SecureNet Limited
ACN 073 665 175

SecureNet TrustedNet Connect

Security Target

For

SecureNet Limited

Version:	2.0 Revision 8
Copy:	1
Release Date:	29 January 2003
Release Authority:	Chris Nicholls

SECURENET™

History of Changes

Date	Version	Change Summary	Author
	0.1	Initial Draft for internal review	Chris Moyle
	0.2	Second Draft for internal review	Chris Moyle
8/11/00	0.3	Draft incorporating client comments	Tony Hall
10/11/00	0.4	Draft incorporating client comments, Final draft prior to submission to DSD	Tony Hall
17/11/00	0.5	Incorporating comments from Ed Zuk.	Tony Hall
24/11/00	1.0	Release to DSD	Tony Hall
18/12/00	1.1	Addressing DSD comments	Michael Sterling, Graeme Mahon
21/02/01	1.2	Addressing comments from Ed Zuk.	Tony Hall Graeme Mahon
25/09/01	1.3	Addressing EORs	Ed Zuk
5/03/02	1.4	Addressing EORs, and correspondence to SPM.	Ed Zuk
22/07/02	2.0.5	Change version numbering to be consistent with configuration management plan; qualify conditions for SoF claim, update supported operating systems.	Ed Zuk
27/08/02	2.0.6	Update version of Connect and remove cryptography related functions from strength of function analysis	Ed Zuk
13/11/02	2.0.7	Minor changes to TSF3	Ed Zuk
29/01/03	2.0.8	Added documentation version numbers	Mike Lee

Last Printed:	**13/11/2002 17:23**
Last Saved:	**29/01/2003 15:55**
Last Saved By:	**Michael Lee**
Current Version:	**2.0 Revision 8**

Table Of Contents

List of Tables

SECURENET™

List of Figures

Conventions

The notation, formatting and conventions used in this Security Target are consistent with those used in Version 2.1 of the Common Criteria (CC). The CC allows several operations to be performed on functional requirements; *refinement*, *selection*, *assignment* and *iteration* are defined in Section 2.1.4 of Part 2 of the CC.

Terminology

In the CC, many terms are defined in Section 2.3 of Part 1. The following terms are a subset of those definitions. They are listed here to aid the user of the Security Target.

CC	Common Criteria
EAL	Evaluation Assurance Level
PP	Protection Profile
SAR	Security Assurance Requirement
SF	Security Function
SFP	Security Function Policy
SFR	Security Functional Requirement
SOF	Strength of Function
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functions
TSC	TSF Scope of Control
TSFI	TSF Interface
TSP	TOE Security Policy
TSS	TOE Summary Specification
TTP	Trusted Third Party

The following terminology and abbreviations specific to the TOE and its environment is also provided to aid the user of the Security Target.

API	Application Programming Interface.
Cardlet	Within the context of this security target, cardlets refer to smart card application code and data. Cardlets are loaded into MULTOS smart cards where the code can be executed.
Certificate	A binding between an entity's public key and one or more attributes relating to that entity, created by a trusted third party. The certificate provides assurance that the public key belongs to the identified entity and the trusted third party protects this binding through a digital signature. Certificates are coded according to ITU recommendation X.509.

CRL	Certificate Revocation List: a list of certificates which are no longer valid. This list is produced and digitally signed by the same trusted third party that generates the certificates.
Cryptography	The study of codes and ciphers.
Cryptanalysis	The process of breaking cryptographic algorithms and protocols.
CSP	Cryptographic Service Provider.
IKE	Internet Key Exchange: an IETF specification for exchanging keys in order to establish IPsec connections.
IPsec	Internet Protocol Security: an extension to the IEFT Internet Protocol that allows IP connections to be protected for integrity and confidentiality
MD5	Message Digest 5: a one-way hash function that reduces a message (the header and payload), to a 128-bit digest value. The recipient of a message with a digest can perform the same hash operation on the message and compare the result with the received digest to gain confidence that the message has not been changed in transit.
MIME	Multipart Internet Mail Exchange: a set of IETF specifications that provides a way to exchange electronic mail with different character sets and multimedia data.
PC/SC	A specification defining the interaction between personal computers and smart card terminals (readers).
PKI	Public Key Infrastructure: a security infrastructure that uses trusted third parties to associate public keys with users and manage those keys.
SHA-1	Secure Hash Algorithm 1: a one-way hash function similar to MD5, but it produces a 160-bit digest value. SHA1 is slower to calculate than MD5, but provides less chance of collisions.
S/MIME	Secure MIME: an extension to the MIME specification that allows electronic mail to be exchanged securely.
SSL	Secure Sockets Layer: a commonly used secure communications protocol.

Document Organisation

Section 1 provides the labelling and descriptive information necessary to control and identify the ST and the TOE to which it refers.

Section 2 describes the TOE as an aid to understanding its security requirements, and shall address the product or system type.

Section 3 describes the security aspects of the environment in which the TOE is intended to be used, and the manner in which it is expected to be employed.

Section 4 defines the security objectives for both the TOE and the TOE environment.

Section 5 contains the functional and assurance requirements derived from the Common Criteria, Parts 2 and 3 respectively, that must be satisfied by the TOE.

Section 6 The TOE summary specification defines the instantiation of the security requirements for the TOE. This specification provides a description of the security functions and assurance measures of the TOE that meet the TOE security requirements.

Section 7 This part of the ST presents the evidence used in the ST evaluation. This evidence supports the claims that the ST is a complete and cohesive set of requirements, that a conformant TOE would provide an effective set of IT security countermeasures within the security environment, and that the TOE summary specification addresses the requirements.

References

Common Criteria	Common Criteria for Information Technology Security Evaluation, 15 November 1998, Version 15408 FDIS
MULTOS Certification Report	AISEP Certification Report Number 2000/13, KeyCorp Ltd, MULTOS Version 4.02 (Release 1N'-AMD), Issue 1.0, July 2000
PKCS#11	Cryptographic Token Interface Standard, v2.10, RSA Laboratories (http://www.rsalabs.com), December 1999

1. INTRODUCTION

1.1 Identification

Title:	SecureNet TrustedNet Connect Security Target
Authors:	CSC Australia Pty Ltd
Last Updated:	29/01/2003 15:55
Assurance Level:	EAL4
CC Version:	2.1 Final
TOE Version:	SecureNet TrustedNet Connect v2.0.4.9

1.2 Security Target Overview

A digital identity (digital ID) is the combination of a X.509 Certificate and a key pair. TrustedNet Connect provides a way for a user to use their digital IDs in a Public Key Infrastructure securely.

The TOE consists of two major sections: software that is loaded and executed on a smart card (referred to as *cardlets*) and software for a host that interacts with the smart card. The cardlets protect, manage and control cryptographic keys stored on the smart card. The host software provides facilities to control and manage digital IDs, and provides APIs (Application Programming Interfaces). The APIs link the card to end-user applications so that the applications can access the digital IDs in a controlled manner to realise security services such as:

- Signing and decrypting S/MIME compliant e-mail;
- Signing and decrypting files;
- Securing access to web servers using client-side SSL;
- User authentication;
- Transactions signing;
- VPN client authentication via IPsec IKE;
- Secure log on to computers through a smart card.

TrustedNet Connect supports the following types of key pairs:

- Key pairs that can be used to decrypt session keys;
- Key pairs that can be used to sign data (or a cryptographic digest of the data);
- The key pairs that can be used either to decrypt session keys or to sign data;
- Key pairs that can be used to sign data in a manner that supports non-repudiation services;
- Key pairs that can be used to sign or decrypt any data.

Many existing applications including web browsers, e-mail and PKI VPN clients can utilise TrustedNet Connect cryptographic services via either the RSA Inc PKCS #11 interface or the Microsoft CSP interface.

It is assumed the smart card is evaluated to the same level of assurance (or greater) as the cardlets. In this case it is the MULTOS 4.02 (Release 1N'-AMD) smart card, which has been evaluated to ITSEC E6 level, or the MULTOS 4.06 (Release 1Q) smart card on the Infineon SLE66CX320P smart card module, for which evaluation is pending.

1.3 CC Conformance Claim

The TOE is conformant with Parts 2 and 3 of the CC (Version 2.1), and will conform to EAL4 measures in Part 3.

2. TOE DESCRIPTION

This part of the ST describes the TOE as an aid to understanding its security requirements, and addresses the product or system type. The scope and boundaries of the TOE are described in general terms both in a physical way (hardware and/or software components/modules) and a logical way (IT and security features offered by the TOE).

The TOE description provides a context for the evaluation. The information presented in the TOE description will be used in the course of the evaluation to ensure that it is consistent. If the TOE is a product or system whose primary function is security, this part of the ST may be used to describe the wider application context within which such a TOE fits.

2.1 Product Type

TrustedNet Connect is designed for secure provision of key pair operations within a Public Key Infrastructure.

Public key encryption algorithms use two different keys (a *key pair*) to encrypt and decrypt data. The keys are related to each other mathematically so that, even if one of the keys is known, it is not feasible to determine the other. This characteristic is exploited by publishing one of the keys in a key pair – the public key. The other key is the private key and only the key pair owner knows its value. A public key can be used to encrypt data so that only the key pair owner can decrypt the data with the private key. Alternatively, the key pair owner can sign data with the private key so that anyone with the public key can verify that the key pair owner signed the data and that the data is unchanged since it was signed.

Typically, management of public keys is carried out within a Public Key Infrastructure (PKI), where a certification authority acts as a trusted third party to bind public keys to the identity of key pair owners. This binding takes the form of a certificate that the certification authority signs with its private key. The combination of a private key and a corresponding certificate constitutes a digital identity, since the private key operations can be performed only by the user identified in the certificate. Usually, PKI also supports other key management functions, such as key revocation and the distribution of public keys through public directories. Certification authorities publish policies that govern the use of certificates and the public keys contained within them.

TrustedNet Connect is designed to protect users' key pairs within a PKI. However, even though the TOE is designed to support functions typically required for a private key store in a PKI, it is **not** reliant upon it, and is useful in situations where public key cryptography is used outside the scope of a PKI. One such example is when public key cryptography is used to establish a secure communication link and key management is based on the manual loading (and authentication) of public keys.

A typical operation of the TOE within a PKI is depicted in Figure 1. The TOE (indicated by the dotted line in the figure) comprises of:

– Host software (TrustedNet Connect on a user's PC);

– The cardlets on a MULTOS smart card.

