection to the token, provides a trusted path for the entry of critical security parameters (authentication data and plaintext cryptographic parameters) to and from the token.
4. PED Keys, which are serial memory devices used to store critical security parameters for entry through the PED.
5. Enabler software, which provides the human user interface for initial product configuration and maintenance functions.

6. A PKCS #11 Cryptographic API, provided as a Windows DLL or Unix-type SO library depending on the host platform configuration, which provides the visible interface to the host application software application, which normally acts as the user of the Luna® CA^3.

The TOE in the evaluated configuration is supported for the Windows NT Version 4 (SP 5,6), Windows 2000, and the Sun Solaris 2.7 and 2.8 operating system platforms.

2.2 Architecture Model

The overall architectural model for the Luna® CA^3 is based on the PC Card and PKCS #11 device and object models. The model is depicted in Figure 2-2 below and is described in the subsequent sub-sections. Security policy enforcement is performed by the firmware on the token. The upper layers provide convenient external interfaces for software applications running on the host platform to access the security and cryptographic functions provided by the token and for human users responsible for product configuration and monitoring.

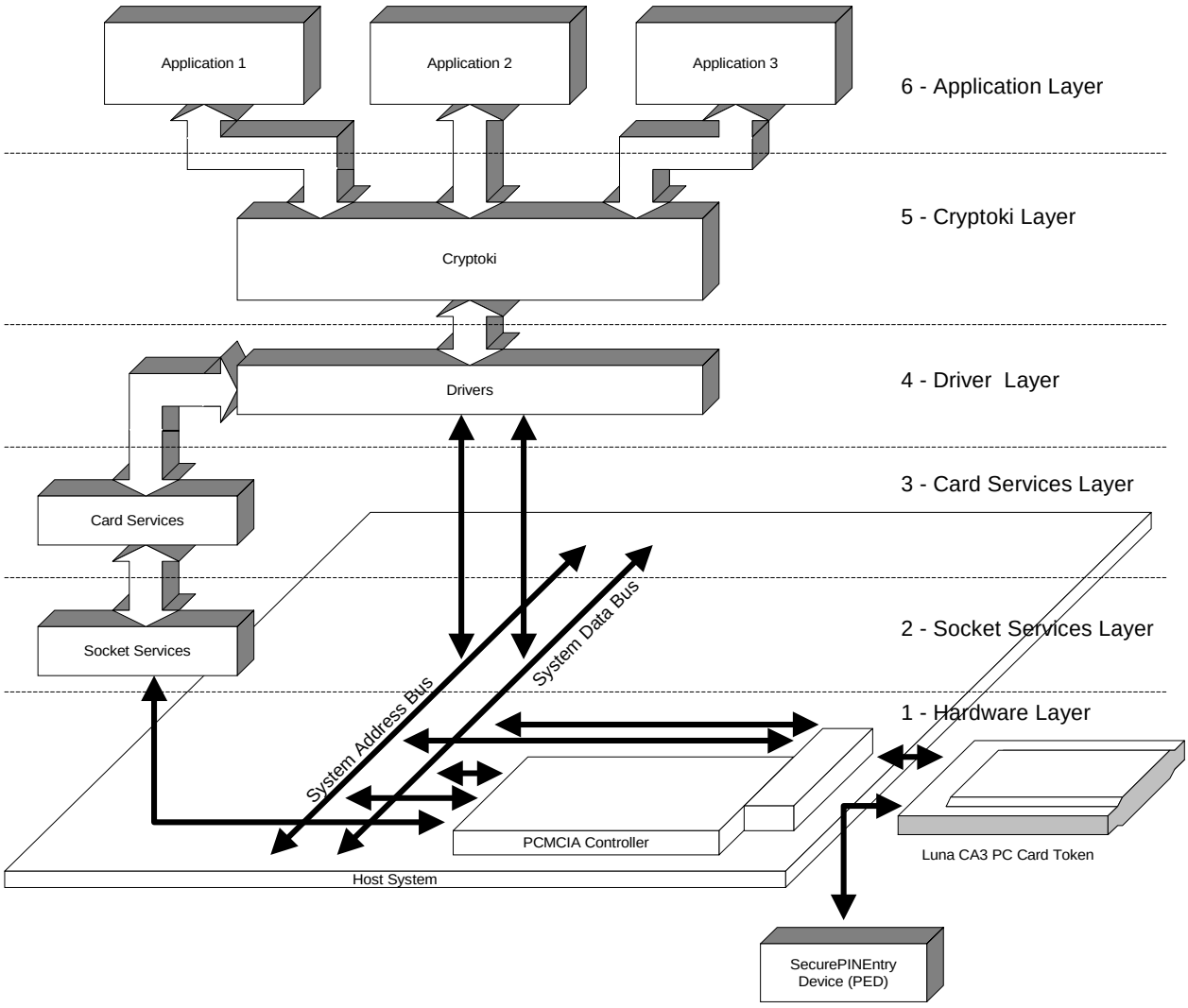

Figure 2-2 Luna® Architecture Model

2.2.1 Layer 1 - HW Layer

The physical layer is the Luna® CA^3 token with its firmware. The 68-pin PCMCIA connector as well as a generic PCMCIA controller are shown to indicate their roles of providing the connection paths to the token within the overall system architecture. The Luna® PED is used to communicate critical security parameters (e.g., PINs) to the token via the specially modified PCMCIA interface.

2.2.2 Layer 2 - Socket Services Layer

The socket services layer is defined in the PC Card standard as the hardware abstraction layer. It provides a software interface to the adapter that controls the PCMCIA socket. None of the Luna® software interacts directly with this layer.

2.2.3 Layer 3 - Card Services Layer

The card services layer is defined in the PC Card standard as the central resource manager for client software that needs to access PCMCIA card functions. The driver registers to this layer as a client. No sensitive data, in particular key material, is directly visible above this layer.

Note: In the case of the Windows NT/2000 and Solaris (UNIX) operating systems, Layer 2 and Layer 3 do not exist. For these operating systems the driver, Layer 4, talks directly to the PCMCIA Controller.

2.2.4 Layer 4 - Driver Layer

For each operating system there is a driver which coordinates the access to the token by the Cryptoki library. Each operating system provides a different mechanism by which the PCMCIA Controller is programmed to provide access to the token. The operating systems currently supported are: Windows 2000, Windows NT and Solaris 2.7/2.8.

2.2.5 Layer 5 - Cryptoki Layer

This layer consists of the Cryptoki library which implements PKCS#11 as a Dynamic Link Library (DLL) under Windows 2000 and Windows NT; and as a static and shared library under the Solaris environment. This layer acts as a translation layer between the PKCS#11 Application Programming Interface and the Luna® commands defined in the Luna® Interface Control Document (ICD).

2.2.6 Layer 6 - Application Layer

This layer represents applications such as PKI software, e-mail packages, word processors, spreadsheet software and databases. This layer allows applications to request that cryptographic functions be performed on their behalf. It is at this layer that the security application software, including the Enabler program, resides.

2.3 General Functionality Description

The Luna® CA^3 securely stores data and manages keying material inside the token boundary. The Luna® CA^3 also performs cryptographic operations on data provided by external applications using the keying material stored in the token. These capabilities are generally referred to as object management, key management, and cryptographic operations.

Before a Luna® CA^3 can be used to perform any cryptographic or key/object management functions, a token must receive a valid identity (also known as a user number), as well as valid authentication data. These two inputs are processed by a token during a LOGIN command. When this command has completed successfully, the token allows the user to perform operations based on the user's identity and the policy settings defined for that device.

In order to support security policies requiring strict separation of duties and/or multi-person control of critical security functions, the Luna® CA^3 implements a M of N secret sharing scheme, known as the Luna® M of N Activation Protocol [see section 6.2.1.1], to control token activation.

The token supports identity-based access control for data access. For purposes of security management, the token has the ability to distinguish three user roles: Security Officer (SO), Token User and Public. The three roles may also be referred to generically as the "user".

Users must be identified and authenticated to the token in order to have the SO or Token User role. The Token User role permits access to all functions except those restricted to the SO role. Users in the Public role do not require identification and authentication and are restricted to initiating token sessions and performing a limited number of token functions.

User authentication data and other critical security parameters must be communicated to the token via the PIN Entry Device (PED), which provides a trusted path directly to the token that is physically and electrically separate from all other data communications paths. To access the cryptographic material on a Luna® CA^3 token, a person needs the token, a PED, a specially constructed PC Card reader (Luna Dock) capable of supporting communications between the token and the PED, the PED Key containing the SO and/or Token User authentication codes and an optional password entered through the PED key pad.

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 privileged 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® CA^3, there is no limit on the number of users in the Token User role that can be created by the SO. All Token Users are subject to the same policy settings as established by the SO. Each Token User has its own authentication code initially assigned under the control of the SO, which is used internally to protect the data that the user owns and to support accountability.

The Luna® CA^3 supports access by only one logged in user at a time. The user having access to the device may establish any number of concurrent sessions, but the sessions cannot be shared with any other users. In practice, the Luna® CA^3 is operated as a single-user device.

In order to support backup and transparent recovery of the cryptographic keys and supporting data stored on a token, an optional token cloning function, based on the Luna® Key Cloning protocol, is available. Based on the local policy settings, the cloning function may be used to allow all of the objects to be securely copied from one token to the other within the same cryptographic domain. The token mounted in Slot 1 is the primary and must be mounted when the Luna® CA^3 is in operation. Depending on the local policy, a cloned token may be mounted in the reader or physically secured separately for backup and recovery in the event of disaster or catastrophic failure. Again depending on local policy, more than one cloned token backup can be maintained – one on-site backup, one locked in a safe and one stored at a secure recovery off-site location, for example.

2.3.1 Cryptography

The Luna® CA^3 provides the full range of cryptographic and key management functions. The major functions supported are the following:

1. generate public/private key pairs and secret (symmetric) keys;
2. encrypt (wrap) private keys and secret keys for export;
3. import and unwrap private keys and secret keys;
4. securely store key material;
5. securely dispose of key material;
6. encrypt data using supported symmetric and asymmetric algorithms; and,
7. compute digital signatures and verify digital signatures.

Handling of cryptographic keys and the use of cryptographic functions must be done in accordance with the key management procedures and policies of the user organization.

The Luna® CA^3 uses a dedicated cryptographic coprocessor and special purpose firmware that executes DES, triple DES, RSA and other public domain algorithms much faster than software implementations can.

2.3.2 Luna® Dock Card Reader

The Chrysalis-ITS Luna® Dock PC Card reader provides the interface between the Luna® CA^3 tokens and the host platform running the application software that uses the services of the Luna® CA^3.

The reader provides power to the Luna® CA^3 tokens. It also provides the physical/electrical interface to the input/output port connector that supports all communications to and from the tokens. In particular, the card reader provides the separate secure data port connection for the PED.

2.4 Environment

The Luna® CA^3 is designed to be operated in a physically secured environment by users who have been specifically authorized to do so by the owning organization. It is normally used within a larger Public Key Infrastructure (PKI) as part of a Certification Authority (CA) or Certificate Issuing and Management System (CIMS), or by a Validation Authority (VA). The environment will often, therefore, include a variety of hardware, software, telecommunications and networking devices as well as uninterruptible power supplies and environmental controls.

Because of its small size, the Luna® CA^3 token is susceptible to theft when it is outside its normal operating environment. Tokens must, therefore, provide for detection of attempts at physical or electrical probing and alteration that may be accomplished using personal computers and standard tools and laboratory equipment used without any supervision.

Note also that the TOE does not, in itself, provide any security audit functionality. If security audit is required for the system within which the TOE is operating, the host IT environment must provides the means of recording security relevant events, so as to assist an administrator in the detection of potential attacks or misconfiguration of the system, of which the TOE is a part, that could leave the host system and/or the TOE itself susceptible to attack, and also to hold users accountable for any actions they perform that are relevant to the security of the system.

3 TOE Security Environment

3.1 Secure Usage Assumptions

The specific conditions listed below are assumed to exist in the TOE environment.

3.1.1 Physical Security of the Initialization and Operating Environment Facilities

A.CONTROLLED_ACCESS – When in operation and when stored as a backup, the TOE is assumed to be located within a controlled access facility providing physical security that is adequate to prevent physical access by unauthorized persons.

3.1.2 Logical Security of the Host Operating Environment

A.LEGITIMATE_APPLICATIONS – It is assumed that the TOE is operated in conjunction with a host IT environment that has been checked to ensure only legitimate applications will have access to the token.

3.1.3 Environmental Conditions

A.POWER_INTERRUPTION_PROTECTION – It is assumed that the power supplied to the TOE is adequately protected against unexpected interruptions and the effects of surges and voltage fluctuations outside the normal operating range of the device.

A.DISASTER_PROTECTION – It is assumed that the TOE will be operated in an environment that is provided adequate protection against disasters such as fire and flood.

3.1.4 Personnel

A.ADMIN – When in operation, it is assumed that there will be a competent authority assigned to manage the TOE and the security of the information that it contains and who can be trusted not to deliberately abuse their privileges so as to undermine security.

3.1.5 Connectivity

A.HOST_CONNECTION – It is assumed that the TOE is directly connected to the host IT environment and that the connection points, cables and equipment are all contained with the controlled access facility.

3.1.6 Electromagnetic Radiation

A.EM_EMANATIONS – It is assumed that the TOE is located in an environment that is adequate to protect security-relevant and cryptographic key data from unauthorized disclosure due to electromagnetic emanations from the TOE.

A.EMI – It is assumed that the TOE is located in an environment that is adequate to protect security-relevant and cryptographic key data and TOE firmware from interference or inadvertent modification by strong electromagnetic radiation from other sources.

3.2 Threats

In general, the assets that the TOE is required to protect are the following:

1. Confidentiality of and access to user key material – private keys and secret keys.
2. Confidentiality, integrity and authenticity of user data.
3. Confidentiality of user authentication data.
4. Access to cryptographic functions.
5. Integrity of security functions.
6. Confidentiality and integrity of security function data.
7. Integrity of cryptographic material management processes.

The TOE must be capable of addressing threats that could result in compromise of these assets within different environments during the life cycle of the TOE. The environments associated with the phases of the TOE's life cycle are:

- Transit – This environment is concerned with threats associated with the transporting of the TOE between the manufacturer's premises and the customer organization.
- Initialization – This environment deals with the threats when setting up and the initializing process carried out on the TOE at the customer organization.
- Operations – The operations environment deals with the threats in the day to day running and management of the TOE, within the customer organization's production environment.
- Back-up and Storage; This environment relates to threats to a fully initialized token being used as a back-up token and the storage environment of the back-up token, and
- Disposal – Threats to the TOE and the organization during the disposal process.

Threat agents may include both unauthorized and authorized[1] persons acting out of deliberate intent or through errors and omissions. Unauthorized persons (or software entities) are assumed to have a low to moderate level of relevant expertise and a low level of access to required resources. Relevant expertise may be in general semiconductor technology, software engineering, hacking techniques, and the resources may range from personal computers and PC Card reading devices to general purpose test and measurement devices.

Motivation may include economic reward, a desire to damage an organization or the satisfaction and notoriety of defeating expert security.

Threat agents may also include malicious code of varying levels of sophistication, many of which are readily available on the Internet.

The threats will differ significantly with the environments in which the TOE is found and it must be capable of countering the full range of threats despite the fact that it may not be subject to many of them for a large proportion of its life-cycle. For example, the operations environment will normally be associated with a Certificate Issuing and Management System, which, because of its criticality in the infrastructure it supports will be located in a physically secure area and operated in accordance with strict security procedures. In this environment, the threats to the TOE are limited in scope and severity. The transit environment, on the other hand, is open to a much wider range of threats, including deliberate and systematic attempts to recover sensitive information, and the TOE must provide features to counter them.

[1] Authorized in this context means a person or software entity with organizational permission to perform the relevant operations on the TOE. Thus, a Public User requesting general status information regarding the token would be considered authorized, but a Public User attempting to access key material owned by a Token User would not be considered authorized.