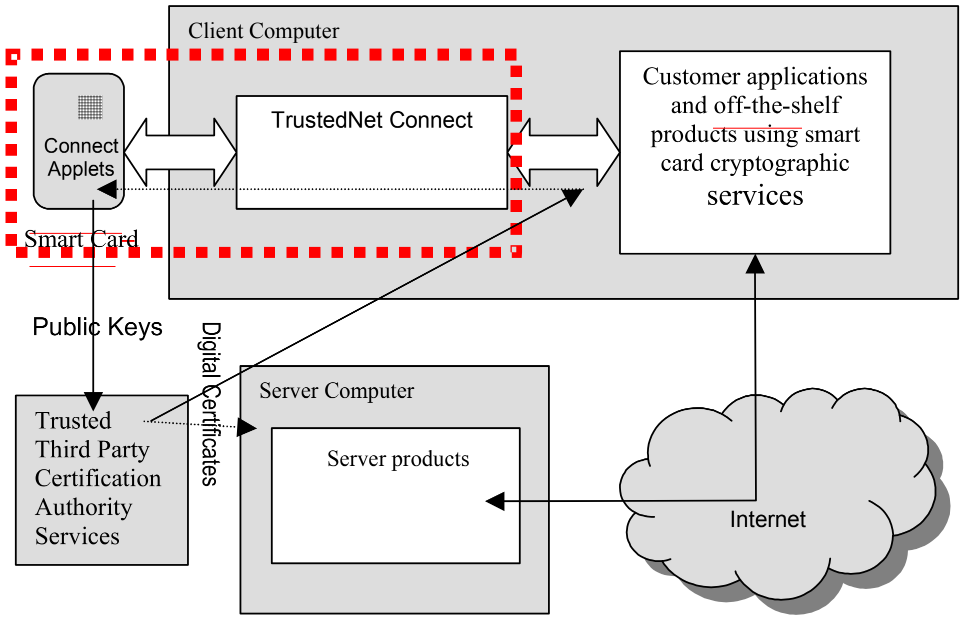
Client Computer
Connect Applets
Smart Card
TrustedNet Connect
Customer applications and off-the-shelf products using smart card cryptographic services
Public Keys
Digital Certificates
Trusted Third Party Certification Authority Services
Server Computer
Server products
Internet

Figure 1 – Logical Scope

The primary role for the TOE is to store and control key pairs on behalf of the key pair owners. The key pairs are stored on smart cards. Public keys are conveyed between smart cards and certification authorities who must then associate them with the key pair owners – the smart card holder.

The certificates that the certification authority produces must be available at the server computer that relies on the public key for communication security. Certificates are inherently protected from modification and can be sent to the server from the client computer. For convenience, the TOE allows certificates to be stored on the smart card; allowing users to use their digital identities from any computer where TrustedNet Connect is installed.

TrustedNet Connect implements a number of APIs – including RSA Inc.'s PKCS#11 v2.1 and Microsoft's CSP Interface – that provide cryptographic services associated with the private key stored on a smart card. These APIs expose smart card functions for data decryption and digital signature operations, and they comply with published specifications which a range of *commercial off-the-shelf* products has adopted. Compatible products exist for secure e-mail, SSL services, file encryption, transaction signing, IPsec clients, and smart card log on to computer systems.

The TOE is designed to be installed on workstations and accessed by interactive users. The operation of the TOE requires that a user must be present to insert a smart card, and it is expected that the user controls the host and applications through an interactive session. In some circumstances it may be advantageous for TrustedNet Connect to provide cryptographic services to servers. For these installations, appropriate precautions must be taken to ensure that all executables on the server can be trusted not to misuse the TOE and some features such as non-repudiation signing will not be available.

The TOE may be installed on a PC that is shared by more than one user from the same organisation. It is assumed that the users are all part of the same security domain and therefore share common security policies and objectives. The TOE must be installed according to the installation guide so that a user cannot interfere with another user's operation of the TOE. It is assumed that an administrator installs the TOE on a host computer that is set up to protect applications from interference by normal users. Administrators must be trusted not to abuse their privileges and not to attack the TOE.

PKI is concerned with security for distributed applications, so it is expected that the host computer is connected to a public network such as the Internet. This exposes the host to potential attacks from remote agents. Furthermore, as part of a PKI, the TOE may be relied upon to protect e-commerce transactions of considerable value. Consequently, the host must be set up and managed to resist attacks originating from public networks, and it is expected that the host is protected with firewall and anti-virus products.

Many Internet applications use Web pages with *active content*. This offers a convenient catalyst for Internet-based transactions and is very important for electronic commerce. Active content pages contain code that is executed in the user’s environment, so some protection is needed to ensure that code does not compromise the integrity of the user’s environment. Active content is often constrained to run in a controlled environment that prohibits access to sensitive resources, however restrictions are relaxed if a trusted author signed the active content. One of the functions for active content is to obtain endorsement of transactions from user through digital signatures. The TOE supports this functionality.

The TOE does not affect the computer log on process. However, a program or module such as a GINA for Windows NT, and Windows 2000 smart card log on can take advantage of the services offered through the TOE to enforce a "smart card enabled" log on. Support for the log on process demands that the TOE is set up as a trusted system service as it must be executing before users log in and the operating system relies on it to authenticate users.

The TOE does not distribute keys nor revoke keys directly. It is the responsibility of the application to perform these tasks, usually with the aid of a certification authority. However, the TOE can store certificates and make them available to client applications.

2.2 General TOE Functionality

The TOE is concerned primarily with protecting the confidentiality and use of private key components of users' key pairs. Protection is provided at all stages of a key pair life-cycle from creation or loading, to activation, use and destruction such that the TOE is suitable for use within a PKI.

The typical requirements a PKI has for key pairs are:

- Unique association between a key pair and a user;
- Private key operations: sign and decrypt data;
- Control of the use of private keys;
- Access to the public key component of key pairs; and
- Storage for certificates.

Certification authorities in a PKI create certificates that bind public keys to key holders. Users of certificates rely on this binding for security services such as non-repudiation of the origin of data, secure transmission of data to a user, and user authentication. The TOE allows users to store key pairs on smart cards that protect the keys, allowing users to be associated with public key through possession of smart cards that contain those keys.

To have any value, users' applications must have the ability to perform cryptographic operations on the key pairs. The smart card and the installed cardlets implement these cryptographic operations while the TrustedNet Connect host software implements APIs that allow applications to access these cryptographic functions. The host software implements some cryptographic functions, however these do not operate on private keys. The APIs also have functions for managing the TOE.

In addition to the programmatic interfaces, the TOE offers a Graphical User Interface that allows users to generate and load key pairs, and control security attributes.

Certification authorities create certificates under *certificate policies* that place restrictions on how the keys associated with certificates can be used. Some certificate policies hold key owners liable for anything they sign with their certified key pairs so that parties relying on signatures have non-repudiation. The TOE supports these types of policies as it can restrict the use of private keys to certain mechanisms. The TOE supports a key pair usage attribute that controls which operations (signing, non-repudiation signing, session key decryption, data decryption) are possible on private keys.

Users may specify these restrictions on key pairs to assist them in meeting policy while certification authorities may be interested in determining that the keys cannot be used in violation of the certificate policy.

The public key component of a key pair is a critical component of a certificate. The TOE allows public keys to be read from the smart card, and the resulting certificate stored on the card. Users are able to carry all personal information to access their digital identity in their smart cards offering users portability of their digital identities without any increased exposure of their keys.

Provision has been made for distributing smart cards in various stages of personalisation. According to the requirements of the PKI or users, cards can be distributed without any user-specific information, with partial information (such as the number of supported key pairs and their attributes) or fully personalised. MULTOS smart cards can be issued before applications are loaded since MULTOS provides the security services to protect application loading and to protect applications from interfering with each other. Consequently, smart card application loading is outside the scope of the TOE. The production of a smart card application (cardlet) includes the generation of data structures which issuers may specify. Inclusion of data on smart cards allows Connect to meet a large number of issuer specific requirements.

2.2.1 Security Features

The TOE protects the interests of the users since it implements cryptographic services on behalf of, and in favour of users. Users have an interest in keeping the TOE secure. However, a number of threats from external agents can be identified for the intended operation of the TOE and security features are designed to protect against those threats.

2.2.1.1 Key Management

If an attacker obtains a copy of a user's private key, the attacker is able to impersonate the user and read data encrypted for that user. Weaknesses in the key generation process can lead to attack agents recreating user keys, or drastically reducing the number of keys that have to be searched to find a user's key. The TOE utilises the smart card random number generator to generate random key pairs from a key space sufficiently large enough to result in unique keys and to resist exhaustive searching. The smart card can perform all necessary cryptographic operations on private keys, which avoids generated keys from becoming available outside the smart card. The TOE also makes available the smart card random number generator to applications so that they can generate random session keys.

In some circumstances, it is desirable to have a trusted third party generate users' key pairs. To avoid disclosure of private keys, the third party must be trusted not to disclose the keys generated (except perhaps under an agreed policy), to generate keys that are randomly selected from a sufficiently large key space, and to deliver those keys to users securely. In support, the TOE allows encrypted private keys to be loaded on to the smart card and the keys decrypted at a later time.

2.2.1.2 Strong Cryptography

Cryptanalysis of a set of generated signatures and encrypted data can lead to decryption of data or the generation of additional signatures. Strong cryptographic algorithms are used to ensure the TOE is not susceptible to cryptanalysis. The cardlets build upon MULTOS cryptographic primitives to implement standard cryptographic mechanisms and protocols for session key decryption, digital signature generation for data larger than the cipher block size, and decryption of data larger than the cipher block size. The TOE also implements cryptographic hash algorithms and public key operations for data encryption and signature verification.

2.2.1.3 Authentication and Access Control

The TOE identifies users as owners of particular smart cards. More meaningful identification for TrustedNet Connect client applications may be present in certificates for key pairs stored in a smart card. Users' unique private keys are found on their smart cards which users must insert into the reader for the TOE to perform any private key operation. In addition, password authentication is required to use private keys. Depending on the level of risk associated with unauthorised use of a private key, access control attributes can be set for a private key to require authentication for every cryptographic operation with that key.

Password authentication is performed on smart cards and this protects users in the case where their smart cards are lost or stolen. The card fully protects this password, and the card has a built-in policy of blocking passwords which disables authentication if an incorrect password is entered consecutively a pre-set number of times. This count is maintained regardless of the intervening events (such as removing the card and reinserting in the same or different card reader). The count is cleared when the correct password is entered prior to exhausting the allowed number of consecutive entries.

A blocked password can be unblocked (re-activated) through the use of a secondary code that a card issuer or security manager may know or determine (generation of unblocking codes by card issuers or security managers to unblock a password is outside the scope of the TOE). Limitations similar to those for passwords apply on incorrect entry of the unblocking code, and after a pre-set number of successive unblocking attempts the unblocking facility is disabled.

Administrative actions and supporting systems for determining whether a password should be unblocked are considered outside of the scope of the TOE. However, it is anticipated that the issuer may wish to unblock passwords through a helpdesk after they are satisfied that the cardholder should be allowed to unblock the password. The unblock codes are short enough to be easily conveyed over the telephone, but secure enough to hinder any eavesdroppers from attempting to unblock any password.

Password re-authentication can be specified for every cryptographic operation with a private key. The user can then track all uses of the private key and verify that applications are behaving as expected. This is especially important for web pages with active content that access the TOE as the user can confirm that private keys are only accessed when expected.

2.2.1.4 Sensitive Data Protection

The TOE processes secret information that needs protection. It decrypts confidential data or session keys that protect data, it reads passwords to unlock resources on the smart cards, and it generates random numbers that might be used as cryptographic keys. All this information has to be handled carefully to ensure that the TOE does not expose it to other users. TrustedNet Connect erases such data from their containers before it relinquishes control of the data containers.