3.2.1 Threats to User Identification and Authentication

T.Intercept_PIN – An unauthorized entity gains access to the authentication data of authorized users by intercepting the authentication data as it is entered.

T.Weak_PIN_Gen – An unauthorized person who has gained access to the TOE may attempt to compromise the TOE by guessing weak authentication data that was selected by an authorized person or that was generated by the TOE and permitted to exist.

T.Impersonate – An unauthorized person who has gained access to the TOE may gain access to security data or resources controlled by the TOE by impersonating an authorized user of the TOE.

3.2.2 Threats to User Data Access Control

T.Weak_Access_Control – An authorized user may gain access to and use sensitive data, such as private keys, stored on the TOE that are not owned by that user.

3.2.3 Threats to User Data Import and Export

T.Data_Export - An authorized user may export data from the TOE in a manner that is inconsistent with the TOE security policy.

T.Data_Import - An authorized user may import data to the TOE in a manner that is inconsistent with the TOE security policy.

3.2.4 Threats to Cryptographic Operations

No specific threats to cryptographic operations were identified.

3.2.5 Threats to Cryptographic Material Management

T.Key_Forced – An authorized person compromises the security of the cryptographic module by entering known cryptographic key data into the TOE.

T.Key_Read – An authorized person gains access to cryptographic key data by reading and recording cryptographic key data from the TOE in operation.

T.Weak_Key_Gen – An unauthorized person may compromise protected user data by exploiting weak cryptographic keys generated on the TOE.

T.Brute_Force – An unauthorized person attempts to gain access to plaintext cryptographic key data through direct memory access or brute force logical attacks.

3.2.6 Threats to Data Exchange

T.Exchange_Disclosure – An unauthorized entity may gain access to sensitive user data exchanged between the TOE and other IT entities.

T.Exchange_Modification – An unauthorized entity may cause undetected modifications of sensitive user data exchanged between the TOE and other IT entities.

T.Repudiate – A Certification Authority or other entity that uses the TOE may attempt to deny having performed a transaction such as certifying a subscriber's public key or issuing certificate revocation/status information for use by relying parties.

3.2.7 Threats to Control of TOE Functions

T.Improper_Operation – An administrator or operator may inadvertently place the TOE in operation before a secure operating environment[2] has been established or may inadvertently allow the TOE to be operated in a manner that is inconsistent with the local security policy.

3.2.8 Threats to Security Data Access Control

T.Disclosure_Physical – An unauthorized person who has gained access to the TOE may attempt to discover security data within the TOE by physical and electrical probing.

T.Weak_Policy_Settings – An unauthorized person who has gained access to the TOE may gain access to security data by exploiting policy settings for the TOE that produce conditions that could lead to the disclosure of TOE security data.

T.Unauth_Function – An unauthorized person who has gained access to the TOE may reveal, discover or modify security data within the TOE by exploiting unauthorized functions[3] of the TOE.

T.Interrupt – An uncontrolled interruption of the TOE initialization process may cause a disclosure of security data to an unauthorized entity by leaving residual security data present in accessible volatile and/or non-volatile memory.

3.2.9 Threats to Security Data Import and Export

T.Leak_Security_Data – Security-critical data entered or transferred through the host IT environment may be inadvertently communicated to an external entity resulting in a possible compromise of the TOE security functions

3.2.10 Threats to Logical Protection of the Security Functions

T.UA_Program_Load – An authorized person or an unauthorized person who has gained access to the TOE may modify the existing firmware and compromise the security functions of the TOE by loading unauthorized code.

T.Init_Data_Errors_Accidental - An authorized person accidentally places the TOE in an untrusted state or otherwise compromises the security functions of the TOE by making an error in inputting initialization data.

T.Init_Fail – A malfunction or other unexpected interruption during the initialization of the TOE may place the TOE in an untrusted state or otherwise compromise the security functions of the TOE.

3.2.11 Threats to Availability of Data and TOE Functions

T.Forget_PIN- An authorized person is unable to perform their required functions, causing a disruption in the operations of the organization operating the TOE, by failing to remember their PIN authentication code.

T.Failure – The failure of an individual token may cause a disruption in the operations of the organization operating the TOE.

---

[2] Establishing a secure operating environment includes ensuring all organizational security policies are met, all users are appropriately trained, all required initialization has taken place and that the host platform is correctly configured.
[3] An unauthorized function is any function that is not supposed to be executed by the TOE. In this case, the concern is mainly with an attacker trying to load unauthorized firmware on the device. The TOE protects itself by enforcing the loading of authorized firmware only, and protects against attempts to subvert its existing fuctions.

4 Security Objectives

There are two types of security objectives, which must be satisfied - the security objectives for the TOE, and the security objectives for the environment.

4.1 Security Objectives for the TOE

4.1.1 Objectives for User Identification and Authentication

O.I&A - The TOE must uniquely identify all users, and must authenticate the claimed identity before granting a user access to the TOE facilities.

O.Auth_Data_Protect – The TOE must protect authentication data in a way that ensures that user authentication data cannot be easily guessed.

O.TrustedPath - The TOE must provide a trusted path for entry of authentication data and other critical security parameters in which both endpoints have assured identities.

O.Login_Limit - The TOE must have the ability to limit the number of incorrect login attempts and to carry out actions commensurate with the level of authority associated with the role attacked.

O.Restrict_Actions_Before_Authentication – The TOE must restrict the actions a user may perform before the TOE verifies the identity of the user to entry of the user's I&A data and diagnostic actions.

4.1.2 Objectives for User Data Access Control

O.Access - The TOE must control access to mediated commands, information, and administrative privileges based on a user's identification, the requested access and the security attributes associated with the object to which access is requested.

4.1.3 Objectives for User Data Transfer, Import and Export

O.User_Data_Protect – The TOE must ensure the confidentiality of user data stored and transferred internally within the TOE.

4.1.4 Objectives for Cryptographic Operations

O.Random_Generate - The TOE must generate pseudo-random numbers using an approved algorithm and an initial seed from a random source.

O.Crypto_Algorithms - The TOE must implement approved cryptographic algorithms for encryption/decryption, authentication, and signature generation/verification.

4.1.5 Objectives for Cryptographic Material Management

O.Key_Generate - The TOE must generate symmetric keys and asymmetric key pairs using approved algorithms and validated random number generator.

O.Key_Destroy – The TOE must destroy cryptographic key material when requested by an authorized user.

O.Key_Zeroize – The TOE must completely erase all cryptographic key data from its memory content in response to a logical brute force attack by an unauthorized person.

4.1.6 Objectives for Data Exchange

O.Exchange_Confidentiality – The TOE must provide the capability to protect data exchanged with other IT entities from unauthorized disclosure.

O.Exchange_Integrity – The TOE must provide the capability to protect data exchanged with other IT entities from unauthorized modification.

O.Data_Authenticity - The TOE must provide the capability to guarantee the identity of the originator of exchanged data.

4.1.7 Objectives for Control of TOE Functions

O.Access – See section 4.1.2.

O.Admin - The TOE must provide facilities to enable the Security Officer (SO) to effectively manage the TOE and its security functions, and must ensure that only authorized users are able to access such functionality.

O.Limitation_of_Privilege – The TOE must limit the access privileges of the SO and other users to permit access to only those functions, resources and data objects for which the user requires access in order to perform his/her duties in such a way as to ensure the security of the TOE's functions and data and sensitive user data.

O.Multi-Person_Control_of_Sensitive_Functions - The TOE must provide a capability for multi-person control of sensitive functions with authentication based on split knowledge techniques for secret sharing.

O.Security_Roles – The TOE must provide the capability to maintain the Public, Token User and Security Officer roles and the association of identified users with those roles.

4.1.8 Objectives for Security Data Access Control

O.Object_Sec_Attributes - The TOE must store and preserve the integrity of a set of security attributes for data objects it stores and processes.

O.User_Sec_Attributes – The TOE must store and preserve the integrity of a set of security attributes associated with individual users.

O.Security_Data_Protect – The TOE must ensure the confidentiality of security data stored and transferred internally within the TOE.

4.1.9 Objectives for Logical Protection of the Security Functions

O.Secure_Init - The TOE must assume its initial secure state immediately upon power-up, reset, or after other restart conditions.

O.Self_Protect - The TOE must protect itself against attempts to logically subvert or bypass the TOE security functions.

O.Import_Code – The TOE must prevent executable firmware code from being loaded on the TOE unless it is encrypted and signed as per an authorized firmware update.

O.Inadvertent - The TOE must protect itself against inadvertent errors placing the TOE into an untrusted condition.

4.1.10 Objectives for Physical Protection of the Security Functions

O.Tamper_Evidence - The TOE must be constructed in a manner that provides evidence of physical tampering and attempts at probing.

4.1.11 Objectives for Availability of Data and TOE Functions

O.Alt_Authentication – The TOE must provide alternatives to authentication based only on data the user knows.

O.Backup - The TOE must have the ability to back up keys in a manner that protects their confidentiality and integrity.

4.2 Security Objectives for the Environment

4.2.1 Security Objectives for the non-IT Environment

OE.Authdata - Those responsible for the TOE must ensure that the authentication data for each user account for the TOE is held securely and not disclosed to persons not authorized to use that account.

OE.Admin - Those responsible for the TOE must ensure that the TOE is installed, managed, and operated in a manner that maintains IT security.

OE.Legitimate_Applications – The host IT environment must be configured and checked to ensure that any applications installed in the host environment that require access to the token are legitimate.

OE.Physical - Those responsible for the TOE must ensure that those parts of the TOE that are critical to security policy enforcement are protected from unauthorized physical access.

OE.Recovery - Those responsible for the TOE must ensure that procedures are in place to ensure that, after system failure or other discontinuity, recovery without compromise of IT security is obtained.

OE.Competence – Those responsible for the TOE must ensure that Security Officers and users of the TOE have a level of competence sufficient to ensure its correct management and operation. This competence may be established through a combination of training and the TOE guidance documentation.

OE.EM_Emanations - Procedural and physical measures must be taken to prevent the disclosure of cryptography related IT assets to unauthorized individuals or users via the electromagnetic emanations of the TOE.

OE.Connect - Those responsible for the host IT environment must ensure that no connections are provided to outside systems or users that would undermine IT security.

OE.Power_Interruption_Protection – Those responsible for the host IT environment must ensure that the power supplied to the TOE is adequately protected against unexpected interruptions and the effects of surges and voltage fluctuations outside the normal operating range of the device.

OE.Disaster_Protection – Those responsible for the host IT environment must ensure that the TOE is operated in an environment that is provided adequate protection against disasters such as fire and flood.

OE.EMI – Those responsible for the host IT environment must ensure that the TOE is located in an environment that is adequate to protect security-relevant and cryptographic key data and TOE

firmware from interference or inadvertent modification by strong electromagnetic radiation from other sources.

4.3 Mapping of Objectives

The correspondence or mapping showing the necessity of the security objectives to counter the threats is shown in Table 8-1. Tables relating assumptions and threats to objectives as well as the rationale for the sufficiency of the objectives is presented in Table 8-2 and Table 8-3.

5 IT Security Requirements

5.1 Security Functional Requirements

The following sections detail the Security Functional Requirements for the TOE. The summary table below plus separate tables in each section provide a mapping of IT Security Requirements to specific security functional components from ISO 15408-2 plus one explicitly stated requirements in the area of TOE backup.

The following convention is used to indicate operations that have been performed on the CC functional components:

- Assignment and selection are indicated by **bold** lettering.
- Refinement is indicated by *italic* lettering.
- Iterations are indicated by supplementary bracketed information with the functional component, such as **FIA_AFL.1.1 (SO) and FIA_AFL.1.1 (Token User).**

Table 5-1: Summary of Security Functional Requirements

Security Requirement		CC Functional Component	CC Requirement Title
User account data	Controls over creation of user accounts and modifications to user account status.	FIA_ATD.1	User Attribute Definition
User Identification and Authentication	Identification of users	FIA_UID. 1	Timing of identification
	Authentication of users	FIA_UAU.1	Timing of authentication
	Multiple Authentication Mechanisms	FIA_UAU.5	Multiple authentication mechanisms
	Limits on repeated login failures (e.g. enforcement of lockout or time delay)	FIA_AFL.1 (SO) FIA_AFL.1 (Token User)	Authentication failure handling
	User-Token Session Binding	FIA_USB.1	User-subject binding
Trusted Path	Trusted path for logon	FTP_TRP.1	Trusted path
Authentication data selection	Controls on selection of user-generated Passwords (e.g. minimum length)	FIA_SOS.1	Verification of secrets
	Automated generation of passwords by TOE	FIA_SOS.2	TSF generation of secrets
Token Access Control Policy	Scope of policy (subjects, objects and operations covered by the policy)	FDP_ACC.1	Subset access control
	Rules governing access by subjects to objects	FDP_ACF.1	Security attribute based access control
Object Re-Use	Protection of residual information in token memory	FDP_RIP.2	Full residual information protection
Cryptographic Material Protection	Import of data without security attributes	FDP_ITC.1	Import of user data without security attributes
	Export of data without security attributes	FDP_ETC.1	Export of user data without security attributes
Cryptography	Cryptographic Operations	FCS_COP.1	Cryptographic operation

Security Requirement		CC Functional Component	CC Requirement Title
Cryptographic Material Management	Key Generation	FCS_CKM.1	Cryptographic key generation
	Key Access	FCS_CKM.3	Cryptographic key access
	Key Destruction	FCS_CKM.4	Cryptographic key destruction
Data Exchange	Data Confidentiality	FDP_UCT.1	Basic data exchange confidentiality
	Integrity and authenticity of exchanged information	FDP_DAU.2	Data authentication with identity of guarantor
Security Function Management	Management of Security Roles	FMT_SMR.2	Restrictions on security roles
	Management of Security Functions	FMT_MOF.1	Management of security functions behaviour
Security Data Management	Management of Security Attributes	FMT_MSA.1(UAV) FMT_MSA.1(SOV) FMT_MSA.1(Object Attributes)	Management of security attributes
	Secure Attributes	FMT_MSA.2	Secure security attributes
	Static Attribute Initialization	FMT_MSA.3	Static attribute initialisation
	Management of TOE Security Data	FMT_MTD.1	Management of TSF data
Logical Protection of Security Functions	Abstract Machine Test	FPT_AMT.1	Abstract machine testing
	Fail-safe Behaviour	FPT_FLS.1	Failure with preservation of secure state
	Reference Monitor	FPT_RVM.1	Non-bypassibility of the TSP
	Separation	FPT_SEP.1	TSF domain separation
	Self-test	FPT_TST.1	TSF testing
	Firmware load/update	FTP_ITC.1	Inter-TSF trusted channel

Security Requirement		CC Functional Component	CC Requirement Title
Token Cloning	Intra-TOE User Data Transfer	FDP_ITT.1	Basic internal transfer protection
	Intra-TOE TSF Data Transfer	FPT_ITT.1	Basic internal TSF data transfer protection
Physical Protection of Security Functions	Tamper Evidence	FPT_PHP.1	Passive detection of physical attack
Failure handling	Maintenance of TOE operation in event of failures (fault tolerance)	FRU_FLT.1	Degraded fault tolerance
Backup	Failure recovery	FPT_RCV.1	Manual recovery
	Token backup	FDP_LUNA_BKP.1	LUNA backup

5.1.1 Requirements for Identification and Authentication

5.1.1.1 FIA_ATD.1 User Attribute Definition

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users:

- **User ID number**
- **User checkword**
- **User function vector**
- **User "locked" flags**

5.1.1.2 FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow **the following actions** on behalf of the user to be performed before the user is identified:

- **Open a session**
- **Perform basic diagnostic functions, such as checking the communications from the host to the card, checking firmware level and token info and checking information on mechanisms supported.**
- **Access Public data objects.**

FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

5.1.1.3 FIA_UAU.1 Timing of authentication

FIA_UAU.1.1 The TSF shall allow **the following actions** on behalf of the user to be performed before the user is authenticated:

- **Open a session**
- **Perform basic diagnostic functions, such as checking the communications from the host to the card, checking firmware level and token info and checking information on mechanisms supported.**
- **Access Public data objects.**

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

5.1.1.4 FIA_UAU.5 Multiple authentication mechanisms

FIA_UAU.5.1 The TSF shall provide **the following authentication mechanisms** to support user authentication:

- **M of N secret sharing**.
- **PED Key entry**
- **PED PIN entry**

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to **the following rules:**

- **A user must enter their authentication data using, at a minimum, the PED and a PED Key.**
- **If required by the security policy defined for the device, the user must enter a 6 to 16 digit Personal Identification Number (PIN) via the PED in addition to the entering the PED Key.**
- **If required by the security policy defined for the device, M out of N secret shares must first be entered via the Luna PIN Entry Device (PED) in order to enable the device for operation.**

5.1.1.5 FIA_AFL.1 (SO) Authentication failure handling

FIA_AFL.1.1 (SO) The TSF shall detect when **three (3)** unsuccessful authentication attempts occur related to **Security Officer login**.

FIA_AFL.1.2 (SO) When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **zeroize the device**.

5.1.1.6 FIA_AFL.1 (Token User) Authentication failure handling

FIA_AFL.1.1 (Token User) The TSF shall detect when **"X" (X is set by the SO as part of the configurable policy)** unsuccessful authentication attempts occur related to **Token User login**.

FIA_AFL.1.2 (Token User) When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **remove the Token User and clear the Token User's memory space and permanent storage**.

5.1.1.7 FIA_SOS.1 Verification of secrets

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet **the minimum length established by the TOE for each authentication secret**.

5.1.1.8 FIA_SOS.2 TSF generation of secrets

FIA_SOS.2.1 The TSF shall provide a mechanism to generate secrets that meet the **minimum lengths for each function for which they are required and that are random:**

1. **User authentication 48 bytes**
2. **M of N activation 32 bytes**
3. **Cloning 24 bytes.**

FIA_SOS.2.2 The TSF shall be able to enforce the use of TSF generated secrets for **the following TSF functions:**

1. **User authentication**
2. **M of N activation**
3. **Cloning**

5.1.1.9 FIA_USB.1 User-subject binding

FIA_USB.1.1 The TSF shall associate the appropriate user security attributes with subjects acting on behalf of that user.

5.1.1.10FTP_TRP.1 Trusted path

FTP_TRP.1.1 The TSF shall provide a communication path between itself and **local users**[4] that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from modification or disclosure.

FTP_TRP.1.2 The TSF shall permit **local users** to initiate communication via the trusted path.

FTP_TRP.1.3 The TSF shall require the use of the trusted path for **initial user authentication data entry** and **the following additional functions**:

- **Upload of the TSF-generated authentication data to the PED Key**
- **Entry of M of N Activation secret shares**
- **Entry of the Token Cloning Domain key**

5.1.2 Requirements for Token Access Control

5.1.2.1 FDP_ACC.1 Subset access control

FDP_ACC.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP on the following:**

(1) Device sessions (subjects).

(2) All objects created, stored, handled and destroyed in the device.

(3) Operations:

a. Create

b. Read (Query Attribute Value)

c. Copy

d. Modify

e. Destroy

f. Generate

g. Derive

h. Wrap

i. Unwrap

j. Use

k. Clone

5.1.2.2 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP** to objects based on **the following:**

(1) Subject attributes:

a. Session and Access ID

b. User ID associated with session (Access Owner)

[4] ISO 15408-2 calls for a selection between local users and remote users. In the case of the TOE, all users are local.

c. Role.

(2) Object attributes:

a. Private. If True, object is Private. If False, object is Public.

b. Owner. Object ownership is assigned to the object creator, if the object is Private. Public objects are not owned by a user. Ownership is enforced via internal key management.

c. **Sensitive. If True, object is Sensitive. If False, object is Non-Sensitive.**

d. Extractable. If True, object may be extracted. If False, object may not be extracted.

e. **Modifiable. If True, object may be modified. If False, object may not be modified.**

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **Token Access rules described in Table 5-2 below**.

The operations allowed for each user type in Table 5-2 have been abbreviated as indicated below.

Operation	Abbreviation	Operation	Abbreviation
Create	Cr	Generate, Derive	G
Read	R	Wrap	W
Copy	C	Unwrap	Un
Modify	M	Use	U
Destroy	D	Clone	Cl

Table 5-2: Access Matrix

Object Attribute				User	
Private	**Sensitive**	**Modifiable**	**Extractable**	**Logged in**	**Public**
0	0	0	N/A	Cr,R,C(1),D	Cr,R,C(1),D
0	1	0	0	R(2),C(1),D,G,U,Cl	R(2),C(1),D
0	1	1	0	R(2),C,M,D,G,U,Cl	R(2),C,M,D
0	0	1	N/A	Cr,R,C,M,D	Cr,R,C,M,D
0	1	0	1	R(2),C(1),D,G,W,Un,U,Cl	R(2),C(1),M,D
0	1	1	1	R(2),C,M,D,G,W,Un,U,Cl	R(2),C,M,D
1	0	0	N/A	Cr,R,C(1),D	---
1	1	0	0	R(2),C(1),D,G,U,Cl	---
1	1	1	0	R(2),C(1),M,D,G,U,Cl	---
1	0	1	N/A	Cr,R,C,M,D	---

Object Attribute				User	
1	1	0	1	R(2),C(1),D,W,Un,U,Cl	---
1	1	1	1	R(2),C(1),M,D,W,Un, U,Cl	---

1. The attributes of an object whose CKA_MODIFIABLE attribute is not set cannot be changed as part of a Copy operation.

2. The plaintext value of key material stored in objects whose CKA_SENSITIVE attribute is set cannot be read although the object may otherwise be accessible.

3. A "0" in the above table indicates that the attribute labeling the column has not been set. A "1" indicates that the attribute has been set.

4. A "Logged in" user will normally be in the Token User role, although the specification does not preclude the SO role from also performing the controlled operations covered by this table.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

A subject shall have access to objects for which the CKA_PRIVATE attribute is not set and to objects whose owner is set to the identity of the user to which the subject is bound.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the **following additional rules**:

- **A subject in the Public role shall not have access to objects for which the CKA_PRIVATE attribute is set.**
- **A subject in the Token User role cannot clone objects if the TPV_DISABLE_CLONING_BY_USER bit is set.**
- **A subject shall not have access to the plaintext value of an object whose CKA_SENSITIVE attribute is set.**

5.1.2.3 FDP_RIP.2 Full residual information protection

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to** all objects.

5.1.3 Requirements for Cryptographic Material Protection

5.1.3.1 FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP** when importing user data, controlled under the SFP, from outside of the TSC.

FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TSC.

FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data, controlled under the SFP, from outside the TSC*:* **The CKA_SENSITIVE attribute of the object storing user data imported via an Unwrap operation shall be set.**

5.1.3.2 FDP_ETC.1 Export of user data without security attributes

FDP_ETC.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP** when exporting user data, controlled under the SFP, outside of the TSC.

FDP_ETC.1.2 The TSF shall export the user data without the user data's associated security attributes.

5.1.4 Requirements for Cryptographic Operation

5.1.4.1 FCS_COP.1 Cryptographic operation

FCS_COP.1.1 The TSF shall perform **pseudo random number generation, symmetric and asymmetric encryption and decryption, signature generation, signature verification, authentication code generation, authentication code verification, and hash generation** in accordance with *the* specified cryptographic *algorithms* **listed below** and cryptographic key sizes **specified for each algorithm** that meet the following: **standards noted for each algorithm**.

(1) Pseudo random number generation in accordance with ANSI X9.31, Appendix A.

(2) Asymmetric encryption/decryption (key management) RSA 1024, 2048 bits in accordance with PKCS #1 and PKIX-CMP.

(3) Symmetric encryption/decryption (key wrap/unwrap) 3DES 168 bits in accordance with PKIX-CMP.

(4) Symmetric encryption/decryption 3DES 112, 168 bits (FIPS PUB 46-3, ANSI X 9.52), CAST5 80, 128 bits (RFC 2144).

(5) Signature generation/verification RSA 1024, 2048 (PKCS #1, FIPS PUB 186-2, and ANSI X9.31) with SHA-1 (FIPS PUB 180-1, ANSI X9.30 Part 2), DSA 1024 bits (FIPS PUB 186-2, ANSI X9.30) with SHA-1.

(6) Hash generation SHA-1 (FIPS PUB 180-1, ANSI X9.30 Part 2)

5.1.5 Requirements for Cryptographic Material Management

5.1.5.1 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with *the* specified cryptographic key generation *algorithms* **listed below** and specified cryptographic key sizes **specified for each algorithm** that meet the following: **standards noted for each algorithm**:

(1) RSA 1024, 2048 bits key pairs in accordance with PKCS #1.

(2) 3DES 112, 168 bits (ANSI X9.52), CAST5 80, 128 bits (RFC 2144).

(3) DSA 1024 bits key pairs in accordance with FIPS PUB 186-2, ANSI X9.30.

5.1.5.2 FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1(Storage) The TSF shall perform **key storage** in accordance with a specified cryptographic key access method, **as an opaque (encrypted) key value attribute,** that meets the following: **PKCS #11 standard**.

FCS_CKM.3.1(Access) The TSF shall perform **key access** in accordance with a specified cryptographic key access method, **return of a key handle,** that meets the following: **PKCS #11 standard**.

5.1.5.3 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1(User Delete Command) The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method:

- ***In response to a user command*, by deleting the key object being destroyed,** that meets the following: **PKCS #11 standard**.

FCS_CKM.4.1(SO Authentication Failure) The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method:

- ***In response to exceeding the threshold for SO authentication failures*, by active zeroization of all volatile and non-volatile memory,** that meets the following**: FIPS 140-1 Level 3.**

FCS_CKM.4.1(Token User Authentication Failure) The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method:

- ***In response to exceeding the threshold for Token User authentication failures*, by active zeroization of all volatile and non-volatile memory allocated to the user,** that meets the following**: FIPS 140-1 Level 3.**

FCS_CKM.4.1(Token Initialize) The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method:

- ***In response to a Token Initialize command*, by active zeroization of all volatile and non-volatile memory,** that meets the following**: FIPS 140-1 Level 3.**

5.1.6 Requirements for Data Exchange

5.1.6.1 FDP_UCT.1 Basic data exchange confidentiality

FDP_UCT.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP** to be able to **transmit and receive** objects in a manner protected from unauthorised disclosure.

5.1.6.2 FDP_DAU.2 Data authentication with identity of guarantor

FDP_DAU.2.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **user data objects**.

FDP_DAU.2.2 The TSF shall provide the **Security Officer and Token Users** with the ability to verify evidence of the validity of the indicated information and the identity of the user that generated the evidence.

5.1.7 Requirements for Management of TOE Functions

5.1.7.1 FMT_MOF.1 Management of security functions behaviour

FMT_MOF.1.1 The TSF shall restrict the ability **to disable, enable and modify the behaviour of** the functions **listed below to the roles indicated:**

(1) M of N Activation - SO may enable and disable.

(2) Token Cloning - SO may enable and disable the user capability to perform Token Cloning.

(3) Authentication Data rules – SO may enable the use of both PED key and PED PIN[5] for authentication.

(4) Login failures – SO may set the number of user login failures before user is removed.

(5) **User Firmware Update – SO may enable and disable the user capability to perform a token firmware update.**

5.1.7.2 FMT_SMR.2 Restrictions on security roles

FMT_SMR.2.1 The TSF shall maintain the roles: **Public, Token User and Security Officer**.

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions **listed below** are satisfied:

(1) No user can assume either the Token User or Security Officer role before identification and authentication;

(2) No identity can assume both a Token User and a Security Officer role; and

(3) **When M of N Activation has been enabled, no user can assume either the Token User or the Security Officer role before the M of N activation has been completed.**

5.1.8 Requirements for Security Data Management

5.1.8.1 FMT_MSA.1 (UAV) Management of security attributes

FMT_MSA.1.1 (UAV - USV/UserLockedFlag) The TSF shall enforce the **TAC SFP** to restrict the ability to **change_default, query, modify and delete** the security attributes **listed below** to the **Security Officer role:**

- **UserFunctionVector; also known as USV;**
- **UserLockedFlag**

FMT_MSA.1.1 (UAV - UserID/Checkword - SO) The TSF shall enforce the **TAC SFP** to restrict the ability to **change_default and delete** the security attributes **listed below** to the **Security Officer role:**

- **UserID;**
- **Checkword, which includes the encrypted form of user secret key plus a fixed value used for authentication**

FMT_MSA.1.1 (UAV - UserID – SO& Token User) The TSF shall enforce the **TAC SFP** to restrict the ability to **query** the security attributes **listed below** to the **Security Officer and Token User roles:**

- **UserID**

FMT_MSA.1.1 (UAV - Checkword - Token User) The TSF shall enforce the **TAC SFP** to restrict the ability to **modify** the security attributes **listed below** to the **Token User roles:**

- **Checkword**

[5] The PED PIN is entered at time of user creation as required for the security policy defined for the installed TOE.

5.1.8.2 FMT_MSA.1 (SOV) Management of security attributes

FMT_MSA.1.1 (SOV –Checkword - SO) The TSF shall enforce the **TAC SFP** to restrict the ability to **change_default and modify** the security attributes **listed below** to the **Security Officer role:**

- **Checkword, which includes the encrypted form of the SO secret key plus a fixed value used for authentication**

5.1.8.3 FMT_MSA.1 (Object Attributes) Management of security attributes

FMT_MSA.1.1 (Object Attributes) The TSF shall enforce the **TAC SFP** to restrict the ability to **modify, query and delete** the security attributes **shown in Table 5-3 to** the **Token User and SO roles.**

Table 5-3: Security Policy-Related Object Attributes

Attribute Type	Data Type	Size	Meaning
CKA_PRIVATE	Boolean	8	If set to true, this object is accessible only when the user that created it is logged in. If set to false, the object is public and can be accessed in a public session or by any user.
CKA_SENSITIVE	Boolean	8	This attribute is used in private or secret keys to indicate whether key material is stored in encrypted form and cannot be revealed externally in plain-text.
CKA_EXTRACTABLE	boolean	8	This attribute is present in all private and secret keys to indicate whether sensitive key material can be revealed externally in encrypted form. Wrapping is one such operation that can be disabled by this attribute.
CKA_MODIFIABLE	boolean	8	This attribute is present in all objects to indicate whether attributes can be modified.

5.1.8.4 FMT_MSA.2 Secure security attributes

FMT_MSA.2.1 The TSF shall ensure that only secure values are accepted for security attributes.

5.1.8.5 FMT_MSA.3 Static attribute initialization

FMT_MSA.3.1 The TSF shall enforce the **Token Access Control (TAC) SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow **No Role** to specify alternative initial values to override the default values when an object or information is created.

5.1.8.6 FMT_MTD.1 Management of TSF data

FMT_MTD.1.1 The TSF shall restrict the ability to **change_default** the **Token Policy Vector settings listed below** to the **Security Officer:**

(1) Number of User Login Fails Allowed
(2) User Firmware Update
(3) M of N Activation
(4) Key Flags Lock

(5) Key Single Function
(6) Local Signing Key Required
(7) User Zeroize[6]
(8) **Disable Cloning by User**

5.1.9 Requirements for Logical Protection of the Security Functions

5.1.9.1 FPT_AMT.1 Abstract machine testing

FPT_AMT.1.1 The TSF shall run a suite of tests **during initial start-up and at the request of an authorized user** to demonstrate the correct operation of the security assumptions provided by the abstract machine that underlies the TSF.