The TOE also maintains secure operations in the event that a smart card is removed from the reader or there is power loss to the smart card.

An attacker might attempt to extract private keys from a smart card. Smart cards provide security to prevent physical access to data and the analysis of electrical signals or electromagnetic emanations. All operations on the private key are performed in the smart card, and the security policy implemented in the Digital ID cardlet prohibits reading private keys.

2.2.1.5 Security Attributes

Once set, the TOE does not allow security attributes to be changed. Users can have confidence that the attributes displayed for a key are the ones enforced.

Passwords have the following attributes:

- Minimum length;
- The number of retries allowed to successfully authenticate with the password;
- A maximum number of time a user can be authenticated with the password;
- The number of unblocking operations allowed on the password.

Key pairs have a usage attribute that controls the way in which the private key can be used. Through this attribute the TOE defines five (5) types of key pairs as follows:

- **Encryption Keys**: key pairs for key exchange so that session keys can be securely received. This is typically required in a secure e-mail or file encryption application.
- **Signing Keys**: key pairs for digitally signing, using the protocol specified in PKCS #1 version 2 with the RSA algorithm (up to 1024 bit RSA on-card).
- **Utility Keys**: key pairs for either session key decryption, or for digitally signing data.
- **Unrestricted Keys**: key pairs for decrypting data, which allows direct access to the RSA modulo exponentiation operation for the private key.
- **Non-Repudiation keys**: key pairs for *non-repudiation* signing (up to 1024 bit RSA). Digital signatures for non-repudiation keys are also generated in accordance with PKCS #1 version 2.

With active web pages, there is a risk that data displayed is not the actual data that is sent to the TOE for signing. This is especially serious for keys that provide non-repudiation services. To manage this risk, a non-repudiation key pair has the following special properties:

- The TOE requires user authentication through password entry prior to each generation of a digital signature.
- The software displays the data to be signed for the cardholder to verify. If the software cannot display the contents sensibly because of the MIME type, the contents are saved for display with an external viewer. This is used to give the user confidence in the information being signed. The compatible MIME types are text/plain, application/pkcs7 and application/pkcs10.
- The signature is verified to match the data displayed or saved.
- The smart card maintains a counter for the number of digital signatures formed. The card can issue a secondary signature using a different algorithm (ISO 9798-2) on the signed data and the contents of the counter. This can be used to detect card forgeries since a forged card will quickly have a different count from the original card. This inconsistency can be detected at a verifying host. However, the TOE does not provide the functionality to detect such an inconsistency.

2.2.1.6 Operation of Security Features

A typical sequence of operations for generating and using key pair in the TOE are as follows:

1. A password is set so that the user has complete control of the key that will be generated.
2. The user sets the security attributes that apply for the new key pair.
3. A key pair is generated on the card.
4. The public key is sent to a certification authority that, after validating the identity of the user, returns a certificate. The certification authority may check that the security attributes of the certified key pair comply with its certificate policy. The certificate is distributed to relying partied as appropriate.
5. The certificate is loaded onto the card.
6. The user starts an application that requests the use of a private key, and the user is asked to enter a password for authentication and access to the private key.
7. The card performs the desired cryptographic operation with the private key.

2.3 Scope and Boundaries

2.3.1 Logical

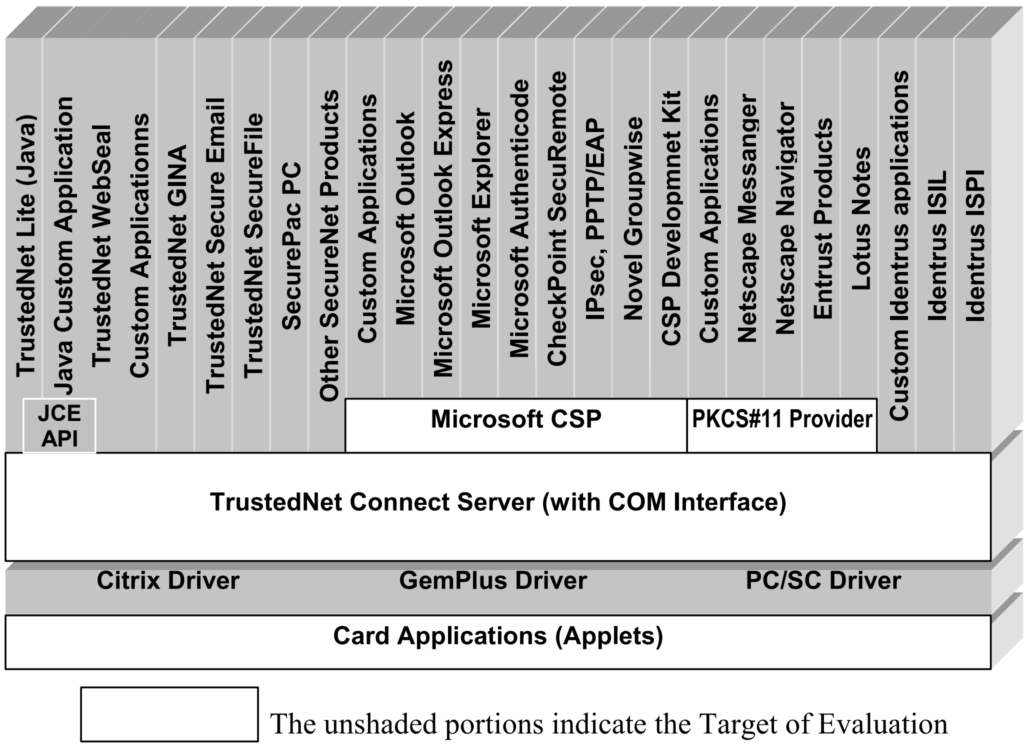

Figure 2 – Logical Environment

2.3.1.1 TOE Security Features Summary

The TOE provides the following security features:

- Functions to sign data with the user's private key;
- Functions to decrypt data with a user's private key;
- Cryptographic key generation;
- Secure loading of private keys;
- Authentication of key pair holders (users);
- Management of keys, passwords and configuration data;
- Control of access to cryptographic keys, including control of cryptographic operations on keys;
- Ability for users to confirm that data to be signed with non-repudiation keys is correct;
- Erasure of protected information before memory is released to the operating system;
- Maintaining secure operations in the event of power interruption to the smart card.

2.3.1.2 Operating System

The operating system for the PC portion of the TOE is:

- Microsoft Windows 2000 Professional (Service Pack 2) operating system.

The operating system for the smart card portion of the TOE is MULTOS 4.02 or MULTOS 4.06. The MULTOS operating systems by themself do not contain any applications.

2.3.1.3 Software

The software to be evaluated in the TOE is identified in Figure 2 (unshaded area).

- The PKCS#11 Provider delivers the interface between TrustedNet Connect Server and client programs such as Netscape Navigator and Lotus Notes.
- The Microsoft CSP provides cryptographic services to programs that interface to Microsoft CryptoAPI. Programs include: Outlook, Outlook Express, Internet Explorer, Windows 2000 Log on, and various IPsec clients.
- The core component of the TOE is the TrustedNet Connect Server. This offers a COM interface for programs to access the functionality of TrustedNet Connect directly. SecureNet products use this interface. The TrustedNet Connect Server automatically detects smart card insertion and interfaces with the smart card cardlets.
- The cardlets are SecureNet proprietary and designed to simplify third-party application's use of smart card functionality. The cardlets handle all file and data manipulation operations on the smart card. The design of TrustedNet Connect allows users to complete personalisation of their smart cards. Consequently, smart card activity to do with key generation, loading, storage, management, and password loading, use, and change, are within the scope of the TOE. MULTOS provides the means of loading applications and application data, such as initial cryptographic keys, and the MULTOS loading process is excluded from the scope of the TOE.

2.3.2 Physical

The physical scope and boundary is limited to:

- The PC containing the TrustedNet Connect product, and
- The smart card containing the MULTOS operating system and the cardlets.

The operation of the TOE requires a smart card reader that provides the interface between the user's smart card and the host executing the TrustedNet Connect software. The reader is outside the scope of the TOE because the TOE is required to work with any reader and associated device driver from third parties that comply with PC/SC specifications.

The TOE is:

- TrustedNet Connect 2.0.4.9 host software which contains –
  - TrustedNet Connect Server,
  - The PKCS#11 Provider,
  - The Microsoft CSP implementation.
- The Digital ID 2.0.4.9 cardlet on the MULTOS 4.02 or MULTOS 4.06 smart card.
- The Key Management 2.0.4.9 cardlet on the MULTOS 4.02 or MULTOS 4.06 smart card.

Only MULTOS smart cards are considered for this ST.

The TOE requires the following hardware:

- An IBM-compatible PC (minimum Intel Pentium or compatible 166Mhz, 64 megabytes RAM, 10 megabytes hard disk space, a CDROM drive, and an available serial, PCMCIA or USB port for smart card reader)
- A Keycorp MULTOS 4.02 (Release 1N'-AMD) E6 evaluated smart card (AISEP Certification Report Number 2000/13) OR
- A Keycorp MULTOS 4.06 (Release 1Q) smart card on Infineon SLE66CX320P (Pending evaluation)
- A Gemplus, or PC/SC compliant smart card reader such as the Gemplus GemPC 430.

3. TOE SECURITY ENVIRONMENT

The TrustedNet Connect is primarily installed and used on client workstations to enable smart card-based user authentication, digital signatures, key exchange, data encryption and secure communications.

It is anticipated that TrustedNet Connect will be used in a variety of organisations and potentially by a large number of users, to facilitate end user authentication and smart card-based encryption for transactions performed over open networks such as the Internet. In particular, TrustedNet Connect is likely to be important for banking and telecommunication applications.

Typical implementations of applications that utilise TrustedNet Connect include the following smart card based applications:

- User authentication;
- Secure transaction services;
- File encryption;
- Secure communications.

The assets which are protected by the TOE are the user's private keys stored on the smart card, and the operations which the smart card can perform.

3.1 Secure Usage Assumptions

The following assumptions are made in relation to the operation of the TOE:

Name	Description
A.Protect	The TOE is installed in an operating environment that is set up to protect the TOE from modifications and to withstand attacks by sophisticated external agents.
A.Card_Security	The smart card is protected from disclosure of information through electromagnetic emanations and from physical attacks on the card.
A. Key_Distribution	Public keys on smart cards are correctly associated with users' identities and distributed to relying parties; the association is revoked immediately when requested or when loss of the smart card is reported.
A.Trained_Staff	Authorised TOE users and administrators are trusted to follow the guidance provided for secure operation of the TOE.
A.Card_Link	Information which passes through smart card reader drivers and smart card readers to smart cards is not intercepted or modified.
A.Trusted_System	The OS on which the TOE is deployed correctly interfaces to the TOE.
A.Card_OS	A trustworthy card operating system is used that loads card applications (cardlets) securely, protects them from other card applications, and erases all application data when they are deleted.