5.1.9.2 FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

- **Corruption or failure of the card reader device driver software on the host platform.**
- **Failure of the interface device on the host platform**
- **Firmware operation failure**
- **Failure of the card reader or the reader-token interface**
- **Loss of power for reasons other than the above**
- **Removal of the token from the reader**

5.1.9.3 FPT_RVM.1 Non-bypassibility of the TSP

FPT_RVM.1.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.1.9.4 FPT_SEP.1 TSF domain separation

FPT_SEP.1.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The TSF shall enforce separation between the security domains of subjects in the TSC.

5.1.9.5 FPT_TST.1 TSF testing

FPT_TST.1.1 The TSF shall run a suite of self-tests **during initial start-up and at the request of the authorised user** to demonstrate the correct operation of the TSF.

FPT_TST.1.2 The TSF shall provide authorised users with the capability to verify the integrity of TSF data.

FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of stored TSF executable code.

[6] The default setting is enabled and it is configurable only through the Application Programming Interface, not the normal administrative interface (Enabler).

5.1.9.6 FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and a remote trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2 The TSF shall permit **the remote trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for **firmware load and update**.

5.1.10 Requirements for Token Cloning

5.1.10.1FDP_ITT.1 Basic internal transfer protection

FDP_ITT.1.1 The TSF shall enforce the **Token Access Control (TAC) SFP** to prevent the **disclosure, modification and loss of use** of user data when it is transmitted between physically-separated parts of the TOE.

5.1.10.2FPT_ITT.1 Basic internal TSF data transfer protection

FPT_ITT.1.1 The TSF shall protect TSF data from **disclosure and modification** when it is transferred between separate parts of the TOE.

5.1.11 Requirements for Physical Protection of the Security Functions

5.1.11.1FPT_PHP.1 Passive detection of physical attack

FPT_PHP.1.1 The TSF shall provide unambiguous detection of physical tampering that might compromise the TSF.

FPT_PHP.1.2 The TSF shall provide the capability to determine whether physical tampering with the TSF's devices or TSF's elements has occurred.

5.1.12 Requirements for Availability of Data and TOE Functions

5.1.12.1FRU_FLT.1 Degraded fault tolerance

FRU_FLT.1.1 The TSF shall ensure the operation of **TOE's user data protection capabilities** when the following failures occur: **power failure or data input/output failure**.

5.1.12.2FPT_RCV.1 Manual recovery

FPT_RCV.1.1 After a failure or service discontinuity, the TSF shall enter a maintenance mode where the ability to return the TOE to a secure state is provided.

5.1.12.3FDP_LUNA_BKP.1 LUNA backup

Hierarchical to: No other components.

FDP_LUNA_BKP.1.1 The TSF shall provide the capability to backup cryptographic material without compromising the confidentiality or integrity of the cryptographic material.

FDP_LUNA_BKP.1.2 The Security Officer or Token User shall be capable of invoking the backup function on demand.

FDP_LUNA_BKP.1.3 The data stored in the backup shall be sufficient to recreate the cryptographic state of the system at the time the backup was created using only the backup token.

Dependencies: FMT_SMR.2, FDP_ITT.1, FPT_ITT.1.

`Rationale:` This component is specified to address a unique requirement for the Luna® CA[3] to be capable of performing a secure backup of cryptographic material that can be used in the recovery of the host processing environment.

5.2 Security Assurance Requirements

The assurance requirements for this TOE are as specified in the Common Criteria Version 2.1 Part 3-EAL 4 package with one augmentation. The EAL 4 package has been augmented by the addition of the Part 3 requirement, ALC_FLR.2 (Flaw Reporting Procedures).

5.2.1 Security Assurance Requirements Augmentation to EAL 4

5.2.1.1 ALC_FLR.2 Flaw reporting procedures

Dependencies:

No dependencies.

Developer action elements:

ALC_FLR.2.1D The developer shall document the flaw remediation procedures.

ALC_FLR.2.2D The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.

Content and presentation of evidence elements:

ALC_FLR.2.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.2.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.2.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.2.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.

ALC_FLR.2.5C The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.

ALC_FLR.2.6C The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.

5.3 Strength of Function Claim

The minimum Strength of Function (SoF) required for the TOE is **SoF – basic**. The explicit claim made for the authentication mechanisms and random number generation mechanism employed within the TOE is **SoF - high**. This applies to the following security functional requirements:

- FIA_UAU.1 Timing of authentication
- FIA_UAU.5 Multiple authentication mechanisms
- FIA_SOS.2 TSF generation of secrets

- FDP_ITT.1 Basic internal transfer protection (The cloning domain identifier, generated and used as part of the cloning transfer, authenticates the source and target tokens as part of the same cloning domain and, therefore, separate parts of the TOE.)
- FPT_ITT.1 Basic internal TSF data transfer protection (The cloning domain identifier, generated and used as part of the cloning transfer, authenticates the source and target tokens as part of the same cloning domain and, therefore, separate parts of the TOE.)

This is considered to be necessary and reasonable for the TOE because its primary function of protecting user key material in all environments relies heavily on the authentication processes.

6 TOE Summary Specification

6.1 Overview

The Luna® CA^3 is used primarily as a hardware security module for the protection of the private signing keys at the Certification Authority (CA), or Certification Service Provider (CSP), within a Public Key Infrastructure (PKI). As such, its primary functions are to securely generate and protect the private signing key used by the CA or CSP when signing digital certificates.

The token provides storage capability for cryptographic material generated by the token or generated by the host application as well as storage for generic data provided by the host application. Data can be stored in the form of certificate objects or in the form of data objects. When storing or generating keys (secret or private), the token imposes some restrictions on how these keys are handled. Security policy enforcement is described in more detail in section 6.2.

6.1.1 Object Model

All information stored on the token is managed as objects. These objects are characterized by different attributes used by the token to determine the handling rules of objects. The token provides two ways of storing objects: permanent (also known as token objects) and volatile (also known as session objects). Permanent objects are kept inside the token even when no power is applied to it. They are stored encrypted in a flash memory device. Session objects only exist when power is applied to the token and they are stored in volatile RAM. Another characteristic of all the objects is that they can either be visible from outside the token or invisible. Invisible objects are used by internal processes to keep information such as states and status of the token, and are not for external use. Visible objects, also referred to as external objects, are owned by external processes/users and manipulated by the token.

The Luna® object model is very closely related to the PKCS#11 standard. More details on the Luna® interface exists in the Luna® Interface Control Document.

6.1.2 Multi-Session Capability

The Luna® token manages processes on a per session basis. Applications running on the host system which require data and cryptographic services from the token have to open a session with the token before gaining access to the token's functions and objects. The session provides a logical connection between the application and the token. It is possible for an application to open multiple sessions with the token or have multiple applications each opening various sessions with the token.

The token provides an even higher level of a connection based on an access id that associates a group of sessions to a particular application. This approach allows an application or applications to share sessions opened within the scope of that access id.

6.1.3 Multi-User Capability

The Luna® token is capable of holding the security material (private keys, certificates and other private data) for users configured to access the token. A user consists of an identity number and associated authentication data. It is possible for the same user to be provided with multiple identities on the same token or that more than one user have their identity on the same token. In either case, the token is said to support multiple users. Users are created, managed and destroyed by the token Security Officer (SO).

A user must access the token through a session. Sessions are opened as Public sessions and may remain Public or become Private (logged in) following a successful user login operation. The token only allows for one user identity to be logged in at a time. Once logged in, a session becomes bound to the user identity and has access to all cryptographic operations and may generate private objects on behalf of the user or access private objects generated on behalf of the user in previous sessions. Other users may

open additional Public sessions but they are not permitted to perform a login operation and they are not permitted access to the already-logged in user's private objects. For example, if user A opens a session with an inactive token (a token that does not currently support an active session) and then performs a login process within the established session, user A becomes logged-in to the token. From then on, user A can open any number of sessions on the token, up to the maximum allowed, and log into these sessions without any conflicts. Meanwhile, it is still possible for user B to open one or more sessions (as Public sessions) with the token, but this user is prevented from logging into any of these sessions, even if the user has a legitimate PIN and identity. This restriction remains as long as user A is active on the token (i.e., user A is still logged-in to at least one session).

6.1.4 Security Policy Tools

6.1.4.1 Fixed Policy Vector (FPV)

The FPV contains the settings necessary to enforce policy rules that apply across a wide range of customer environments, regardless of the individual organizational policies. For example, one bit in the FPV defines whether the token can be used to wrap keys to export them to another module. This allows the token to be configured at manufacturing time to meet the requirements of some security policies that do not permit the extraction of private keys under any conditions.

The FPV cannot be modified by the SO or any of the users. 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 CRC computation.

The format of the FPV is a 32-bit vector that is divided into four fields of eight bits. These fields and their contents are defined in the Security Policy Model document.

6.1.4.2 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 User.

The format of the TPV is a 32-bit vector that is divided into four fields of eight bits. These fields and their contents are defined in the Security Policy Model document.

6.2 IT Security Functions

6.2.1 User Identification and Authentication

A user is defined as an entity that acts to perform an operation on the token. In most instances, this will be a host application program such as a PKI Certification Authority implementation. The Luna® token supports three user roles; Public, Token User and Security Officer. Public users are unauthenticated and may perform a limited set of functions, such as opening a session with a token and performing pre-defined diagnostics such as checking the communications from the host to the token, checking firmware level and token info and checking information on mechanisms supported. For a user to assume either the Token User or Security Officer role the token enforces user identification and authentication. In the case of Token Users, the user number must be provided to the token to identify the user as well as valid authentication data to authenticate the identity before access to private data and token services are granted. In the case of the Security Officer (SO), only authentication data is required. Token Users also have a textual name associated with them. The name corresponding to a particular user number can be queried from the token.

The Luna® CA^3 requires that Token Users and the SO be authenticated using, at a minimum, the Luna PIN Entry Device (PED) and a properly initialized PED Key. In the case of a Token User, which is normally an application program, the entry of the PED Key is done by a human operator acting on behalf of the application in response to a prompt issued by the Luna® CA^3 through the PED. The PED Key, once initialized, is an Identification and Authentication device and must be safeguarded accordingly by the administrative or operations staff responsible for the operation of the Luna® CA^3 within the customer's environment. The PED Key contains the user's identification number and a pseudo-randomly generated authentication string for the user and is entered into the key receptacle in the PED in order to identify and authenticate the user. The Security Officer (SO) has a special PED Key that is used during initialization and for any changes to the local security policy that may be necessary. The SO can define the security policy to require the entry of a 6 to 16 digit Personal Identification Number (PIN) via the PED in addition to the entry of the PED Key for Identification and Authentication.

Following a successful login, the user is bound to the subject acting on its behalf by having the User Authorization Vector (UAV) data included in the state data maintained by the session manager. In the case of the Luna® token, the subject acting on behalf of a user is a session. The relationship between the user and the session is discussed in more detail in section 6.1.3 and the data contained in the UAV is described in section 6.2.4.

The Luna® token also implements a maximum login attempts policy. This feature serves to prevent an exhaustive search approach to find the authentication data of the SO or a user. The implementation of this feature differs for an SO authentication data search and a user authentication data search.

In the case of a user PIN search:

- If "y" consecutive user logon attempts fail ("y" is defined by the SO in the Token Policy Vector (TPV)), the token removes the user and clears the user's memory space and permanent storage. The user must be re-created by the SO in order to be recovered.

In the case of an SO PIN search:

- If "x" consecutive SO logon attempts fail, the token is zeroized. "x" is normally set at three and is defined in the Fixed Policy Vector (FPV) which is created when the token is manufactured and cannot be modified afterwards without invalidating the CRC value of the software load.

The Luna® CA^3 can also be configured by the Security Officer, via the Token Policy Vector (TPV), to require the use of an M of N secret sharing authentication scheme to enable the token for operation. M of N activation is described in more detail in the following sub-section.

6.2.1.1 M of N Activation

The M of N activation protocol provides the capability to implement a policy of separation of duties and multi-person integrity to control the activation of the Luna® CA^3 token. It does this by generating a 32 byte secret on the token and protecting it by "splitting" it into "N" pieces and storing each piece on a serial memory device known as a PED Key. Any "M" of these "N" pieces must be reassembled by inserting the corresponding PED Keys into the Luna® PIN Entry Device or PED in order to reconstruct the original secret. Many ways exist to achieve this capability and the one used in the Luna® tokens is based on Shamir's threshold scheme.

6.2.2 Trusted Path

In the case of the Luna® CA^3, user authentication data and other critical security parameters are protected through the use of a separate port and data path for their transfer, and by providing mechanisms to protect their confidentiality and integrity. Attached to this separate data port via the PC Card reader is a PIN Entry Device or PED. Through the PED and the separate data path, identification data and randomly generated authentication data for the SO and Token Users is stored on a serial memory token device

known as a PED Key. The authentication data stored on a PED Key is entered through the PED to the token via the PED's dedicated data port during the login process. Because the PED has a direct serial communications interface to the token, only local entry of authentication data is possible.

Critical security parameters other than user authentication data that must be exchanged via the trusted path include the key cloning domain identifier and the M of N shares because they are considered plain-text cryptographic key components.

6.2.3 Authentication Data Selection

The user authentication data is a 48 byte value that is randomly generated by the token and stored on the PED Key. The PED Key represents the user to the token and, therefore, must be properly protected within the environment in which the token operates. The TOE can be configured by the SO, as part of the token initialization process, to require the entry of a separate PIN to the PED in order to unlock the PED Key. A Token User and the SO can request to change their respective authentication data at any time using the Enabler interface.

6.2.4 User Account Data

The Security Officer is the only role allowed to create users, modify user status and delete users. The TOE maintains a user's account data in a User Authorization Vector (UAV) that is stored in memory reserved for the TOE's use. The UAV includes the following data:

- User ID number
- User checkword
- User function vector
- User "lockout" status

The user checkword contains the user's secret key and a validation string encrypted using a key derived from the user's authentication data. The secret key is randomly generated by the token at the time the user is created and is used to encrypt a user's objects on the token. The validation string is a known byte string used to verify that the checkword has been decrypted correctly.

6.2.5 Token Access Control

The Luna® token implements an identity-based access control policy that applies to all objects on the Token, in particular to private key and secret key objects, and governs a subject's access to an object using the following operations:

- Create
- Read (Query Attribute Value)
- Copy
- Modify
- Destroy
- Generate
- Derive
- Wrap
- Unwrap
- Use

- Clone

A subject's access to objects stored on the token is mediated on the basis of the following subject and object attributes:

- Subject attributes:
  - Session and Access ID
  - User ID associated with session (Access Owner)
  - Role.
- Object attributes:
  - Private. If True, object is Private. If False, object is Public.
  - Owner. Object ownership is assigned to the object creator, if the object is Private. Public objects are not owned by a user. Ownership is enforced via internal key management.
  - Sensitive. If True, object is Sensitive. If False, object is Non-Sensitive.
  - Extractable. If True, object may be extracted. If False, object may not be extracted.
  - Modifiable. If True, object may be modified. If False, object may not be modified.

Private data objects are labeled with a number corresponding to their owner and are encrypted using the owner's secret key. Sensitive objects are encrypted with either the owner's secret key, if they are Private or the token secret key if the object is Public. Private data objects are only accessible by the object owner. Public data objects may be accessed by any user with an active session on the token. Key objects are always created as Sensitive objects and can only be used for cryptographic operations by a logged in SO or Token User. Only data and certificate objects can be non-sensitive. Key objects that are marked as extractable may be exported from the token using the Wrap operation. RSA private keys are never extractable from the Luna® CA[3].

The token does not allow any granularity of access other than owner or public (i.e., a data object cannot be accessible by two users and restricted to other users). Ownership of an object gives the owner access to the object through the allowed operations but does not allow the owner to assign a subset of rights to other users. Allowed operations are those permitted by the Fixed and Token Policy Vector settings and the access matrix in Table 5-2. The SO can specifically disable the ability of a Token User to access the cloning function.

6.2.6 Object Reuse

The token enforces an object reuse policy in that every object is allocated its own portion of memory (flash or volatile RAM). Permanent objects (stored in flash) are maintained in an encrypted state at all times, and their information content is, therefore, never available outside the TSF boundary. The policy also ensures that no permanent object is placed in a previously allocated memory location unless all previous memory content is purged and zeroized. When cryptographic functions are performed, a cryptographic context is created to hold data required by the function (e.g., a DES key schedule for a DES function). The cryptographic context only exists in volatile RAM memory and is not accessible to any functions except those defined by its owner function. The memory assigned to a cryptographic context is always purged of its content before being handed over to another function. Direct access to either volatile or flash memory locations is never provided to users; all user interaction with the token is via memory handles.

6.2.7 Cryptographic Material Protection

Private objects are protected within the token by encryption with the user's secret key as well as being protected by token's access control function. Sensitive objects are also protected by encryption,

regardless whether they are Public or Private. Private key and secret key objects always have their sensitive attribute set.

Sensitive objects may only be exported from the TOE boundary in a wrapped (encrypted) form if the Extractable attribute is True. Objects are exported from the token without their associated security attributes. If the Extractable attribute is False, the object may not be exported from the token boundary under any condition.

Objects may be imported into the token under the control of the Token Access Control policy. Secret keys and/or private keys generated in the host IT environment may only be imported into the token by an unwrapping operation on the token. Any attributes of keys imported in this way are ignored by the TOE and their attributes are set to default values by the TOE. Unwrapped keys have their Sensitive attribute set True by the TOE. The local policy (TPV) may also be set to prohibit the use of externally generated private keys for signing operations on the token.

6.2.8 Cryptographic Operations

Because of its generic nature, the Luna® token firmware supports a wide range of cryptographic algorithms and mechanisms a number of which are not relevant to the Luna® CA^3 specifically. The cryptographic functions and algorithms that are relevant to the Luna® CA^3 are the following:

(1) Pseudo random number generation in accordance with ANSI X9.31, Appendix A.

(2) Asymmetric encryption/decryption (used for key management) RSA 1024, 2048 bits in accordance with PKCS #1 and PKIX-CMP.

(3) Symmetric encryption/decryption (key wrap/unwrap) 3DES 168 bits in accordance with PKIX-CMP.

(4) Symmetric encryption/decryption 3DES 112, 168 bits (FIPS PUB 46-3, ANSI X 9.52), CAST5 80, 128 bits (RFC 2144).

(5) Signature generation/verification RSA 1024, 2048 (PKCS #1, FIPS PUB 186-2, and ANSI X9.31) with SHA-1 (FIPS PUB 180-1, ANSI X9.30 Part 2), DSA 1024 bits (FIPS PUB 186-2, ANSI X9.30) with SHA-1.

(6) Hash generation SHA-1 (FIPS PUB 180-1, ANSI X9.30 Part 2)

All cryptographic operations are performed in the token's cryptomodule that has been formally validated as meeting the FIPS 140-1 Level 3 standard.

DES and triple-DES are symmetric cipher algorithms. They are used by the host applications to encrypt or decrypt data. Triple-DES is an enhancement of the DES algorithm by applying DES three times (Encrypt-Decrypt-Encrypt) using a double or triple-length key (168 or 112 bits vice 56 bits) on each cipher block. This process results in slower performance but provides a much stronger cryptographic algorithm.

CAST and CAST5 are block cipher algorithms developed by Entrust Technologies Inc.

The necessary keying material needed by these algorithms may be generated or derived on-board. Random data needed to produce sound key material is generated by the token's Pseudo Random Number Generator. In some cases, key material may be imported from an external source in an encrypted (wrapped) form and decrypted (unwrapped) inside the token.

The triple-DES algorithm is also used by the token to encrypt sensitive information stored in flash memory. The keys used in this case are owned by users or by the Security Officer (SO) on the token. These symmetric keys are randomly generated when the token is initialized in the case of the SO or at the time a token user is created by the SO. They are stored in the parameters area of the token and protected by Triple-DES Password-Based Encryption (PBE) using either the user's PIN or the SO's PIN as appropriate.

SHA-1 is a data hashing algorithm used to derive a digest from a large data input.

RSA and DSA are used to provide signature capabilities as well as electronic key transport functions in the case of RSA. These algorithms are based on public cryptographic ciphers that require two keys to function properly: a private key and a public key. Private keys generated by the token cannot be extracted from the token except under the protection of a strong cryptographic process and only as permitted by local security policy. The public keys however, can be extracted from the token by any application (for example, for certification by a Certification Authority (CA)).

The RNG process is based on Appendix A of the ANSI X9.31Standard. It uses triple-DES to perform random bit generation and performs on-line testing during all generation phases. An optional statistical test can be performed to validate the quality of the RNG process.

6.2.9 Cryptographic Material Management

Cryptographic material (key) management functions protect the confidentiality of key material throughout its life-cycle. The key management functions provided by the Luna® CA[3] are the following:

(1) Cryptographic key generation in accordance with the following indicated standards:

   a. RSA 1024, 2048 bits key pairs in accordance with PKCS #1.

   b. 3DES 112, 168 bits (FIPS PUB 46-3, ANSI X9.52), CAST5 80, 128 bits (RFC 2144).

   c. DSA 1024 bits key pairs in accordance with FIPS PUB 186-2, ANSI X9.30.

(2) Secure key storage and key access following the PKCS #11 standard.

(3) Destruction of cryptographic keys is performed in one of three ways as described below in accordance with the PKCS #11 and FIPS PUB 140-1 Level 3 standards.

   a. An object on the token that is destroyed using the PKCS #11 function C_DestroyObject (the user delete command available through the API) is marked invalid and remains encrypted with the user's key or the token's general secret key until such time as its flash locations are re-allocated for additional data on the token; at which time they are purged and zeroized before re-allocation. The same strategy of marking an object invalid and purging the memory content before re-allocation is followed for volatile memory as well as flash.

   b. Objects on the token that are destroyed as a result of authentication failure are zeroized (all flash blocks in user's memory turned to 1's). If it is an SO authentication failure all flash blocks on the token are zeroized.

   c. Objects on the token that are destroyed through C_InitToken (the user command to initialize the token available through the API) are zeroized, along with the rest of the flash memory being used by the SO and User.

Keys are always stored as secret key or private key objects with the Sensitive attribute set. The key value is, therefore, stored in encrypted form using the owning user's secret key, if its Private attribute is set, or the general token secret key if the Private attribute is not set. Access to keys is never provided directly to a calling application. A handle to a particular key is returned that can be used by the application in subsequent calls to perform cryptographic operations. Key storage and access is performed in accordance with the PKCS #11 object model and function specifications.

All cryptographic material management functions are performed in the token in accordance with the appropriate cryptographic standards using mechanisms that have been formally validated as meeting the FIPS PUB 140-1 Level 3 standard.

6.2.10 Data Exchange

The Luna® token provides security functions that support secure data exchange in three main ways:

- Data integrity and authenticity is protected through the use of RSA and DSA digital signatures. The digital signature of the data object provides evidence of data validity. The Luna® token provides logged in users (SO and Token Users) the ability to generate evidence in the form of a digital signature provided they have access to the private signing key and to verify the evidence and the identity of the originator who generated the evidence provided they have possession of the digitally signed information and access to the signer's verification public key.
- Data confidentiality is protected through the use of symmetric and/or asymmetric encryption/decryption in the Wrapping and Unwrapping operations. A cryptographic channel is established for the transfer of wrapped objects as described below.

Wrapping and unwrapping of key material between the TOE and other entities can only take place if prior agreement has been reached regarding the key to be used for the wrap and unwrap operations. This can either be through key sharing of a secret key for use with a symmetric encryption algorithm or through the use of the public key of the intended recipient with an asymmetric encryption algorithm.

6.2.11 Security Function Management

The Luna® token supports Public, Token User and Security Officer roles, under the following conditions:

- No user can assume either the Token User or Security Officer role before identification and authentication;
- No identity can assume both a Token User and a Security Officer role; and
- 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.

The token provides security management capabilities for the Security Officer (SO) to disable, enable and modify the behaviour of the following functions:

1. Enable and disable M of N Activation.
2. Enable and disable Token Cloning by Token Users
3. Authentication Data rules – SO may select use of both PED key and PIN and minimum/maximum PIN lengths
4. Set the number of user login failures before user is erased
5. Enable and disable the user capability to perform a token firmware update.

6.2.12 Security Data Management

The Luna® token allows the Security Officer and Token User to manipulate security-relevant data stored on the token. Specifically, it allows only the Security Officer to change the default values of Token Policy Vector (TPV) settings listed below:

1. Number of User Login Fails Allowed
2. User Firmware Update
3. M of N Activation
4. Key Flags Lock
5. Key Single Function
6. Local Signing Key Required
7. User Zeroize

8. Disable Cloning by User

The SO establishes the TPV settings during the initialization process. Once set by the SO, the TPV cannot be subsequently modified without re-initializing the token.

The User Authorization Vector, described in section 6.2.4, is the data structure used by the token to store the user's security attributes. The Token Access Control policy restricts the ability to manipulate the UAV data as described below:

1. Only the Security Officer role can change_default, query, modify and delete the following security attributes:
    - UserFunctionVector; also known as USV;
    - UserLockedFlag
2. Only the Security Officer role can change_default and delete the following security attributes:
    - UserID;
    - Checkword, which includes the user secret key plus a fixed value used for authentication in encrypted form
3. Only the Security Officer and Token User roles can query the following security attributes:
    - UserID
4. Only the Security Officer and Token User roles can modify the following security attributes:
    - Checkword

The Token Access Control policy also restricts the ability to modify, query and delete security-relevant object attributes, listed in Table 5-3 to the Security Officer and Token User roles.

The token assigns default attributes to objects as they are created. The creator of the object may specify values different from the defaults with the exceptions described below.

There are security-relevant object attributes that are set to restrictive default values based on the Fixed Policy Vector (FPV) in the Luna® CA^3 that cannot be changed by anyone. These attributes and their settings are the following:

1. The CKA_SENSITIVE attribute is set TRUE for all secret and private key objects.
2. The CKA_EXTRACTABLE attribute is set FALSE for all private key objects.

6.2.13 Logical Self-Protection of Security Functions

The Luna® token ensures the logical protection of its security functions from attempts to subvert or bypass security enforcement by implementing a number of self-protection measures. The main self-protection features are the following:

The firmware integrity is protected by an error detection code based on a cryptographic hash function. Integrity is checked when the firmware is initially loaded or updated and every time the token is activated. The token will halt if the firmware integrity is not verified.

The token performs a number of tests of security-critical functions each time it is activated and on demand from a user. The token offers three categories of self-tests that can be called up by the user at any time: Hardware, cryptographic and RNG statistical checks. The hardware self test verifies access to all of the volatile RAM memory. The cryptographic self-test performs a test of all of the cryptographic algorithms provided by the token. These self-tests are based on a known answer test methodology where a known key is used to process a known data input and the result obtained is compared to a previously-calculated

answer. The RNG statistical test consists of a suite of statistical tests performed on the output of the random number generation process.

The token prevents bypass by ensuring that TSP enforcement functions are invoked and succeed before allowing a subsequent token firmware function to proceed. It maintains a separate domain for its own execution that is protected from external agents. It also separates users by encrypting private objects with the user's secret key and by allowing only one logged in user to be active on the token at any time as well as allowing only one thread of execution on the token at any one time.

The token preserves itself in a secure state in the event of common failure events listed below:

1. Corruption or failure of the card reader device driver software on the host platform.
2. Failure of the interface device on the host platform
3. Firmware operation failure
4. Failure of the card reader or the reader-token interface
5. Loss of power for reasons other than the above
6. Removal of the token from the reader.

Luna CA^3 requires the use of a cryptographically protected trusted channel for initial firmware loading at Chrysalis-ITS prior to delivery to the customer and when the firmware is later updated at the customer's site. The trusted channel is provided as described in the following sentences. Firmware can only be initially loaded onto a Luna CA^3 token from a separate token dedicated for the purpose and containing a firmware image that has been digitally signed by Chrysalis-ITS and encrypted using a secret key generated specifically for this purpose and if the token itself is a valid Luna CA^3 token. For firmware updates, the updated image is signed and encrypted using a dedicated token at Chrysalis-ITS and distributed to customer sites in software form along with a separately distributed authorization code. The digital signature assures the receiving token that the updated image originated at Chrysalis-ITS and that it has not been modified. Encryption of the image ensures that its confidentiality is protected. The use of the authorization code ensures that only authorized customers may perform the update and specially designed symmetric key management techniques ensure that only valid CA^3 tokens can decrypt the image in order to perform the update. The trusted channel for communicating the firmware image from the dedicated Chrysalis-ITS token to the target CA^3 token is initiated by the dedicated token because it is the one that generates the symmetric keys and authorization code, digitally signs the image and encrypts it for transmission to the target. The actual firmware update process is performed by the target CA^3 token and the first part of that process completes the communication that was initiated by the dedicated Chrysalis-ITS token.

6.2.14 Token Cloning

For performance and secure backup purposes, Luna® tokens may be grouped in clusters of tokens that are referred to as "domains." A domain is established by generating a 24 byte secret, known as a cloning domain key or cloning domain identifier, on one token (that could be considered to be the "master" token for the domain) and transferring the secret securely via the PED to other tokens that are to be part of the domain. The cloning domain key is then used during the mutual authentication and key agreement exchange that takes place between tokens as described briefly below. This mutual authentication ensures that the two tokens participating in the cloning operation belong to the same cloning domain and can thus be treated as physically separated parts of the same TOE.

When tokens are members of a domain, they must be capable of operating in such a way that they behave as one identical token to the calling application. The Token Cloning function provides the capability to duplicate the cryptographic state of a token by cloning token objects from a source token to a target token within the same cryptographic domain in a cryptographically protected fashion that prevents modification and disclosure.

When Token Cloning is invoked, the Cloning Protocol protects security-relevant data from disclosure and modification when it is transferred between separate parts of the TOE. The protocol is designed such that source and target tokens both participate in ensuring that objects are all transferred correctly between tokens. The source token maintains its original state and, therefore, any sort of failure of the cloning function will not result in a loss of use of the original objects. It also ensures that any data exchanged during the cloning operation cannot be replayed in order to gain unauthorized access to the token.

The Luna® Cloning Protocol implements a mutual authentication mechanism to ensure that both tokens are members of the same domain by providing mutual authentication of the two tokens. The mechanism uses cryptographic techniques to provide mutual authentication, proof of origin, integrity and confidentiality of the objects being transferred from source to target token within a domain. The key management scheme used within the cloning protocol also protects against replay attacks and minimizes the impact of possible key compromise by ensuring that a unique triple-DES key is used for each cloning operation.

6.2.15 Physical Self-Protection

Tamper-evident features are implemented in the manufacture of the token. Any tampering that might compromise the token's security can be detected by visually inspecting the physical integrity of the token.

The token's physical design also resists visual inspection of the device design, physical probing of the device and attempts to access sensitive data on individual components of the token and provides evidence of the occurrence of such physical tampering.

6.2.16 Failure Handling

If power is lost to the token for whatever reason, permanent objects (private keys, etc.) are preserved and remain cryptographically protected; session objects are cleared from the token. The token can be placed back into operation without compromise of its functionality or permanently stored data. In case of power failure in the host IT environment, host system restart or other circumstances that do not affect the token's operational capability, the token will ensure continued protection of sensitive material and will permit recovery from the last logged in state.

Data input/output failures would only affect the processing of the current command and, because no PKCS #11 API function returns sensitive plaintext data, there could be no compromise of the user data protection capabilities. Because of the way in which commands are handled, the token would remain in the state it was at the last successful command completion. When data input/output capability is restored the token would resume operation of that state.

6.2.17 Backup and Recovery

As described in the previous sub-section, the token maintains its secure state in the event of a failure. Depending on the nature of the failure, the token will maintain its secure state and resume operation as described below:

1. In the event of host system discontinuity the token maintains its current logged in state and resumes that state when the host system restarts.
2. In the event that power is lost to the token, it maintains its secure state by maintaining the encryption of all sensitive data and it requires the SO or Token User to login prior to resuming operation. It will resume operation with all security properties intact but the operational state of the token prior to loss of power will be lost.
3. In the event of a catastrophic damage to or failure of the token itself, recovery is accomplished by inserting and activating a backup token, as described below.