Table 3-1 - Secure Usage Assumptions

3.2 Threats to Security

Either the TOE or its environment (eg, through personnel, physical or administrative safeguards) may address threats. Threats originating within the TSC are prefaced by 'T.', environmental threats are prefaced by 'TE.'. These two classes of threats are discussed separately.

3.2.1 Threats addressed by the TOE

The TOE addresses the following threats:

Name	Description
T.CryptAnalysis	An attacker may recover protected data through cryptanalysis or by exploiting any key management weakness.
T.Object_Reuse	Protected secret data (secret cryptographic keys and passwords) may be revealed due to being kept in residual memory after the resource has been reallocated.
T.Power_Disruption	Intentional or accidental power disruption may reveal protected information.
T.Repudiation	The creator of a digitally signed message may deny having endorsed it.
T.Unauth_Access	A person who is not authorised to use the TOE is able to obtain unauthorised access to TOE functionality or information protected by the TOE.

Table 3-2 - Threats addressed by the TOE

3.2.2 Threats addresses by the environment

The TOE operating environment addresses the following threats:

Name	Description
TE.Envioronment	Unauthorised users or skilled external agents exploit weaknesses in the operating environment to gain access to or modify the TOE.
TE.Card_Extract	An attacker may successfully physically attack a smart card or analyse electromagnetic emanations to recover protected data stored on a smart card.
TE.Card_Hack	The key material held on the smart card can be compromised during all stages of the card life-cycle resulting in an attacker either gaining unauthorised access to protected information on the card or disclosure of protected information. Attacks can occur at all stages of the smart card life-cycle from the manufacturer introducing malicious code to physical attacks on the card or attacks from other card applications once it has operational keys loaded onto it.
TE.Key_User	Applications using the TOE incorrectly associate a user with a public key resulting in loss of information or authentication failure from man-in-the-middle attacks on the applications.
TE.Driver_Tampering	An attacker may interfere with the operation of third party card readers or drivers to obtain protected information.
TE.User_Error	User or configuration errors may leave the TOE in a state that is not secure, or may otherwise enable unauthorised persons access to the secure functions and/or protected information within the TOE.

TE.Trusted_System	Failure of the OS on which the TOE is deployed to interface to the TOE correctly allows an unauthorised person to obtain protected information or access secure functions.

Table 3-3 - Threats addressed by the Environment

4. SECURITY OBJECTIVES

The security objectives are a high-level statement of the intended response to the security problem. These objectives indicate how the security problem, as characterised in the "Security Environment" section of the ST (Section 3.2.2), is to be addressed.

Objectives addressed by the TOE are prefaced by 'O.', environmental objectives are prefaced by 'OE.'.

Table 4-1 describes security objectives for the TOE, while Table 4-2 describes objectives for the environment.

4.1 Security Objectives for the TOE

Name	Description
O.Access_Control	The TOE provides the means by which access to IT assets can be restricted.
O.Cryptographic	The cryptography implemented in the TOE is sufficient to withstand the cryptanalysis capabilities of likely attackers.
O.Feedback	When the non-repudiation key is used, the TOE provides users with means of checking the message to be signed. Users must then confirm the signing operation. The TOE checks that the generated signature matches the message to give users assurance that they are signing the intended message.
O.I&A	The TOE uniquely identifies all users, and authenticates the claimed identity before granting users access to secure operations on protected IT assets.
O.Key_Management	The TOE fully defines cryptographic components, functions, and interfaces to ensure appropriate protection for cryptographic keys throughout their life cycle, covering generation, distribution, storage, use, and destruction.
O.No_Residual_Info	The TOE ensures that there is no residual information in information containers or system resources used for passwords and secret cryptographic keys upon their re-allocation to different users.
O.Secure_State	The TOE recovers to a secure state without security compromise after a system error or other interruption of system operation as a result of power interruptions.

Table 4-1 – Security Objectives for the TOE

4.2 Security Objectives for the Environment

Name	Description
OE.Protect	The operating environment is set up to protect the TOE from modification and is capable of resisting skilled attacks from external agents.
OE.Card_Security	The smart card is protected from disclosure of information through sufficiently low levels of electromagnetic emanations which correlate to card information and resistance to physical attacks on the card (met by a MULTOS card).
OE.Key_Distribution	The correct associations between users and their public keys are distributed to relying parties and revoked in a timely manner.
OE.Trained_Staff	Those responsible for the configuration and operation of the TOE have read the provided guidance, and consequently shall set up and operate the TOE securely in accordance with the provided guidance.
OE.Secure_Drivers	The third party smart card readers and drivers do not allow protected information to be read or interfered with and card readers and connectors are free of data interception devices.
OE.Trusted_System	The OS on which the TOE is deployed correctly implements its published interfaces.
OE.Crypto_Primitives	The operating system on the smart card correctly implements cryptographic primitives and provides a source of random numbers with sufficient entropy that it will not reduce the effective length of cryptographic keys.
OE.Card_OS	The card operating system has been evaluated to at least an equivalent level as the TOE and protects card applications from disclosure or modification during loading, from interference from other card applications, and from disclosure of data after the application is deleted.

Table 4-2 - Security Objectives for the Environment

5. IT SECURITY REQUIREMENTS

5.1 TOE Security Functional Requirements

The following SFRs are from Common Criteria Part 2.

In the following sections, selections and assignments are underlined. Refinements are in **bold text** within [square brackets]. Iterations have a **/label** added.

5.1.1 Class FCS: Cryptographic Support

5.1.1.1 FCS_CKM.1 Cryptographic key generation

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm RSA, and specified cryptographic key sizes RSA 1024, that meet the following: standards which meet the requirements of DSD as the National COMSEC authority.[FCS_CKM.1.1]

Dependencies:	FCS_COP.1	Cryptographic operations
	FCS_CKM.4	Cryptographic key destruction
	FMT_MSA.2	Secure security attributes

5.1.1.2 FCS_CKM.4 Cryptographic key destruction

The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwrite with zeros keys stored in RAM when no longer required, and smart card-based key erase command that meets the following: standards which meet the requirements of DSD as the National COMSEC authority.[FCS_CKM.4.1]

Dependencies:	[FDP_ITC.1	Import of user data without security attributes
	or FCS_CKM.1	Cryptographic key generation]
	FMT_MSA.2	Secure security attributes

5.1.1.3 FCS_COP.1/RSA Cryptographic operation

The TSF shall perform:

a. decryption of session keys (under an RSA key),

b. digest calculation and digital signing of messages (using SHA-1 with RSA),

c. digital signature generation on a message digest (with an RSA private key),

d. decryption of the private key of a RSA key pair using the DES or 3DES algorithm in CBC mode

in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes of 1024 bits, DES cryptographic key size of 56 bits and 3DES with cryptographic key size of 112 bits that meet the following:

a. PKCS#1v2,

b. PKCS#7v1.5 and ISO 9796-2,

c. PKCS#1v2,

d. AS 2805.5.4.2000, FIBS PUB 81,

all standards which meet the requirements of DSD as the National COMSEC authority. FCS_COP.1.1

Dependencies:	[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

5.1.1.4 FCS_COP.1/Verify Cryptographic operation

The TSF shall perform

a. calculations of password digests (using SHA1),

b. calculation of message digests (using MD5, SHA1),

c. encryption of session keys (under an RSA public key),

d. verification of digital signatures on messages (using SHA-1 or MD5 with RSA),

e. verification of digital signatures on a message digest (with RSA)

in accordance with specified cryptographic algorithms SHA1, MD5, RSA and cryptographic key size RSA: 1024 that meet the following:

a. FIBS PUB 180-1,

b. FIBS PUB 180-1 and RFC 1321,

c. PKCS#1v2,

d. PKCS#1v2, FIBS PUB 180-1 and RFC 1321,

e. PKCS#1v2

All standards which meet the requirements of DSD as the National COMSEC authority. FCS_COP.1.1

Dependencies:	[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

5.1.2 Class FDP: User Data Protection

5.1.2.1 FDP_ACC.2/Keys Complete access control

The TSF shall enforce the Private Key SFP on

a. subject: user

b. object: private key,

and all operations among subjects and objects covered by the SFP. FDP_ACC.2.1

The TSF shall ensure that all operations between any subject in the TSC and any object within the TSC are covered by an access control SFP. $^{\text{FDP\_ACC.2.2}}$

Dependencies: FDP_ACF.1

5.1.2.2 FDP_ACC.2/Card data Complete access control

The TSF shall enforce the Card Data SFP on

a. subject: user,

b. object: certificates, public keys stored on the smart card

and all operations among subjects and objects covered by the SFP. $^{\text{FDP\_ACC.2.1}}$

The TSF shall ensure that all operations between any subject in the TSC and any object within the TSC are covered by an access control SFP. $^{\text{FDP\_ACC.2.2}}$

Dependencies: FDP_ACF.1

5.1.2.3 FDP_ACF.1/Keys Security attribute based access control

The TSF shall enforce the Private Key SFP to objects based on: the private key's password, private key access permissions, and the private key usage attribute.$^{\text{FDP\_ACF.1.1/krys}}$

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

a. Prohibit the operation if the private key access permission that applies to the operation is set to "never", or it is set to password and the user is not authenticated with respect to the password.

b. Further to the above, prohibit cryptographic operations with private keys if the key is encrypted, the user is not authenticated with respect to the password, or the operation is not allowed by the key usage attribute.

c. The only allowed cryptographic operation with a non-repudiation key is to create signatures on messages that the user has authenticated.

d. Private keys can only be stored in smart cards.$^{\text{FDP\_ACF.1.2}}$

The TSF shall explicitly authorise access of subjects to objects based on the following additional rule: a private key that is encrypted can be decrypted if the correct decryption key is presented.$^{\text{FDP\_ACF.1.3/keys}}$

The TSF shall explicitly deny access of subjects to objects based on the rule that a stored private key cannot be read off the smart card.$^{\text{FDP\_ACF.1.4/keys}}$

Dependencies:	FDP_ACC.1	Subset access control
	FMT_MSA.3	Static attribute initialisation

5.1.2.4 FDP_ACF.1/Card data Security attribute based access control

The TSF shall enforce the Card Data SFP to objects based on: the password associated with the object, and the data access permissions, $^{\text{FDP\_ACF.1.1}}$

The TSF shall enforce the following rule to determine if an operation among controlled subjects and controlled objects is allowed:

a. Prohibit access to the object unless the condition specified by the data access permission that applies for the operation is satisfied. $^{\text{FDP\_ACF.1.2}}$

The TSF shall explicitly authorise access of subjects to objects based on the following additional rule: none.FDP_ACF.1.3

The TSF shall explicitly deny access of subjects to objects based on the rule: none.FDP_ACF.1.4

Dependencies:	FDP_ACC.1	Subset access control
	FMT_MSA.3	Static attribute initialisation

5.1.2.5 FDP_ITC.1/Keys Import of user data without security attributes

The TSF shall enforce the Private Key SFP when importing user data, controlled under the SFP, from outside of the TSC. FDP_ITC.1.1

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

The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: the private key can be loaded encrypted and later decrypted: ie. the key can be loaded in two separate components. FDP_ITC1.3