The Luna® CA^3 provides the capability to securely backup a token using the Token Cloning function. Because the Token Cloning function securely duplicates all objects from the primary to the backup token,

the backup token allows resumption of operations with the backup in the same way as described in point 2 above. The backup token may be securely stored and retrieved to allow recovery or it may be permanently inserted in the second slot of the card reader, permitting quicker recovery.

6.3 Strength of Function

The minimum Strength of Function (SOF) required by the TOE is SOF-basic. The Strength of Function claimed for the authentication mechanisms and random number generation mechanism employed within the TOE is SOF-high. This applies to the following IT Security Functions:

- 6.2.1 User Identification and Authentication
- 6.2.1.1 M of N Activation
- 6.2.3 Authentication Data Selection
- 6.2.14 Token Cloning

The TOE provides this Strength of Function based on the following:

1. Authentication requires a protected data storage device to hold one component of the authentication data, a specially constructed PIN Entry Device and a specially constructed PC Card reader in order to input authentication data to the token.
2. The authentication data stored on the protected data storage device is 48 bytes in length and is generated by a pseudo-Random Number Generator (RNG), which itself has a rating of SOF-high.
3. Authentication, in the evaluated configuration, also requires the input of a Personal Identification Number (PIN) of between 6 and 16 digits.
4. The M of N secret sharing scheme that may also be employed for authentication is based on a recognized strong secret sharing algorithm, with the base secret being 256 bits in length.
5. The cloning domain identifier used to authenticate used to authenticate source and target tokens as part of the cloning protocol is 192 bits in length.

The Strength of Function (SOF) claimed for the pseudo-Random Number Generator (RNG) used by the authentication mechanism is SOF-high based on its having been validated as part of the Luna CA^3 validation against the FIPS 140-1 standard for cryptographic modules.

6.4 Assurance Measures

The assurance requirements for this TOE are as specified in the EAL 4 package augmented by Flaw Reporting (ALC_FLR.2). Appropriate documentation, plans and procedures are in place to satisfy the specified assurance requirements. References to the appropriate supporting documentation are provided in Table 8-9: Assurance Measures.

7 PP Claims

This Security Target does not claim conformance with any Protection Profile.

8 Rationale

The mappings and descriptions of the rationale for the Security Objectives, Security Requirements and Dependencies and Assurance Measures are presented in the tables.

8.1 Security Objectives Rationale

The tables Table 8-1, Table 8-2 and Table 8-3 demonstrate the necessity of the security objectives and their appropriateness in countering the stated threats and providing for the stated assumptions.

8.2 Security Requirements Rationale

Table 8-4 shows the necessity of the Security Functional Requirements and Table 8-5 maps Security Functional Requirements to Security Objectives and provides the rationale that the SFRs, singly or in combination, meet the Security Objectives. Table 8-6 demonstrates that all dependencies for the SFRs have been met. Table 8-7 maps IT Security Functions to IT Security Requirements and Security Functional Requirements. This shows the necessity of each of the IT Security Functions presented in the TOE Summary Specification. Table 8-8 maps Security Functional Requirements to IT Security Functions and presents the rationale for why each IT Security Function satisfies the requirements of a Security Functional Requirement.

8.3 Appropriateness of Assurance Requirements

The assurance requirements chosen for the TOE, EAL 4 augmented by ALC_FLR.2, are considered to be appropriate for the TOE in its assumed (and intended) operating environment for the following reasons:

1. There are specific customer requirements for Certification Authority (CA) or Certificate Issuing and Management System (CIMS) components that meet the EAL 4 assurance requirements. The TOE, as part of a larger CA or CIMS, must meet the EAL 4 requirements at a minimum, but does not need to exceed them.
2. Because the CA and CIMS systems, for example, are critical infrastructure systems, customers require a relatively high level of assurance that the components that make them up have been developed and are maintained using sound engineering security practices.
3. It is assumed that, for most of its life-cycle, the TOE will be contained within a larger secure environment. It will, therefore, not be exposed to a threat environment that includes highly capable, untrustworthy persons. The main exception to this is when it is in transit when it will be in a state that is either zeroized or where all of its sensitive data will be encrypted using 3DES encryption. Thus, the assumption of low attack potential for vulnerability analysis is appropriate.
4. The augmentation of including ALC_FLR.2 is in response to existing company practice that has been implemented to meet customer requirements for flaw reporting and fixing.

8.4 Assurance Measures

Table 8-9: Assurance Measures shows each of the security assurance requirements of the TOE and maps each to the applicable assurance evidence provided for the evaluation.

8.5 Appropriateness of Strength of Function

The minimum Strength of Function requirement is SOF-basic. This is consistent with the EAL 4 assurance requirements that assume a low attack potential and is considered to be appropriate for the TOE security functions given its assumed (and intended) operating environment because attackers can be assumed to have only low attack potential, as described in section 3.2. The explicit Strength of Function claims of SOF-high for the authentication mechanisms and the Random Number Generator are considered to be necessary given the intended role of the TOE as the hardware cryptographic module in what is typically a critical infrastructure system.

Table 8-1 Necessity of Security Objectives

Objective	Necessitated by:
O.I&A	T.Impersonate
O.Auth_Data_Protect	T.Weak_PIN_Gen, T.Impersonate
O.TrustedPath	T.Intercept_PIN, T.Impersonate, T.Leak_Security_Data
O.Login_Limit	T.Impersonate
O.Restrict_Actions_Before_Authentication	T.Weak_Access_Control
O.Access	T.Weak_Access_Control, T.Data_Export, T.Data_Import, T.Key_Read
O.Object_Sec_Attributes	T.Weak_Access_Control
O.User_Sec_Attributes	T.Weak_Access_Control
O.User_Data_Protect	T.Weak_Access_Control, T.Key_Read
O.Random_Generate	T.Weak_Key_Gen, T.Weak_PIN_Gen
O.Crypto_Algorithms	T.Exchange_Disclosure, T.Exchange_Modification, T.Repudiate
O.Key_Generate	T.Key_Forced, T.Weak_Key_Gen
O.Key_Destroy	T.Brute_Force
O.Key_Zeroize	T.Brute_Force
O.Exchange_Confidentiality	T.Exchange_Disclosure
O.Exchange_Integrity	T.Exchange_Modification
O.Data_Authenticity	T.Repudiate
O.Admin	T.Weak_Access_Control, T.Weak_Policy_Settings
O.Limitation_of_Privilege	T.Weak_Policy_Settings
O.Multi-Person_Control_of_Sensitive_Functions	T.Improper_Operation
O.Security_Roles	T.Improper_Operation
O.Security_Data_Protect	T.Weak_Policy_Settings, T.Unauth_Function
O.Secure_Init	T.Interrupt, T.Init_Fail, T.Init_Data_Errors_Accidental
O.Self_Protect	T.Unauth_Function
O.Import_Code	T.Unauth_Function, T.UA_Program_Load
O.Inadvertent	T.Init_Data_Errors_Accidental, T.Interrupt , T.Init_Fail
O.Tamper_Evidence	T.Disclosure_Physical

Objective	Necessitated by:
O.Alt_Authentication	T.Forget_PIN
O.Backup	T.Failure
OE.Authdata	A.Admin, T.Impersonate
OE.Admin	A.Admin, T.Disclosure_Physical
OE.Legitimate_Applications	A.Legitimate_Applications
OE.Competence	A.Admin, T.Improper_Operation, T.Weak_Policy_Settings, T.Init_Data_Errors_Accidental
OE.EM_Emanations	A.EM_Emanations
OE.Connect	A.Host_Connection
OE.Power_Interruption_Protection	A.Power_Interrupt_Protection
OE.Disaster_Protection	A.Disaster_Protection
OE.Physical	A.Controlled_Access, A.Host_Connection
OE.Recovery	A.Admin, A.Power_Interrupt_Protection, A.Disaster_Protection, T.Init_Fail, T.Failure
OE.EMI	A.EMI

Table 8-2: Mapping of Objectives to Threats

Threats	Objectives	Rationale
T.Intercept_PIN	O.TrustedPath	This objective counters the threat by ensuring that authentication data is protected from potential attackers.
T.Weak_PIN_Gen	O.Random_Generate, O.Auth_Data_Protect	This combination of objectives counters the threat by ensuring that easily guessed authentication data is not allowed to exist.
T.Impersonate	O.I&A, O.Auth_Data_Protect, O.TrustedPath, O.Login_Limit, OE.Authdata	This combination of objectives counters the threat by ensuring that users must be identified and authenticated and the data used for authentication is protected. Any attempt to impersonate a legitimate user will also be made significantly more difficult by imposing a limit on the number of authentication attempts before the user is erased.
T.Weak_Access_Control	O.Restrict_Actions_Before_Authentication, O.Access, O.Object_Sec_Attributes, O.User_Sec_Attributes, O.User_Data_Protect, O.Admin	This combination of objectives counters the threat by ensuring that users are limited in the actions they may perform before I&A and that their actions following successful I&A are controlled in accordance with the TOE security policy. They also ensure that the security attributes used to enforce the policy are protected and that data stored on the token, particularly private keys, is protected at all times.
T.Data_Export	O.Access	This objective counters the threat by controlling a user's ability to export objects from the token.
T.Data_Import	O.Access	This objective counters the threat by controlling a user's ability to import objects into the token.
T.Key_Forced	O.Key_Generate	This objective counters the threat by ensuring that keys can only be generated on the token using secure techniques.
T.Weak_Key_Gen	O.Key_Generate, O.Random_Generate	This combination of objectives counters the threat by ensuring that keys can only be generated on the token using secure techniques and that they are properly protected.
T.Key_Read	O.User_Data_Protect, O.Access	This combination of objectives counters the threat by ensuring that sensitive key material cannot be read in plaintext form inside or outside the TOE boundary.

Threats	Objectives	Rationale
T.Brute_Force	O.Key_Destroy, O.Key_Zeroize	This combination of objectives counters the threat by ensuring that key data is properly destroyed upon request by an authorized user and by ensuring that key data is completely erased from the TOE in response to a logical brute force attack by an authorized person.
T.Exchange_Disclosure	O.Crypto_Algorithms, O.Exchange_Confidentiality	This combination of objectives counters the threat by ensuring that the TOE has the capability to protect data exchanges from unauthorised disclosure.
T.Exchange_Modification	O.Crypto_Algorithms, O.Exchange_Integrity	This combination of objectives counters the threat by ensuring that the TOE has the capability to protect data exchanges from unauthorised modification.
T.Repudiate	O.Crypto_Algorithms, O.Data_Authenticity	This combination of objectives counters the threat by ensuring that the TOE has the capability to generate the evidence needed to support non-repudiation and to verify such evidence.
T.Improper_Operation	O.Multi-Person_Control_of_Sensitive_Functions, O.Security_Roles, OE.Competence	This combination of objectives counters the threat by making it unlikely that the TOE's security functions can be inadvertently compromised by improper operation by ensuring that competent personnel are in place and the TOE can require the co-operation of more than one individual to perform critical functions.
T.Disclosure_Physical	O.Tamper_Evidence, OE.Admin	This combination of objectives counters the threat by ensuring that evidence of physical tampering is provided allowing an authorised individual to take appropriate action.
T.Leak_Security_Data	O.Trusted_Path	This objective counters the threat by ensuring that security-critical data is only entered via a trusted path and is not present in the host IT environment.
T.Weak_Policy_Settings	O.Admin, O.Limitation_of_Privilege, O.Security_Data_Protect, OE.Competence	This combination of objectives counters the threat by ensuring that competent personnel are in place and that proper security administration functions exist and that access to privileged functions is controlled. The confidentiality of stored security data is also protected.
T.Unauth_Function	O.Security_Data_Protect, O.Self _Protect,	This combination of objectives counters the threat by

Threats	Objectives	Rationale
	O.Import_Code	ensuring that stored security data is protected and that the TOE provides self-protection capability plus the ability to control the code that may be loaded into the device. This ensures that unauthorised functions cannot be executed either by subverting the TOE itself or by introducing unauthorised firmware code.
T.Interrupt	O.Secure_Init, O.Inadvertent	This combination of objectives counters the threat by ensuring that the TOE will not be left in a non-secure state in the event that the token initialization or start-up process is interrupted.
T.UA_Program_Load	O.Import_Code	This objective counters the threat by ensuring that the TOE will control the loading of firmware code and will check to verify that the integrity of the code is preserved prior to each activation of the code.
T.Init_Data_Errors_Accidental	O.Inadvertent, O.Secure_Init, OE.Competence	This combination of objectives counters the threat by ensuring that competent personnel are in place and that the TOE provides a startup procedure that maintains the TOE in a secure state in case of errors made in inputting data during startup.
T.Init_Fail	O.Secure_Init, O.Inadvertent, OE.Recovery	This combination of objectives counters the threat by ensuring that the TOE provides a startup procedure that maintains the TOE in a secure state in case of failure during startup and that there are procedures in place in the environment to recover from such a failure.
T.Forget_PIN	O.Alt_Authentication	This threat is countered by the TOE providing alternatives to user password-based authentication.
T.Failure	O.Backup, OE.Recovery	This threat is countered by the TOE providing a secure means for backing up key material and that there are procedures in place in the environment to recover from a failure using the backup material.

Table 8-3: Mapping of Objectives to Assumptions

Assumptions	Objectives	Rationale
A.Controlled_Access	OE.Physical	The objective on the non-IT environment directly addresses the assumed environmental condition.
A.Legitimate_Applications	OE.Legitimate_Applications	The objective on the non-IT environment directly addresses the assumed environmental condition.
A.Power_Interruption_Protection	OE.Power_Interruption_Protection, OE.Recovery	The combination of objectives on the non-IT environment directly addresses the assumed environmental condition.
A.Disaster_Protection	OE.Disaster_Protection, OE.Recovery	The combination of objectives on the non-IT environment directly addresses the assumed environmental condition.
A.Admin	OE.Authdata, OE.Admin, OE.Recovery, OE.Competence	The combination of objectives on the non-IT environment directly addresses the assumed environmental condition.
A.Host_Connection	OE.Connect, OE.Physical	This combination of objectives on the non-IT environment directly addresses the assumed environmental condition.
A.EM_Emanations	OE.EM_Emanations	The objective on the non-IT environment directly addresses the assumed environmental condition.
A.EMI	OE.EMI	The objective on the non-IT environment directly addresses the assumed environmental condition.