Dependencies:	[	FDP_ACC.1	Subset access control,
	or	FDP_IFC.1	Subset information flow control]
	FMT_MSA.3	Static attribute initialisation	

5.1.2.6 FDP_ITC.1/Card data Import of user data without security attributes

The TSF shall enforce the Card Data SFP when importing user data, controlled under the SFP, from outside of the TSC. FDP_ITC.1.1

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

The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: a certificate can only be imported if the certified public key matches a public key of a key pair stored on the card. FDP_ITC1.3

Dependencies:	[	FDP_ACC.1	Subset access control,
	or	FDP_IFC.1	Subset information flow control]
	FMT_MSA.3	Static attribute initialisation	

5.1.2.7 FDP_DAU.2 Data authentication with identity of guarantor

The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of messages that are to be signed with a non-repudiation key. FDP_DAU 2.1

The TSF shall provide users with the ability to verify evidence of the validity of the indicated information and the identity of the user that generated the evidence. FDP_DAU.2.2

Dependencies: FIA_UID.1 Timing of identification

5.1.2.8 FDP_RIP.1 Subset residual information protection

The TSF shall ensure that any previous information content of a resource is made unavailable upon the de-allocation of the resource from the following objects:

a. passwords

b. unblock passwords

c. data decrypted with a private key

d. private keys

e. keys for decrypting private keys

f. random numbers generated by the TOE. FDP_RIP.1.1

Dependencies: No Dependencies

5.1.3 Class FIA: Identification and Authentication

5.1.3.1 FIA_AFL.1 Authentication failure handling

The TSF shall detect when a pre-set maximum number of unsuccessful authentication attempts occur related to password entry. FIA_AFL.1.1

When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall disallow further authentication attempts with the password and blocks access to operations on objects that the password protects. FIA_AFL.1.2

Dependencies: FIA_UAU.1 Timing of authentication

5.1.3.2 FIA_UAU.1 Timing of authentication

The TSF shall allow:

a. generation of random numbers,

b. reading the types of objects stored on the smart card,

c. selection of objects,

d. reading card password policy constraints if present,

e. reading of password policies,

f. reading the number of password retries, authentications and unblocks left,

g. reading the number of authentications left for an unblock password,

h. reading the status of a key pair,

i. reading the key pair usage attribute,

j. decryption of a stored encrypted private key,

k. setting an initial password value,

l. password entry,

m. checking passwords against verification values on smart cards,

n. password re-entry if a password is incorrect and number retries left for a password is greater than zero,

o. operations allowed under the Card Data SFP or Private Key SFP without the user being authenticated against a password on behalf of the user to be performed before the user is authenticated.FIA_UAU.1.1

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

Dependencies: FIA_UID.1 Timing of identification

5.1.3.3 FIA_UAU.4 *Single-use authentication mechanisms*

The TSF shall prevent reuse of authentication data related to password unblocking.FIA_UAU.4.1.

Dependencies: No dependencies

5.1.3.4 FIA_UAU.6 *Re-authenticating*

The TSF shall re-authenticate the user under the conditions:

a. after removal and reinsertion of smart card, for operations requiring authentication,

b. after a reset or loss of power to the smart card, for operations requiring authentication,

c. change password,

d. unblock a password,

e. generation of a signature when a non-repudiation key is used,

f. an operation that is conditional under the Private Key SFP or Card Data SFP on authentication against a password that requires re-authentication for every operation.FIA_UAU.6.1

Dependencies: No dependencies

5.1.3.5 FIA_UID.2 *User identification before any action*

The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user. FIA_UID.2.1

Dependencies: No dependencies

5.1.4 Class FMT: Security Management

5.1.4.1 FMT_MSA.1 *Management of security attributes*

The TSF shall enforce the Private Key SFP and Card Data SFP to restrict the ability to perform operations in Table 5-1 on the security attributes listed in Table 5-1 to the roles indicated in Table 5-1.FMT_MSA.1.1

Security Attribute	Operations	Role
Password	Create	User
Password Digest (verification data)	Set initial value to correspond to a password	User
	Change to correspond to a new password (i.e. change password – resets authentications left)	User Issuer

Password Policy	On password creation, set whether an authentication is only valid for one operation, (i.e. whether password re-authentication is required for every controlled operation.)	User
	On password creation, set the maximum number of authentication retries.	User
	On password creation, set the maximum number of authentications before the password must be changed.	User
	On password creation, set the minimum length of the password.	User
	On password creation, set whether passwords can apply to multiple key pairs.	User
	Read password policy.	User
Password Unblocks left	On password creation, set the initial value	User
	Read value	User
Password retries left	Reset the number of authentication retries to the maximum (unblock password).	Issuer
	Read value.	User
Password authentications left before password must be changed	Read value. (Value is reset to the maximum after a password change.)	User
Card password policy constraints	Read constraints.	User
Unblock Password authentications left	Read number of authentications left.	User
Key password	On creation of a key, set password that applies	User
Key Status	Read Value.	User
Key Usage attribute	Set value on key object creation.	User
	Read value.	User

Table 5-1 Management of user security attributes

Dependencies:	[	FDP_ACC.1	Subset access control
	or	FDP_IFC.1	Subset information flow control]
	FMT_SMR.1	Security roles	

5.1.4.2 FMT_MSA.2 Secure security attributes

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

Dependencies:	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

5.1.4.3 FMT_MSA.3 Static attribute initialisation

The TSF shall enforce the Private Key SFP and Card Data SFP to provide restrictive default values for security attributes that are used to enforce the SFP.[FMT_MSA.3.1]

The TSF shall allow the issuer or user to specify alternative initial values to override the default values when an object or information is created.[FMT_MSA.3.2]

Dependencies:	FMT_MSA.1	Management of security attributes
	FMT_SMR.1	Security roles

5.1.4.4 FMT_SMR.1 Security roles

The TSF shall maintain the roles user, issuers.[FMT_SMR.1.1]
The TSF shall be able to associate users with roles.[FMT_SMR.1.2]

Dependencies:	FIA_UID.1	Timing of identification

5.1.5 Class FPT: Protection of the TSF

5.1.5.1 FPT_FLS.1 Failure with preservation of secure state

The TSF shall preserve a secure state when the following types of failures occur: smart card removed, reset or powered down during operation, user Authentication Failure.[FPT_FLS.1.1]

Dependencies:	ADV_SPM.1	Informal TOE security policy model

5.2 IT Security Requirements for the Environment

5.2.1 MULTOS Smart Card

The E6-evaluated MULTOS 4.02 (Release 1N'-AMD) or the MULTOS 4.06 (Release 1Q) (pending evaluation) smart card provides the following security functions for the TOE:

5.2.1.1 FCS_COP.1/Primitives Cryptographic operation

The TSF shall perform:

a. DES cryptographic primitive operation,

b. SHA1 cryptographic primitive operation,

c. modulo exponentiation,

in accordance with specified cryptographic algorithms: (a) DES, (b) SHA1, (c) RSA, and cryptographic key sizes: (a) DES: 64 bits, (b) SHA1: none, (c) RSA: 1024bits that meet the following:

- FIBS PUB 46-3
- FIBS PUB 180-1
- PKCS#1v2
- standards which meet the requirements of DSD as the National COMSEC authority. FCS_COP.1.1

Dependencies:	[	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

5.2.1.2 FCS_COP.1/Random Cryptographic operation

The TSF shall perform

a. random number generation,

in accordance with a specified cryptographic algorithm uniformly distributed random numbers and cryptographic key sizes none that meet the following:

- standards which meet the requirements of DSD as the National COMSEC authority. FCS_COP.1.1

Dependencies:	[	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

5.3 TOE Security Assurance Requirements

This TOE is assured to EAL4 as described in Common Criteria Part 3.

6. TOE SUMMARY SPECIFICATIONS

This section presents the Security Functions implemented by the TOE and the Assurance Measures applied to ensure their correct implementation.

6.1 IT Security Measures

This section presents the security functions performed by the TOE and provides a mapping between the identified security functions and the Security Functional Requirements that it must satisfy.

6.1.1 TSF1 – Cryptographic Services

This TSF provides cryptographic services. Cryptographic services are implemented both on the card and in the host software.

FCS_COP.1/RSA

The implementation of cryptographic services on the card is based on functions provided by the smart card operating system. The on-card application uses these function to implement cryptographic protocols for:

- decryption of session keys in PKCS#1v2 encoded blocks with 1024 bit RSA keys;
- generating signatures on a message digest with padding specified in PKCS#1v2, the RSA mechanism and 1024 bit keys;
- when a non-repudiation key is used, an audit signature can be additionally generated based on the message digest in the PKCS#1 signature and a count of signatures generated with the key, using the ISO 9697-2 standard;
- decrypting encrypted private keys that have been loaded in split form using the two key or single key triple DES algorithm in CBC mode (AS 2805.5.4.2000, FIBS PUB 81).

Access is also provided to the full results of an on-card RSA multi-precision exponentiation operation provided through the smart card operating system.

FCS_COP.1/Verify
FCS_COP.1/RSA

Cryptographic services are also implemented in the host software. Implemented functions are:

- SHA1 cryptographic hash function: to calculate digests of passwords for setting the password verification data on the card and to calculate message digests according to FIBS PUB 180-1;
- Generating SHA-1 digests of PKCS#7v1.5 messages and authenticated attributes, so that the data can be signed by the on-card application to form PKCS#7 signed blob;
- MD5: to calculate messages digests according to RFC 1321;
- RSA public key encryption: to *wrap* user supplied session keys according to PKCS#1v2;
- verification of PKCS#1v2 encoded digital signatures on a messages using RSA public keys; and

- verification of PKCS#1v2 encoded digital signatures against supplied message digests using RSA public keys.

All secret cryptographic keys (private keys, key decryption keys and session keys) and the results of private key decryption operations in volatile memory (RAM) are overwritten with zeros as soon as cryptographic operations are completed. *FCS_CKM.4* *FDP_RIP.1*

6.1.2 TSF2 – Identity and Authentication

TrustedNet Connect, through its cryptographic operations on the smart card, is capable of authenticating the user to applications that utilise TrustedNet Connect. In turn, TrustedNet Connect relies on passwords to authenticate users. The advantage that TrustedNet Connect provides is that it stores password information on smart cards that can control all access to their stored data. Consequently, while users are able to check passwords, users cannot read any data used to verify passwords.

TrustedNet Connect allows users to enter a separate password for each key pair stored on a card. Alternatively, a single password can be used for multiple keys if the password policy allows it to be *synchronised*. The password for a key pair also applies to certificates related to that key pair. TrustedNet Connect maintains the authentication status of users against each password, and this is used as the basis for controlling access to protected resources.

All protected data is contained on a smart card. The user must insert the smart card into the reader for the rest of the TOE to have any access to a user's protected information. A user is identified by inserting a properly configured card into the smart card reader. No mediated operation is available unless the card is inserted. *FIA_UID.2*

Authentication of the user consists of the user presenting a password that matches the password verification information stored in the smart card. The stored password verification information is derived from a SHA1 digest of the password. Passwords have an attribute that specifies the maximum number of retries, and have a current count of retries remaining for each password. Each attempt at authentication decrements the count of remaining retries for the password. A correct password resets the count to the maximum. The maximum retries is set during password creation and cannot be changed. *FIA_UAU.1*

When client applications request services that require the authentication of users, the client application must either pass in the password, or TrustedNet Connect can display a dialog box requesting entry of the specified password. Users must authenticate themselves before the request is successfully completed. This dialog box feature allows all password entry to be handled by TrustedNet Connect.

If an incorrect password is presented, operations that require authentication against that password are prohibited. *FPT_FLS.1*

If a user fails to present the correct password within the maximum allowed number of retries for the password, the password becomes blocked. Users cannot authenticate themselves against blocked passwords. Also, all access is denied to any operation that requires authentication against a blocked password. *FIA_AFL.1*

Special single-use unblocking passwords exist. Once a user is successfully authenticated against an unblocking password using stored verification data, *FIA_UAU.4* new password verification data applies for the next authentication. Furthermore, each password has unique verification data. Even though the TOE can verify a password, it cannot generate the password.

Every identified user is given the role of "User" while only those that are authenticated against an unblocking password are given a role of "Issuer". *FMT_SMR.1*

Unblock passwords also have a maximum retries attribute, and limitations on consecutive incorrect presentations of passwords also apply to unblock passwords. When the number of allowed retries is exhausted, the unblock password is itself permanently blocked. *FIA_AFL.1*

Passwords have a *scope* attribute that determines whether an authentication is only valid for a single operation. If set, users must re-authenticate themselves for every operation that requires authentication against that password. *FIA_UAU.6*

Client applications that interface to TrustedNet Connect to access services can store passwords and bypass requirements for re-authentication when an operation is conditional on authentication against a password with the scope of authentication set to a single operation. To prevent this, TrustedNet Connect only accepts entry of passwords with single operation scope from dialog boxes that it creates.

On smart card power-ups or resets, all data in volatile memory (RAM) is erased. Erased RAM is interpreted as 'no authentication having occurred'. Since all memory of authentications is erased, users must re-authenticate themselves after a smart card is removed and re-inserted, reset, or powered down and up.

The number of authentication retry attempts remaining for passwords is stored in non-volatile memory. On card insertion, the number of retries remaining is not lost. ***FPT_FLS.1***

6.1.3 TSF3 – Data Protection

TrustedNet Connect controls access to all user data and security attributes. This extends to protection against power failures to smart cards and assuring that data is not passed to users through re-allocation of memory.

All protected user data and security attributes are stored on smart cards which control all access to contained data. The smart card operating system does not provide any means of directly accessing or processing data: all access is mediated through the application code executing on the card. *FDP_ACC.2.2/Keys* *FDP_ACC.2.2/Card data*

In particular, the smart card application that is part of the TOE implements specific rules that apply to all operations between users and their stored private keys, or between users, and public keys and certificates stored on the card. *FDP_ACC.2.1/Keys* *FDP_ACC.2.1/Card data*

Access to the user's private keys is prohibited unless the user has been identified and, for cryptographic operations, authenticated. Private key access permissions are the set of all permissions that control operations that affect private keys. For any of these operations, the access permission associated with that specific operation controls whether the operation is prohibited, allowed, or requires authentication. *FDP_ACF.1/Keys* *FIA_UAU.1.1*

There are overriding rules that apply: a user can always decrypt an encrypted private key if the user presents the correct decryption key (redundancy in the private key is used to verify that the correct decryption key has been presented), and TrustedNet Connect does not allow private keys stored on the card to be read.

Public keys and certificates have a simpler access control policy. Operations are controlled through access permissions which may specify one of the following: *FDP_ACF.1/Card data* *FIA_UAU.1.1*

- Always grant access,
- Grant access if the user is authenticated with the password associated with the object,
- Never grant access.

Access to the random number generator does not require authentication. *FIA_UAU.1.1*

Any data decrypted with a private key has to be treated as confidential. Also, TrustedNet connect allows client applications to access the random number generator on the smart card. Since client applications typically use random number generators to form session keys, generated random numbers are considered confidential. *FDP_RIP.1*

When the smart card is removed, and when TrustedNet Connect is shutting down, TrustedNet Connect ensures all passwords, unblock passwords, data that has been decrypted, generated random numbers, any triple DES keys used for decrypting private keys, and private key data that is stored in RAM is overwritten with zeros. The RAM can then be safely returned to the operating system and reallocated to other users.

Changes to the state of the smart card are managed so that the card is secure at all times. Power disruptions naturally lead to a loss of all data stored in RAM. Changes to non-volatile memory are arranged so that if the operation of the smart card is disrupted, the data protection that applies to objects is not reduced. *FPT_FLS.1*

6.1.4 TSF4 – Key Management

The Key Management Security Function provides the ability to set up, generate, import and review the configuration of stored cryptographic keys within TrustedNet Connect.

Private keys have a *key usage attribute* that controls which cryptographic operations are allowed as follows: *FDP_ACF.1/Keys*

Encryption key:	Decrypt a PKCS#1 encrypted session key.
Signing key:	Create a signature.
Utility Key:	Create a signature, and Decrypt a PKCS#1 encrypted session key.
Unrestricted Key:	Return entire decrypted block of data.
Non-repudiation key:	Only create signatures on messages that the user has authenticated with TSF6 – Non-Repudiation.

Cryptographic operations are prohibited on encrypted private keys.

Access rights can only be defined when objects are created. Since objects can be created on smart cards during card application loading, issuers of *FMT_MSA.3.2* cards can specify the access permissions that apply and the initial values of other security attributes. These specifications are translated to static access permissions that form part of the TOE data. This process is part of TOE generation.

When creating a private key object, users can set the key usage attribute and select the password that applies to the key. *FMT_MSA.3.2* *FMT_MSA.1*

If initial values are not specified, restrictive defaults are applied. These ensure that: *FMT_MSA.3.1*

- A password protects access;
- Key pairs and certificates can only be deleted and private keys erased if the user is authenticated;
- Key pairs and certificates cannot be modified;
- The default key usage attribute is *utility key*.

Values for security attributes are checked to ensure that they are secure. In particular TrustedNet Connect will not recognise nor use private keys unless they are protected with secure access permissions. *FMT_MSA.2*

Users can select objects and read information about those objects even before they are authenticated so that they can check the configuration of the card. The allowed operations are: *FIA_UAU.1.1* *FMT_MSA.1*

- read the types of object stored on the smart card and select them,
- reading the status of a key pair,
- reading the key pair usage attribute.

The card is capable of generating random 1024 bit RSA keys based on the random number generator function available through the smart card operating system. The keys are available for use by other end user applications. *FCS_CKM.1*

Generated private keys that are stored on the card are put in the card's non-volatile memory without ever being taken off the card.

The card has an erase function that overwrites stored private keys with zeros. *FCS_CKM.4*

TrustedNet Connect allows key pairs and certificates to be imported and stored in appropriate objects. Importation of certificates and key pairs is dependent on access permissions that apply to the target object. *FDP_ITC.1.1/Keys, FDP_ITC.1.1/Card data*

Security attributes associated with data that is being imported is ignored, and the imported data assumes the security attributes of the target object. *FDP_ITC.1.2/Keys* *FDP_ITC.1.2/Card data*

Encrypted private keys can be imported and decrypted later with the correct triple DES key. *FDP_ITC.1.3/Keys*

Imported private keys can only be stored in private key objects on the smart card. *FDP_ACF.1/Keys*

Certificates will only be imported if TrustedNet Connect can locate a stored key pair with the same public key as that in the certificate. *FDP_ITC.1.3/Card data*

6.1.5 TSF5 – Password Management

The Password Management Security Function provides the ability to set up and review the configuration of passwords within TrustedNet Connect.

When creating passwords, users must specify the initial password and can set initial values for: *FMT_MSA.3.2* *FMT_MSA.1*

- The password scope attribute,
- Whether multiple key pairs can be protected with the password,
- Maximum number of authentication retries,
- Maximum number of authentications before a password must be changed,
- Minimum password length,
- The number of times the password can be unblocked.

Values for security attributes are checked to ensure that they are secure. In particular, passwords must be at least a certain number of characters long, must not be composed of identical characters or characters that are a straight ascending or descending sequence. *FMT_MSA.2*

TrustedNet also supports card-specific constraints for passwords. If present, these constrain possible values for security attributes by setting:

- The absolute maximum number of authentication retries,
- The absolute minimum length of passwords,
- The absolute maximum number of authentication with a password before the password has to be changed,
- The absolute maximum number of times a password can be unblocked.

If initial values are not specified, restrictive defaults are applied. These ensure that: *FMT_MSA.3.1*

- The maximum number of authentication retries for a passwords is set to the value in the card constraints, or 5 if no card constraints are set.
- The passwords must be at least as long as specified card constraints, or a minimum of 4 characters long if card constraints are not set.

Users can change their passwords. *FMT_MSA.1*

A password must be changed after the number of authentications against that password exceeds its maximum. Changing the password also resets this count.

Only the role of Issuer can unblock a password. This is done by resetting the current *retries left* counter to the maximum for the password. An unblocking operation also sets a new value for the password.

Users need to re-authenticate themselves for every password change and password unblock. *FIA_UAU.6*

Users are allowed to set the initial value of passwords before they are authenticated. This allows users to take control of the objects protected with the smart card. A user should set the initial password value before any keys are loaded or generated on the card so those keys are always under the control of the user. *FIA_UAU.1.1* *FMT_MSA.1*

Users can select objects and read information about those objects even before they are authenticated so that the configuration of the card can be always checked on behalf of the user. The allowed operations are:

- reading the types of object stored on the smart card and selecting them

- reading card password policy constraints if present,
- reading of password policies,
- reading the number of password retries, authentications and unblocks left,
- reading the number of authentications left for an unblock password.

6.1.6 TSF6 – Non-Repudiation

TrustedNet Connect gives special treatment to signature generation with *non-repudiation* keys (keys with their key usage attribute set to *non-repudiation*).

For any message that is to be signed with a non-repudiation key, the user is given the option of viewing the message or saving the message in a file for later viewing with an external program (i.e. the display is evidence that the message is a valid message to be signed). *FDP_DAU.2.1*

Users can view the message to confirm that the message is one that they created and intend to sign. *FDP_DAU.2.2*

After users are given the opportunity to verify the messages to be signed, users must authenticate themselves by presenting the key's password before the signature is generated. *FIA_UAU.6*

Following authentication, the PKCS#1v2 signature can be calculated on the message. Immediately following signature generation, the user can request an audit signature on the message.

6.2 Assurance Measures

This section presents the assurance measures that are used to meet Security Assurance Requirements and presents a mapping between the TOE assurance measures and Security Assurance Requirements. The evidence for assurance comprises of a number of documents that are produced as part of the TOE development. These documents are discussed below.