Table 8-4: Necessity of Security Functional Requirements

Security Functional Requirement	Necessitated by:
FIA_ATD.1	O.I&A
FIA_UID. 1	O.I&A, O.Restrict_Actions_Before_Authentication
FIA_UAU.1	O.I&A, O.Restrict_Actions_Before_Authentication
FIA_UAU.5	O.I&A, O.Multi-Person_Control_of_Sensitive_Functions, O.Alt_Authentication
FIA_AFL.1 (SO & Token User)	O.Login_Limit
FIA_USB.1	O.I&A
FTP_TRP.1	O.TrustedPath
FIA_SOS.1	O.Auth_Data_Protect
FIA_SOS.2	O.Auth_Data_Protect
FDP_ACC.1	O.Access, O.User_Data_Protect
FDP_ACF.1	O.Access, O.User_Data_Protect
FDP_RIP.2	O.Access
FDP_ITC.1	O.Access
FDP_ETC.1	O.Access
FCS_COP.1	O.Random_Generate, O.Crypto_Algorithms
FCS_CKM.1	O.Key_Generate
FCS_CKM.3	O.Access
FCS_CKM.4	O.Key_Destroy
FDP_UCT.1	O.Exchange_Confidentiality
FDP_DAU.2	O.Exchange_Integrity, O.Data_Authenticity
FMT_SMR.2	O.Limitation_of_Privilege, O.Multi-Person_Control_of_Sensitive_Functions, O.Security_Roles

Security Functional Requirement	Necessitated by:
FMT_MOF.1	O.Admin, O.Limitation_of_Privilege
FMT_MSA.1(UAV)	O.User_Sec_Attributes, O.Admin, O.Security_Data_Protect
FMT_MSA.1(SOV)	O.User_Sec_Attributes, O.Admin, O.Security_Data_Protect
FMT_MSA.1(Object Attributes)	O.Object_Sec_Attributes, O.Admin, O.Limitation_of_Privilege, O.Security_Data_Protect
FMT_MSA.2	O.Object_Sec_Attributes, O.User_Sec_Attributes
FMT_MSA.3	O.Object_Sec_Attributes, O.Self_Protect
FMT_MTD.1	O.Object_Sec_Attributes, O.User_Sec_Attributes, O.Admin, O.Limitation_of_Privilege, O.Security_Data_Protect
FPT_AMT.1	O.Admin, O.Secure_Init
FPT_FLS.1	O.Inadvertent
FPT_RVM.1	O.Self_Protect
FPT_SEP.1	O.Self_Protect
FPT_TST.1	O.Secure_Init
FTP_ITC.1	O.Import_Code
FDP_ITT.1	O.User_Data_Protect
FPT_ITT.1	O.Security_Data_Protect
FPT_PHP.1	O.Tamper_Evidence
FRU_FLT.1	O.Inadvertent
FPT_RCV.1	O.Inadvertent
FDP_LUNA_BKP.1	O.Backup

Table 8-5: Mapping of Security Functional Requirements to Objectives

Objectives	Security Functional Requirements	Rationale
O.I&A	FIA_UID.1, FIA_UAU.1, FIA_UAU.5, FIA_USB.1, FIA_ATD.1	This combination of SFRs meets the objective by requiring identification and authentication of users.
O.Auth_Data_Protect	FIA_SOS.1, FIA_SOS.2	This combination of SFRs meets the objective by requiring that authentication data cannot be easily guessed by an attacker.
O.TrustedPath	FTP_TRP.1	This SFR meets the objective by requiring that a trusted path is provided for the entry of authentication data and other critical security parameters.
O.Login_Limit	FIA_AFL.1	This SFR meets the objective by requiring that the TOE enforce limits on the number of failed login attempts allowed.
O.Restrict_Actions_Before_ Authentication	FIA_UID.1, FIA_UAU.1	This combination of SFRs meets the objective by requiring that the TOE limit a user's action before I&A and the user be identified and authenticated before he/she is allowed to take any further action.
O.Access	FDP_ACC.1, FDP_ACF.1, FDP_ETC.1, FDP_ITC.1, FCS_CKM.3, FDP_RIP.2	This combination of SFRs meets the objective by requiring that the TOE enforce an access control policy and prevent the access control policy from being bypassed by preventing residual data from being accessed directly from the TOE's memory.
O.Object_Sec_Attributes	FMT_MTD.1, FMT_MSA.1(Object Attributes), FMT_MSA.2, FMT_MSA.3	This combination of SFRs meets the objective by requiring that the TOE implement a set of security attributes for objects that is used to enforce the security policy. The attributes must be set with secure values and the TOE must provide restrictive default values as required by the Fixed Policy Vector.
O.User_Sec_Attributes	FMT_MTD.1, FMT_MSA.1(UAV), FMT_MSA.1(SOV), FMT_MSA.2	This combination of SFRs meets the objective by requiring that the TOE implement a set of security attributes for subjects that is used to enforce the security policy. The attributes must be set with secure values.
O.User_Data_Protect	FDP_ACC.1, FDP_ACF.1, FDP_ITT.1	This combination of SFRs meets the objective by requiring that the TOE enforce an access control policy to protect

Objectives	Security Functional Requirements	Rationale
		user data stored and internally transferred within the TOE.
O.Random_Generate	FCS_COP.1	This SFR meets the objective by requiring that the TOE provide a random number generation capability in accordance with an accepted standard.
O.Crypto_Algorithms	FCS_COP.1	This SFR meets the objective by requiring that the TOE provide cryptographic algorithms in accordance with accepted standards.
O.Key_Generate	FCS_CKM.1	This SFR meets the objective by requiring that the TOE provide key generation in accordance with accepted standards.
O.Key_Destroy	FCS_CKM.4	This SFR meets the objective by requiring that the TOE provide key destruction in accordance with accepted standards.
O.Key_Zeroize	FCS_CKM.4	This SFR meets the objective by requiring that the TOE provide key zeroization in response to a logical brute force attack.
O.Exchange_Confidentiality	FDP_UCT.1	This SFR meets the objective by requiring that the TOE provide a means to protect an object from unauthorised disclosure when exported from the token.
O.Exchange_Integrity	FDP_DAU.2	This SFR meets the objective by requiring that the TOE provide a means to protect an object from unauthorised modification when exported from the token.
O.Data_Authenticity	FDP_DAU.2	This SFR meets the objective by requiring that the TOE provide a means to ensure the validity of a data object and to identify the originator of the object.
O.Admin	FMT_MSA.1(UAV), FMT_MSA.1(SOV), FMT_MSA.1(Object Attributes), FMT_MTD.1, FMT_MOF.1, FPT_AMT.1	This combination of SFRs meets the objective by requiring that the TOE provide the means to administer the security features and attributes.
O.Limitation_of_Privilege	FMT_MOF.1, FMT_MSA.1(Object Attributes), FMT_MTD.1, FMT_SMR.2	This combination of SFRs meets the objective by requiring that the TOE provide the means to administer the security features and attributes in a way that limits the privileges to perform certain actions to authorised roles.

Objectives	Security Functional Requirements	Rationale
O.Multi-Person_Control_of_Sensitive_Functions	FMT_SMR.2, FIA_UAU.5	This combination of SFRs meets the objective by requiring that the TOE provide the means to require multi-person control to perform certain critical actions.
O.Security_Roles	FMT_SMR.2	This SFR meets the objective by requiring that the TOE provide the means to administer separate security roles.
O.Security_Data_Protect	FMT_MTD.1, FMT_MSA.1(UAV), FMT_MSA.1(SOV), FMT_MSA.1(Object Attributes), FPT_ITT.1	This combination of SFRs meets the objective by requiring that the TOE protect security attributes stored and internally transferred within the TOE.
O.Secure_Init	FPT_AMT.1.1, FPT_TST.1	This combination of SFRs meets the objective by requiring that the TOE ensure that it is in a secure state on startup.
O.Self_Protect	FPT_RVM.1, FPT_SEP.1, FMT_MSA.3	This combination of SFRs meets the objective by requiring that the TOE provide the means to ensure that security policy enforcement is not bypassed and that the security functions are provided in a separate operating space that is not accessible by untrusted subjects.
O.Import_Code	FTP_ITC.1	This SFR meets the objective by requiring that the TOE provide a trusted channel for the import of code from a trusted source.
O.Inadvertent	FPT_FLS.1, FPT_RCV.1, FRU_FLT.1	This combination of SFRs meets the objective by requiring that the TOE protect itself from compromise resulting from inadvertent errors caused by a failure of the TOE.
O.Tamper_Evidence	FPT_PHP.1	This SFR meets the objective by requiring that the TOE provide evidence of physical tampering.
O.Alt_Authentication	FIA_UAU.5	This SFR meets the objective by requiring that the TOE provide means of authentication other than user-entered password.
O.Backup	FDP_LUNA_BKP.1	This SFR meets the objective by requiring that the TOE provide a secure means to backup cryptographic material.

Table 8-6: Dependency Rationale for Security Functional Requirements

Security Functional Requirement	Dependencies	Rationale
FDP_DAU.2	FIA_UID.1	Met by inclusion of FIA_UID.1 as SFR
FCS_CKM.1	[FCS_CKM.2 Cryptographic key distribution or FCS_COP.1 Cryptographic operation] FCS_CKM.4 Cryptographic key destruction FMT_MSA.2 Secure security attributes	Met by inclusion of FCS_COP.1, FCS_CKM.4 and FMT_MSA.2 as SFRs
FCS_CKM.3	[FDP_ITC.1 Import of user data without security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction FMT_MSA.2 Secure security attributes	Met by inclusion of FDP_ITC.1 and FCS_CKM.1, FCS_CKM.4 and FMT_MSA.2 as SFRs
FCS_CKM.4	[FDP_ITC.1 Import of user data without security attributes or FCS_CKM.1 Cryptographic key generation] FMT_MSA.2 Secure security attributes	Met by inclusion of FDP_ITC.1 and FCS_CKM.1 and FMT_MSA.2 as SFRs
FCS_COP.1	[FDP_ITC.1 Import of user data without security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction FMT_MSA.2 Secure security attributes	Met by inclusion of FDP_ITC.1 and FCS_CKM.1, FCS_CKM.4 and FMT_MSA.2 as SFRs

Security Functional Requirement	Dependencies	Rationale
FDP_ACC.1	FDP_ACF.1 Security attribute based access control	Met by inclusion of FDP_ACF.1
FDP_ACF.1	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialization	Met by inclusion of FDP_ACC.1 and FMT_MSA.3
FDP_ETC.1	FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control	Met by inclusion of FDP_ACC.1
FDP_ITC.1	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FMT_MSA.3 Static attribute initialization	Met by inclusion of FDP_ACC.1 and FMT_MSA.3
FDP_ITT.1	FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control	Met by inclusion of FDP_ACC.1
FDP_UCT.1	[FTP_ITC.1 Inter-TSF trusted channel, or FTP_TRP.1 Trusted path] [FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]	Met by inclusion of FTP_TRP.1 and FDP_ACC.1
FIA_AFL.1	FIA_UAU.1 Timing of authentication	Met by inclusion of FIA_UAU.1
FIA_UAU.1	FIA_UID.1 Timing of identification	Met by inclusion of FIA_UID.1
FIA_USB.1	FIA_ATD.1 User attribute definition	Met by inclusion of FIA_ATD.1
FMT_MOF.1	FMT_SMR.1 Security roles	Met by inclusion of FMT_SMR.2
FMT_MSA.1	[FDP_ACC.1 Subset access control or FDP_IFC.1 Subset information flow control] FMT_SMR.1 Security roles	Met by inclusion of FDP_ACC.1 and FMT_SMR.2

Security Functional Requirement	Dependencies	Rationale
FMT_MSA.2	ADV_SPM.1 Informal TOE security policy model [FDP_ACC.1 Subset access control or FDP_IFC.1 Subset information flow control] FMT_MSA.1 Management of security attributes FMT_SMR.1 Security roles	Met by inclusion of FDP_ACC.1, FMT_MSA.1, FMT_SMR.2 and ADV_SPM.1
FMT_MSA.3	FMT_MSA.1 Management of security attributes FMT_SMR.1 Security roles	Met by inclusion of FMT_MSA.1 and FMT_SMR.2
FMT_MTD.1	FMT_SMR.1 Security roles	Met by inclusion of FMT_SMR.2
FMT_SMR.2	FIA_UID.1 Timing of identification	Met by inclusion of FIA_UID.1
FPT_FLS.1	ADV_SPM.1 Informal TOE security policy model	Met by provision of Informal Security Policy Model
FPT_PHP.1	FMT_MOF.1 Management of security functions behaviour	Met by inclusion of FMT_MOF.1
FPT_RCV.1	FPT_TST.1 TSF testing AGD_ADM.1 Administrator guidance ADV_SPM.1 Informal TOE security policy model	Met by inclusion of FPT_TST.1, AGD_ADM.1 and ADV_SPM.1.
FPT_TST.1	FPT_AMT.1 Abstract machine testing	Met by inclusion of FPT_AMT.1.
FRU_FLT.1	FPT_FLS.1 Failure with preservation of secure state	Met by inclusion of FPT_FLS.1.

Table 8-7: Mapping of IT Security Functions to IT Security Requirements and Security Functional Requirements

IT Security Function	TSS Reference	IT Security Requirement	CC Requirement Title	CC Functional Component
User Identification and Authentication	6.2.1	Identification of users	Timing of identification	FIA_UID. 1
		Authentication of users	Timing of authentication	FIA_UAU.1
		Multiple Authentication Mechanisms	Multiple authentication mechanisms	FIA_UAU.5
		Limits on repeated login failures (e.g. enforcement of lockout or time delay)	Authentication failure handling	FIA_AFL.1
		User-Token Session Binding	User subject binding	FIA_USB.1
	6.2.1.1	Generation of M of N Secrets	TSF generation of secrets	FIA_SOS.2
Trusted Path	6.2.2	Trusted path for logon	Trusted path	FTP_TRP.1
Authentication data selection	6.2.3	Controls on selection of user-generated Passwords (e.g. minimum length, password filters, password history)	Verification of secrets	FIA_SOS.1
		Automated generation of passwords by TOE	TSF generation of secrets	FIA_SOS.2
User account data	6.2.4	Controls over creation of user accounts and modifications to user account status.	User attribute definition	FIA_ATD.1
Token Access Control Policy	6.2.5	Scope of policy (subjects, objects and operations covered by the policy)	Subset access control	FDP_ACC.1
		Rules governing access by subjects to objects	Security attribute based access control	FDP_ACF.1
Object Re-Use	6.2.6	Protection of residual information in token memory	Full residual information protection	FDP_RIP.2

IT Security Function	TSS Reference	IT Security Requirement	CC Requirement Title	CC Functional Component
Cryptographic Material Protection	6.2.7	Import of data without security attributes	Imported user data without security attributes	FDP_ITC.1
		Export of data without security attributes	Exported user data without security attributes	FDP_ETC.1
Cryptography	6.2.8	Cryptographic Operations	Cryptographic operations	FCS_COP.1
Cryptographic Material Management	6.2.9	Key Generation	Cryptographic key generation	FCS_CKM.1
		Key Access	Cryptographic key access	FCS_CKM.3
		Key Destruction	Cryptographic key destruction	FCS_CKM.4
Data Exchange	6.2.10	Data Confidentiality	Basic data exchange confidentiality	FDP_UCT.1
		Integrity and authenticity of exchanged information	Data authentication with identity of guarantor	FDP_DAU.2
Security Function Management	6.2.11	Management of Security Roles	Restrictions on security roles	FMT_SMR.2
		Management of Security Functions	Management of security functions behaviour	FMT_MOF.1
Security Data Management	6.2.12	Management of Security Attributes	Management of security attributes	FMT_MSA.1
			Secure security attributes	FMT_MSA.2
			Static attribute initialization	FMT_MSA.3
		Management of TSF data	Management of TSF data	FMT_MTD.1
Logical Protection of Security Functions	6.2.13	Abstract Machine Test	Abstract machine testing	FPT_AMT.1
		Fail-safe Behaviour	Failure with preservation of secure state	FPT_FLS.1

IT Security Function	TSS Reference	IT Security Requirement	CC Requirement Title	CC Functional Component
		Reference Monitor	Non-bypassibility of the TSP	FPT_RVM.1
		Separation	TSF domain separation	FPT_SEP.1
		Self-test	TSF testing	FPT_TST.1
		Firmware load/update	Inter-TSF confidentiality during transmission	FTP_ITC.1
Token Cloning	6.2.14	Intra-TOE User Data Transfer	Basic internal transfer protection	FDP_ITT.1
	6.2.14	Intra-TOE TSF Data Transfer	Basic internal TSF data transfer protection	FPT_ITT.1
	6.2.14	Generation of cloning domain identifier	TSF generation of secrets	FIA_SOS.2
Physical Protection of Security Functions	6.2.15	Tamper Evidence	Passive detection of physical attack	FPT_PHP.1
Failure handling	6.2.16	Maintenance of TOE operation in event of failures (fault tolerance)	Degraded fault tolerance	FRU_FLT.1
Backup	6.2.17	Failure recovery	Manual recovery	FPT_RCV.1
		Token backup	LUNA backup	FDP_LUNA_BKP.1