Document	Title	Date	Version
Software Process Overview	Software Development Process Overview	8/05/2002	1.3
Software Process Guidelines	Software Development Process Guidelines	8/05/2002	1.3
Programming Guidelines	Software Development Process C++ Programming Guidelines	17/06/2002	1.3
	Software Development Process MEL Programming GuideLines	12/12/2001	1.1
Security and Access Control (Product Development)	Procedure 407 Security and Access Control (Product Development)	16/08/2002	Issue 3
Requirements Specification	TrustedNet Connect 2.0 Requirements Specification	11/09/2002	2.0 Revision 17
Software Architecture	TrustedNet Connect 2.0 Software Architecture	29/01/2003	2.0 Revision 14

	TrustedNet Connect 2.0 Card Applications	29/5/2002	2.0 Revision 3
Configuration Management Plan	TrustedNet Connect 2.0 Configuration Management Plan	2/10/2002	2.0 Revision 7
Design Specification	TrustedNet Connect 2.0 Design Specification	9/9/2002	2.0 Revision 6
	TrustedNet Connect 2.0 Card Applications Design Specification	29/8/2002	2.0 Revision 8
Product Life-Cycle	TrustedNet Connect 2.0 Product Life Cycle	30/10/2002	2.0 Revision 4
Representation Correspondence	TrustedNet Connect 2.0 Representation Correspondance	21/11/2002	2.0 Revision 9
Security Policy Model	TrustedNet Connect Security Policy Model	21/03/2002	1.0
Guidance Analysis	TrustedNet Connect 2.0 Security Analysis	10/07/2002	2.0 Revision 1
Strength Analysis	TrustedNet Connect 2.0 Security Analysis	10/07/2002	2.0 Revision 1
Vulnerability Assessment	TrustedNet Connect 2.0 Security Analysis	10/07/2002	2.0 Revision 1
Software Release		29/10/2002	2.0.4 Build 9
Online Help		29/10/2002	2.0.4 Build 9
Release Notes		29/01/2003	2.0.4 Build 9
Test Script		29/10/2002	2.0.4 Build 9
Test Plan	TrustedNet Connect 2.0 Test Plan	1/11/2002	2.0 Revision 11
Test Case	TrustedNet Connect 2.0 Test Plan	1/11/2002	2.0 Revision 11
Test Report	Connect 2.0.4.3 Test Report	31/09/2002	2.0.4 Revision 2
	Connect 2.0.4.9 Test Report	30/10/2002	2.0.4 Revision 2
User Manuals	TrustedNet Connect User Guide		2.0.4 Revision 1
	TrustedNet Connect PKCS #11 Cryptographic Interface Reference		2.0.4 Revision 0
	TrustedNet Connect Cryptographic Service Provider API		2.0.4 Revision 0
	TrustedNet Connect Installation Guide		2.0.4 Revision 0

	TrustedNet Connect Card Personaliser		2.0.4 Revision 2

6.2.1 Software Process Overview

The purpose of this document is to describe the components, phases, roles, activities and artefacts of the SecureNet Software Development Process. It provides a high-level definition of the underlying software development methodology and some basic steps that are to be followed by the individuals playing their roles in the process. This document is an essential reading for all members of the development team. As the Process Overview contains the methodology for developers to use to implement a controlled software life-cycle, it is considered suitable to meet requirements ALC_LCD.1.

6.2.2 Software Process Guidelines

The purpose of this document is to provide some rules and guidelines for the implementation of the SecureNet Software Development Process. As the Process Guidelines contain the methodology for developers to use to follow the Process Overview to implement a controlled software life-cycle, it is considered suitable to meet requirements ALC_LCD.1.

6.2.3 Security and Access Control (Product Development)

The purpose of this document is to specify procedures that control access to the product development site and IT systems. As the procedures cover security applied to the development environment, it is considered suitable to meet requirements ALC_DVS.1.

6.2.4 Requirements Specification

The Requirements Specification document describes the system's intended functionality and its environment, and serves as a contract between the customer and the developers. The Requirements Specification document is developed by an analyst and is used by all members of the development team as an essential input to activities in analysis, design, and test. As the Requirements Specification details the behaviour of the TOE and its interfaces to the environment, it is considered suitable to meet the requirement ADV_FSP.2.

6.2.5 Software Architecture Document

The Software Architecture Document provides a comprehensive architectural overview of the system, using a number of different architectural views to describe different aspects of the system. The Software Architecture Document is developed and maintained by the architect and is used by all members of the development team. As the Software Architecture Document details the TOE in terms of structural architecture (i.e. subsystems), it is considered suitable to meet the requirement ADV_HLD.2.

6.2.6 Configuration Management Plan

The Configuration Management Plan describes all configuration management related activities to be performed during the course of the product life-cycle. It is developed by the configuration manager and used by project managers, designers, developers, system integrators and testers. As the Configuration Management Plan contains sufficient information to describe all configuration management activities and processes, including all configuration management tools and systems, and a unique reference for the TOE, it is considered sufficient to meet the requirements ACM_AUT.1, ACM_CAP.4 and ACM_SCP.2.

6.2.7 Design Specification

The Design Specification document describes low-level design including class definitions and object models and serves as an abstraction of the implementation model and its source code. The Design Specification document is developed by the designer and is used by developers as an essential input to activities in implementation and test. As the Design Specification contains sufficient information for the code developers to implement the software design without further consultation, it is considered suitable to meet requirement ADV_LLD.1.

6.2.8 Software Release

A Software Release represents a product baseline, integrated and tested, that is formally versioned and released either internally (at the and of an iteration) or externally to the customers. Software Release is produced by the system integrator and then tested by the testers. Software release documentation includes details on development tools and all options selected to generate the release hence it addresses requirements for ALC_TAT.1. As each release is uniquely versioned and labelled as such it satisfies ACM_CAP.4.

6.2.9 Product Life-Cycle

The Product Life-Cycle contains information about the processes and procedures in place to generate, package and delivery TrustedNet Connect to customers. This document addresses requirements for ADO_DEL.2.1, and addresses requirements for the generation of the TOE in ADO_IGS.1.

6.2.10 Test Plan

The Test Plan contains information about the purpose and goals of testing within the project. Additionally, the test plan identifies the strategies to be used to implement and execute testing and describes resources needed. The Test Plan is produced by the test designer and used by testers, system integrators and other stakeholders to verify that appropriate test strategies are implemented. As the test plan provides the details of the purpose and goals of testing, it addresses the requirement that the TOE satisfy its security functional requirements ATE_FUN.1.

6.2.11 Test Case

A Test Case is a set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. It includes a set of detailed instructions for the set-up, execution, and evaluation of test results. Test Cases are defined by test designer and used by testers when executing tests. As the Test Case provides the details of the purpose and goals of testing, it addresses the requirement that the TOE satisfy its security functional requirements ATE_FUN.1.

6.2.12 Test Script

Test Scripts are the computer readable instructions that automate the execution of a test procedure (or portion of a test procedure). Test scripts may be created (recorded) using a test automation tool, programmed using a programming language, or a combination of recording and programming. Test Scripts are created by test designer and implemented by the testers. As the Test Scripts provides the details of the purpose and goals of testing, it addresses the requirement that the TOE satisfy its security functional requirements ATE_FUN.1.

6.2.13 Test Report

A Test Report contains results of the test evaluation including results of the test, description of the test coverage and code coverage, and suggested actions to improve the product and testing process. Test Report is produced by test analysts. As the Test Scripts provides the details of the analysis of the depth of testing and coverage, it addresses the requirements ATE_FUN.1, ATE_COV.2, and ATE_DPT.1.

6.2.14 User Manuals

The User Manuals are printed or electronic documents that support the end-user in learning, installing and using the product. User Manuals are developed by a technical writer. As the User Manuals provide the administrator with the functions of the TOE, it satisfies the requirements AGD_ADM.1 and AGD_USR.1. The user manuals also address installation and start-up aspects of ADO_IGS.1.

6.2.15 Online Help

The Online Help is an interactive, context-sensitive electronic document with hypertext and search capabilities that supports the end-user in learning and using the product. Online help is produced by a technical writer. As the Online Help provides support to the User Manual, in the provision to the administrator of the functions of the TOE, it satisfies the requirements AGD_ADM.1 and AGD_USR.1.

6.2.16 Release Notes

The Release Notes describe the particulars of a certain product release and are usually included as part of the product release. Release Notes are produced by a technical writer. As the Release Notes provide support to the User Manual, in the provision to the administrator of the functions of the TOE, it satisfies the requirements AGD_ADM.1 and AGD_USR.1.

6.2.17 Programming Guidelines

The Programming Guidelines are specific to the development of the TOE using these programming tools. As the Programming Guidelines help prevent the ill-defined, inconsistent or incorrect development tools from being used to develop the TOE, they satisfy the requirement ALC_TAT.1.

6.2.18 Evaluation Specific Documents

The Representation Correspondence document will be developed to meet requirement ADV_RCR.1.

The Security Policy Model document will be developed to meet requirement ADV_SPM.1.

The Guidance Analysis document will be developed to meet requirement AVA_MSU.2.

The Strength Analysis document will be developed to meet requirement AVA_SOF.2.

The Vulnerability Assessment document will be developed to meet requirement AVA_VLA.2.

The source code will be provided to meet requirement ADV_IMP.1. The TOE will be provided to meet the requirement for independent testing ATE_IND.2.

6.2.19 Suitability of Assurance Measures

Table 6-1 demonstrates that the identified assurance measures are appropriate to meet the assurance requirements by mapping the identified assurance measures onto the assurance requirements.

The specification of assurance measures is done by reference to the appropriate document (e.g. Configuration Management Plan, User Guide, Installation Guide, etc). Obviously, analysis of the relevant documentation is required to show that the referenced document (assurance measure) meets the requirements of the associated assurance requirement.