Table 8-8: Mapping of Security Functional Requirements to IT Security Functions

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Timing of identification	FIA_UID. 1	5.1.1.2	User Identification and Authentication	6.2.1	The security function satisfies the SFR by allowing a user to perform a specified set of actions before identification and by requiring the user to be successfully identified before allowing the user to perform any other actions on the token.
Timing of authentication	FIA_UAU.1	5.1.1.3	User Identification and Authentication	6.2.1	The security function satisfies the SFR by allowing a user to perform a specified set of actions before authentication and by requiring the user to be successfully authenticated before allowing the user to perform any other actions on the token.
Multiple authentication mechanisms	FIA_UAU.5	5.1.1.4	User Identification and Authentication	6.2.1	The security function satisfies the SFR by providing multiple authentication mechanisms including PED key, PED key and PED PIN and M of N.
Authentication failure handling	FIA_AFL.1	5.1.1.5 & 5.1.1.6	User Identification and Authentication	6.2.1	The security function satisfies the SFR by detecting when the maximum number of login failures occur (3 for SO, set in the TPV for Token User) and performing on of the following: Zeroize the device in the case of SO authentication failure Remove the user and zeroize the user's memory space, if a Token User authentication failure.
User subject binding	FIA_USB.1	5.1.1.9	User Identification and Authentication	6.2.1	The security function satisfies the SFR by specifying that the user identity be bound to the subject (session) acting on behalf of the user by including the UAV data within the session state.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Trusted path	FTP_TRP.1	5.1.1.10	Trusted Path	6.2.2	The security function satisfies the SFR by requiring the use of a logically distinct trusted path via the PED and dedicated serial port.
Verification of secrets	FIA_SOS.1	5.1.1.7	Authentication data selection	6.2.3	The security function satisfies the SFR by requiring that PIN values meet the minimum and maximum length constraints established by the SO via the TPV.
TSF generation of secrets	FIA_SOS.2	5.1.1.8	Authentication data selection	6.2.1.1, 6.2.3, 6.2.14	The security functions satisfy the SFR by generating random authentication data of the required lengths for each of the functions for which they are required.
User attribute definition	FIA_ATD.1	5.1.1.1	User account data	6.2.4	The security function satisfies the SFR by maintaining the required list of security attributes within the UAV for each Token User.
Subset access control	FDP_ACC.1	5.1.2.1	Token Access Control Policy	6.2.5	The security function satisfies the SFR by enforcing the Token Access Control policy on subjects (sessions), objects and a set of controlled operations.
Security attribute based access control	FDP_ACF.1	5.1.2.2	Token Access Control Policy	6.2.5	The security function satisfies the SFR by enforcing the TAC Policy based on the specified sets of subject and object attributes. The access rules for subjects, objects and operations are as given by table 5-2.
Full residual information protection	FDP_RIP.2	5.1.2.3	Object Reuse	6.2.6	The security function satisfies the SFR by ensuring that the information content of resources is made unavailable when the resource is re-allocated.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Imported user data without security attributes	FDP_ITC.1	5.1.3.1	Cryptographic Material Protection	6.2.7	The security function satisfies the SFR by enforcing the TAC when data is imported through an Unwrap operation. The TSF ignores any security-related attributes that may have been associated with the imported object and sets the object's attributes to the appropriate values for its type and, in particular, the CKA_SENSITIVE attribute is always set.
Exported user data without security attributes	FDP_ETC.1	5.1.3.2	Cryptographic Material Protection	6.2.7	The security function satisfies the SFR by enforcing the TAC when data is exported through a Wrap operation. Objects are exported without security-related attributes.
Cryptographic operations	FCS_COP.1	5.1.4.1	Cryptography	6.2.8	The security function satisfies the SFR by providing mechanisms that implement the specified set of cryptographic algorithms in accordance with the appropriate standards.
Cryptographic key generation	FCS_CKM.1	5.1.5.1	Cryptographic Material Management	6.2.9	The security function satisfies the SFR by providing mechanisms for the generation of RSA, DSA and 3DES keys of the specified lengths in accordance with the appropriate standards.
Cryptographic key access	FCS_CKM.3	5.1.5.2	Cryptographic Material Management	6.2.9	The security function satisfies the SFR by providing mechanisms for key storage and access in accordance with the PKCS #11 standard.
Cryptographic key destruction	FCS_CKM.4	5.1.5.3	Cryptographic Material Management	6.2.9	The security function satisfies the SFR by providing mechanisms that destroys keys in accordance with the PKCS #11 and FIPS 140-1 Level 3 standards.
Basic data exchange confidentiality	FDP_UCT.1	5.1.6.1	Data Exchange	6.2.10	The security function satisfies the SFR by enforcing the TAC to protect transmitted and received objects from unauthorised disclosure using the Wrap and Unwrap operations.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Data authentication with identity of guarantor	FDP_DAU.2	5.1.6.2	Data Exchange	6.2.10	The security function satisfies the SFR by providing digital signature mechanisms that can be used to guarantee the validity of data objects and verify the identity of the originator who performed the digital signature.
Restrictions on security roles	FMT_SMR.2	5.1.7.2	Security Function Management	6.2.11	The security function satisfies the SFR by supporting the distinction of three roles – Public, Token User and Security Officer under the specified conditions.
Management of security functions behaviour	FMT_MOF.1	5.1.7.1	Security Function Management	6.2.11	The security function satisfies the SFR by restricting the ability to perform the specified security management operations to the SO role.
Management of security attributes	FMT_MSA.1	5.1.8.1 & 5.1.8.2	Security Data Management	6.2.12	The security function satisfies the SFR by enforcing the TAC Policy to restrict the ability to manipulate user and object security attributes as specified.
Secure security attributes	FMT_MSA.2	5.1.8.4	Security Data Management	6.2.12	The security function satisfies the SFR by ensuring that only secure values are accepted for security attributes.
Static attribute initialization	FMT_MSA.3	5.1.8.5	Security Data Management	6.2.12	The security function satisfies the SFR by requiring restrictive values for security attributes that cannot be changed based on the FPV settings.
Management of TSF data	FMT_MTD.1	5.1.8.6	Security Data Management	6.2.12	The security function satisfies the SFR by enforcing the TAC Policy to restrict the ability to manipulate the TPV settings to the SO.
Abstract machine testing	FPT_AMT.1	5.1.9.1	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by running a suite of tests at startup and upon user request to verify the correct operation of the security-relevant aspects of the underlying token hardware.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Failure with preservation of secure state	FPT_FLS.1	5.1.9.2	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by preserving the token in a secure state when the specified failure conditions occur.
Non-bypassibility of the TSP	FPT_RVM.1	5.1.9.3	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by ensuring that TSP enforcement functions are invoked and succeed before each function within the token firmware is allowed to proceed.
TSF domain separation	FPT_SEP.1	5.1.9.4	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by maintaining a separate domain for the execution of the TOE security functions and by separating subject domains by maintaining cryptographic separation of user data, by allowing only one logged in user to be active on the token and by allowing a single thread of execution on the token.
TSF testing	FPT_TST.1	5.1.9.5	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by providing a suite of self-tests to verify the correct operation of the security functions on startup and at the request of an authorised user.
Inter-TSF confidentiality during transmission	FTP_ITC.1	5.1.9.6	Logical Protection of Security Functions	6.2.13	The security function satisfies the SFR by providing a logical trusted channel between a customer token and a separate token containing a firmware image to initially load the customer token or update the firmware on the customer token.
Basic internal transfer protection	FDP_ITT.1	5.1.10.1	Token Cloning	6.2.14	The security function satisfies the SFR by enforcing the TAC Policy to control the cloning of user data objects from one token to another within a cloning domain and to protect the objects from disclosure, modification and loss of use by using the cloning protocol.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
Basic internal TSF data transfer protection	FPT_ITT.1	5.1.10.2	Token Cloning	6.2.14	The security function satisfies the SFR by providing a mechanism as part of token cloning to transfer the TSF data from one token to another within a cloning domain and to protect it from disclosure and modification.
Passive detection of physical attack	FPT_PHP.1	5.1.11.1	Physical Protection of Security Functions	6.2.15	The security function satisfies the SFR by implementing physical security mechanisms that unambiguous evidence of physical tampering and the ability to determine whether physical tampering with security-relevant devices has occurred.
Degraded fault tolerance	FRU_FLT.1	5.1.12.1	Failure Handling	6.2.16	The security function satisfies the SFR by ensuring that the user data protection capabilities are maintained when power failures or data I/O failures occur. The token maintains all Sensitive permanent objects in an encrypted state and, therefore, such failures cannot affect the protection of permanent objects. Volatile objects are wiped from memory when power to the token is lost. Data I/O failures result in suspension of user operations on the token, but data protection capabilities are maintained and the token will return to operation once data I/O is restored in the same state it was prior to the failure.
Manual recovery	FPT_RCV.1	5.1.12.2	Backup	6.2.17	The security function satisfies the SFR by ensuring that the token maintains its secure state in the event of failure or service discontinuity and can be returned to operation in its secure state once the failure has been resolved.

CC Requirement Title	CC Functional Component	ST Reference	IT Security Function	TSS Reference	Rationale
LUNA backup	FDP_LUNA_BKP.1	5.1.12.3	Backup	6.2.17	The security function satisfies the SFR by providing a secure backup capability from a primary token to a backup using the Token Cloning function.

Table 8-9: Assurance Measures

Assurance Class	Assurance Components	Evidence
Class ACM: Configuration management		
	ACM_AUT.1 Partial CM automation	See CR-0274, section 2.2, Configuration Management, for specific references.
	ACM_CAP.4 Generation support and acceptance procedures	See CR-0274, section 2.2, Configuration Management, for specific references. MKS Integrity
	ACM_SCP.2 Problem tracking CM coverage	See CR-0274, section 2.2, Configuration Management, for specific references.
Class ADO: Delivery and operation		
	ADO_DEL.2 Detection of modification	See CR-0274, section 2.2, Delivery and Operation Documents, for specific references.
	ADO_IGS.1 Installation, generation, and start-up procedures	See CR-0274, section 2.2, Delivery and Operation Documents, for specific references.

Assurance Class	Assurance Components	Evidence
Class ADV: Development		
	ADV_FSP.2 Fully defined external interfaces	PKCS #11 plus Chrysalis-ITS extensions See CR-0274, section 2.2, Functional Specification, for specific references.
	ADV_HLD.2 Security enforcing high-level design	See CR-0274, section 2.2, Functional Specification, High Level Design, and Low Level Design, for specific references.
	ADV_IMP.1 Subset of the implementation of the TSF	Firmware code. Subset to be provided in accordance with evaluator's request.
	ADV_LLD.1 Descriptive low-level design	See CR-0274, section 2.2, Low Level Design, for specific references.
	ADV_RCR.1 Informal correspondence demonstration	Correspondence mappings for TSS to Functional Specification, Functional Specification to HLD and HLD to LLD.
	ADV_SPM.1 Informal TOE security policy model	See CR-0274, section 2.2, Security Policy Model, for specific references.

Assurance Class	Assurance Components	Evidence
Class AGD: Guidance documents		
	AGD_ADM.1 Administrator guidance	See CR-0274, section 2.2, Guidance Documents, for specific references. The administration functions are normally carried out by the Security Officer, or possibly a designated Token User, using the Enabler software as the interface. In most cases, these functions will be performed very infrequently.
	AGD_USR.1 User guidance	User guidance documents are not provided because the normal user of the TOE is an application program making function calls to the TOE via the PKCS #11 Cryptographic API. Direct access to the TOE's functions by a human user only occurs in the course of performing administration functions.
Class ALC: Life cycle support		
	ALC_DVS.1 Identification of security measures	See CR-0274, section 2.2, Life Cycle Documentation, for specific references.
	ALC_FLR.2 Flaw reporting procedures	See CR-0274, section 2.2, Life Cycle Documentation, Configuration Management, and Delivery and Operation Documents, for specific references.
	ALC_LCD.1 Developer defined life-cycle model	See document CR-0274, section 2.2, Life Cycle Documentation, for specific references.

Assurance Class	Assurance Components	Evidence
	ALC_TAT.1 Well-defined development tools	See document CR-0274, section 2.2, Life Cycle Documentation, for specific references.
Class ATE: Tests		
	ATE_COV.2 Analysis of coverage	See document CR-0274, section 2.2, Developer's Tests, for specific references.
	ATE_DPT.1 Testing: high-level design	See document CR-0274, section 2.2, Developer's Tests, for specific references.
	ATE_FUN.1 Functional testing	See document CR-0274, section 2.2, Developer's Tests, for specific references.
	ATE_IND.2 Independent testing - sample	To be performed by evaluator.
Class AVA: Vulnerability assessment		
	AVA_MSU.2 Validation of analysis	See document CR-0274, section 2.2, Misuse Documentation, for specific references.
	AVA_SOF.1 Strength of TOE security function evaluation	See document CR-0274, section 2.2, Strength of TOE Security Functions, for specific references.

Assurance Class	Assurance Components	Evidence
	AVA_VLA.2 Independent vulnerability analysis	See document CR-0274, section 2.2, Vulnerabilities Documentation

Appendix A

References and Glossary of Abbreviations

References

Document Number	Revision	Author	Title
ISO/IEC 15408-1	V2.1, August 1999		Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and General Model
ISO/IEC 15408-2	V2.1, August 1999		Common Criteria for Information Technology Security Evaluation – Part 2: Security Functional Requirements
ISO/IEC 15408-3	V2.1, August 1999		Common Criteria for Information Technology Security Evaluation – Part 3: Security Assurance Requirements
	January 1994	U.S. Department of Commerce and National Institute of Standards and Technology	FIPS PUB 140-1: Security Requirements for Cryptographic Modules
	Version 2.01, September 1997	RSA Laboratories	PKCS #11: Cryptographic Token Interface Standard
	Version 1.5, November 1993	RSA Laboratories	PKCS #5: Password-Based Encryption Standard
	Version 2.0, October 1998	RSA Laboratories	PKCS #1: RSA Cryptography Standard
		IETF	RFC 2510, PKIX-Certificate Management Protocol
	February, 1995	PCMCIA/JEIDA	PC Card Standard
800514		Chrysalis-ITS, Inc.	Luna2 Interface Control Document (ICD)
800024		Chrysalis-ITS, Inc.	Luna2 Hardware Design Specification
800017		Chrysalis-ITS, Inc.	Luna2 Physical Security Design
800512		Chrysalis-ITS, Inc.	Luna2 Software Functional Description
802509		Chrysalis-ITS, Inc.	Luna CA^3 Security Policies

Glossary of Abbreviations

Shortforms	***Longform Explanation***
CA	Certification Authority
CAST	Symmetric encryption algorithm developed by Carlisle Adams and Stafford Tavares
CC	Common Criteria
CIMS	Certificate Issuing and Management System
COTS	Commercial Off-the-Shelf
CRC	Cyclic Redundancy Check
CSP	Certification Service Provider
DES	Data Encryption Standard
DLL	Dynamic Linked Library
DSA	Digital Signature Algorithm
EAL	Evaluation Assurance Level
FIPS	Federal Information Processing Standard
FPV	Fixed Policy Vector
ICD	Interface Control Document
IETF	Internet Engineering Task Force
IT	Information Technology
OTP	One-Time Pad
PBE	Password-Based Encryption
PC	Personal Computer
PCMCIA	Personal Computer Memory Card Industry Association
PED	PIN Entry Device
PIN	Personal Identification Number
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
PKIX-CMP	IETF PKI Certificate Management Protocol
RAM	Random Access Memory
RNG	Random Number Generator (Generation)
RSA	Asymmetric algorithm developed by Rivest, Shamir and Adleman
SFP	Security Function Policy

SFR	Security Functional Requirement
SHA	Secure HASH Algorithm
SO	Security Officer
SoF	Strength of Function
ST	Security Target
TAC	Token Access Control
TOE	Target of Evaluation
TPV	Token Policy Vector
TSF	TOE Security Function
UAV	User Authorization Vector
VA	Validation Authority