Assurance Class	CC Assurance Component	Assurance Measure
Configuration Management	ACM_AUT.1.1D The developer shall use a CM system. ACM_AUT.1.2D The developer shall provide a CM plan.	Configuration Management Plan
	ACM_CAP.4.1D The developer shall provide a reference for the TOE. ACM_CAP.4.2D The developer shall use a CM system. ACM_CAP.4.3D The developer shall provide CM documentation.	Configuration Management Plan, Software Release
	ACM_SCP.2.1D The developer shall provide CM documentation.	Configuration Management Plan
Delivery and Operation	ADO_DEL.2.1D The developer shall document procedures for delivery of the TOE or parts of it to the user. ADO_DEL.2.2D The developer shall use the delivery procedures.	Product Life-Cycle
	ADO_IGS.1 The developer shall document procedures necessary for the secure installation, generation and start-up of the TOE.	User Manuals and Product Life-Cycle

Development	ADV_FSP.2.1D The developer shall provide a functional specification.	Requirements Specification
	ADV_HLD.2.1D The developer shall provide the high-level design of the TSF.	Software Architecture Document
	ADV_IMP.1.1D The developer shall provide the implementation representation for a selected subset of the TSF.	Source code
	ADV_LLD.1.1D The developer shall provide the low-level design of the TSF.	Design Specification
	ADV_RCR.1.1D The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.	Representation Correspondence
	ADV_SPM.1.1.D The developer shall provide a TSP model.	Security Policy Model
Guidance Documents	AGD_ADM.1.1D The developer shall provide administrator guidance addressed to system administrative personnel.	User Manuals, Online Help, Release Notes
	AGD_USR.1.1D The developer shall provide user guidance.	User Manuals, Online Help, Release Notes
Life Cycle Support	ALC_DVS.1.1D The developer shall produce development security documentation.	Security and Access Control (Product Development)
	ALC_LCD.1.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE. ALC_LCD.1.2D The developer shall provide life-cycle definition documentation.	Software Process Overview, Software Process Guidelines
	ALC_TAT.1.1D The developer shall identify the development tools being used for the TOE. ALC_TAT.1.2D The developer shall document the selected implementation-dependent options of the development tools.	Programming Guidelines, Software Release

Tests	ATE_COV.2.1D The developer shall provide an analysis of the test coverage.	Test Report
	ATE_DPT.1.1D The developer shall provide the analysis of the depth of testing.	Test Report
	ATE_FUN.1.1D The developer shall test the TSF and document the results. ATE_FUN.1.2D The developer shall provide test documentation.	Test Plan/ Test Case/Test Script/Test Report
	ATE_IND.2.1D The developer shall provide the TOE for testing.	Software Release (the TOE)
Vulnerability Assessment	AVA_MSU.2.1D The developer shall provide guidance documentation. AVA_MSU.2.2D The developer shall document an analysis of the guidance documentation.	Guidance Analysis
	AVA_SOF.1.1D The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.	Strength Analysis
	AVA_VLA.2.1D The developer shall perform and document an analysis of the TOE deliverables searching for ways in which a user can violate the TSP. AVA_VLA.2.2D The developer shall document the disposition of identified vulnerabilities.	Vulnerability Assessment

Table 6-1 – Assurance Components

7. TOE RATIONALE

7.1 Security Objectives Rationale

The first section shows that each IT security objective and each non-IT security objective counters at least one assumption, policy, or threat. The mappings are straightforward and do not require further explanatory text. The second section shows that security objectives are suitable to counter the identified threats to security and to cover all secure usage assumptions.

7.1.1 Assumptions, Policies and Threats Addressed

Objectives **Threats**	O.Access_Control	O.Cryptographic	O.Feedback	O.I&A	O.Key_Management	O.No_Residual_Info	O.Secure_State	OE.Protect	OE.Key_Distribution	OE.Card_Security	OE.Trained_Staff	OE.Secure_Drivers	OE.Trusted_System	OE.Crypto_Primitives	OE.Card_OS
T.CryptAnalysis		✓			✓									✓	
T.Object_Reuse						✓									
T.Power_Disruption							✓								
T.Repudiation			✓		✓										
T.Unauth_Access	✓			✓	✓										
TE.Envioronment								✓							
TE.Card_Extract										✓					
TE.Card_Hack										✓					✓
TE.Key_User				✓					✓						
TE.Driver_Tampering												✓			
TE.Trusted_System													✓		
TE.User_Error											✓				

Table 7-1 - Mapping the Threats to Objectives

Environmental Objectives / Assumptions	OE.Protect	OE.Key_Distribution	OE.Trusted_System	OE.Trained_Staff	OE.Secure_Drivers	OE.Card_Security	OE.Card_OS
A.Protect	✓						
A.Card_Security						✓	
A. Key_Distribution		✓					
A.Trained_Staff				✓			
A.Card_Link					✓		
A.Trusted_System			✓				
A.Card_OS							✓

Table 7-2 - Mapping the Assumptions to Environmental Objectives

7.1.2 Sufficiency of Security Objectives

Threat	Objectives
T.CryptAnalysis	The objective (O.Cryptographic, O.Key_Management, OE.Crypto_Primitives) provides complete coverage as: • The TOE will use trusted cryptographic algorithms which are sufficient to withstand the cryptanalysis capabilities of the majority of likely attackers. • Keys are generated from a random number generator that provides sufficient entropy so keys are not predictable and the difficulty in breaking keys is accurately reflected in key lengths. • The cryptography in the TOE is based on correctly implemented cryptographic primitives suitable to provide an appropriate level of protection. • The key management will provide sufficient protection for the intended use of the keys. • Keys can be securely deleted when they are no longer required so that no more information that could be used for cryptanalysis is generated.
T.Object_Reuse	The objective (O.No_Residual_Info) provides complete coverage as: • Secret cryptographic key and password information stored on media that can be reallocated to other users is destroyed before the media is re-allocated.

T.Power_Disruption	The objective (O.Secure_State) provides complete coverage as: • After a system error, or interruption to operations, as would occur with a power failure, the system maintains secure operations.
T.Repudiation	The objective (O.Key_Management, O.Feedback) provides complete coverage with the non-repudiation key is used, as: • Keys are managed so that a key intended for non-repudiation is identified and suitably protected to generate 'non-repudiation' signatures. • The user must authenticate the data to be signed before the signature is formed. Hence the TOE will not generate or pass on a signature formed with the non-repudiation key for data that the user has not confirmed.
T.Unauth_Access	The objectives (O.Access_Control, O.I&A O.Key_Management) provides complete coverage as: • The TOE has a means of restricting access to assets. • Users are identified to the TOE and can be used as the basis of access control. • Rights to security-critical operations and sensitive data in the TOE are restricted so that only authenticated users are allowed access. • The TOE identifies the key usage so that appropriate access restrictions can be placed on keys.
TE.Envioronment	The objective (OE.Protect) provides complete coverage as: • The operating environment sufficiently protected to resist attacks from attackers who have a high level of skill and motivation.
TE.Card_Extract	The objective (OE.Card_Security) provides complete coverage as: • The card used by the TOE is protected from physical attacks and electromagnetic emanations do not carry detectable information.

TE.Card_Hack	The objective (OE.Card_Security and OE.Card_OS) provides complete coverage as: • The smart card used by the TOE contains sufficiently high security mechanisms to protect itself against disclosure of information through electromagnetic emanations and physical attacks on the card. • An evaluation provides assurance that development, manufacturing and delivery processes are suitable to meet security objectives. • Applications can be loaded onto the card in a manner protected from modification and disclosure of information. • The card operating system protects applications from interference from other applications. • The operating system ensures that no data is available once an application is deleted.
TE.Key_User	The objectives (OE.Key_Distribution and O.I&A) provide complete coverage as: • User's known public keys are distributed in a timely manner to any party who relies on the user's public key. • When the association between a user and public key is no longer valid (possibly because the card was lost or stolen), the binding is revoked and this information is distributed to relying parties in a timely manner. The objective (O.I&A) protects against unauthorised access in the time period between when the card is lost or stolen and when the binding between the user and public key is removed.
TE.Driver_Tampering	The objective (OE.Secure_Drivers) provides complete coverage as: • The third party drivers will not allow protected information to be read or interfered with, and data is not intercepted as it is passed on to the smart card.
TE.Trusted_System	The objective (OE.Trusted_System) provides complete coverage as: • The system on which the TOE is installed implements its interface to the TOE correctly.
TE.User_Error	The objective (OE.Trained_Staff) provides complete coverage as: • TOE users and administrators will follow proper procedures for setting up and operating the TOE and the smart cards, and • TOE users and administrators will be aware of the security implications in not keeping smart cards and passwords secure.

Table 7-3 – Sufficiency of Objectives to counter threats

Assumption	Objectives
A.Protect	The objective (OE.Protect) upholds the assumption as: • The TOE has to operate in an environment sufficient to resist

	attackers with high levels of skill and motivation.
A.Card_Security	The objective (OE.Card_Security) upholds the assumption as: • The smart card used by the TOE has to protect itself against disclosure of information through electromagnetic emanations and physical attacks on the card.
A. Key_Distribution	The objective (OE.Key_Distribution) upholds the assumption as: • Users must be correctly associated with public keys, and this association needs to be distributed and revoked as necessary.
A.Card_Link	The objective (OE.Secure_Drivers) upholds the assumption as: • Information must not be read or modified within card readers and driver, and information must not be accessed as it passes to the smart card.
A.Trusted_System	The objective (OE.Trusted_System) upholds the assumption as: • The system on which the TOE is installed interfaces with the TOE through its published interfaces which must be implemented correctly.
A.Trained_Staff	The objective (OE.Trained_Staff) upholds the assumption as: • The staff responsible for the administration of the TOE, and users responsible for the operation of the TOE must follow provided guidance.
A.Card_OS	The objective (OE.Card_OS) upholds the assumption as: • The card operating system is shown to be trustworthy (which can be done through evaluation) and must protect on-card applications during loading and operation, and clear all application data after the application is deleted.

Table 7-4 – Sufficiency of Environmental Objectives to uphold Assumptions

7.2 Security Requirements Rationale

The purpose of this section is to show that the identified security requirements (Section 5) are *suitable* to meet the security objectives (Section 4). The following tables show that each security requirement (and SFRs in particular) is *necessary*, that is, each security objective is addressed by at least one security requirement, and vice versa. The shaded areas of the Table 7-5 indicate SFR's addressed by environmental objectives.

SFRs \ Objectives	O.Access_Control	O.Cryptographic	O.Feedback	O.I&A	O.Key_Management	O.No_Residual_Info	O.Secure_State	OE.Crypto_Primitive
FCS_CKM.1		✓			✓			
FCS_CKM.4					✓			
FCS_COP.1/RSA		✓			✓			
FCS_COP.1/Verify		✓		✓				
FDP_ACC.2/Keys	✓		✓					
FDP_ACC.2/Card data	✓							
FDP_ACF.1/Keys	✓		✓		✓		✓	
FDP_ACF.1/Card data	✓							
FDP_ITC.1/Keys					✓			
FDP_ITC.1/Card data					✓			
FDP_DAU.2			✓					
FDP_RIP.1				✓	✓	✓		
FIA_AFL.1				✓				
FIA_UAU.1				✓	✓			
FIA_UAU.4				✓				
FIA_UAU.6			✓	✓			✓	
FIA_UID.2				✓				
FMT_MSA.1				✓	✓			
FMT_MSA.2				✓	✓			
FMT_MSA.3	✓			✓	✓			
FMT_SMR.1				✓				
FPT_FLS.1					✓		✓	
FCS_COP.1/Primitives								✓
FCS_COP.1/Random								✓

Table 7-5 – Objectives met by Security Requirements

Objectives	Requirements
O.Access_Control	The SFRs [FDP_ACC.2/Keys, FDP_ACC.2/Card data, FDP_ACF.1/Keys, FDP_ACF.1/Card data, FMT_MSA.3] are sufficient to satisfy the objective because: • All user access to key pairs and certificates stored on the card can be restricted based on passwords, access permissions and private key attributes.[FDP_ACC.2.1/Keys, FDP_ACC.2.1/Card data, FDP_ACF.1/Keys, FDP_ACF.1/Card data]

	• Access to a user's private keys is only permitted for the user that knows the password. [FDP_ACF.1/Keys] • Complete access control is enforced on all operations between any subject and any object. [FDP_ACC.2.2/Keys, FDP_ACC.2.2/Card data] • Users or issuers can set the security attributes that restrict access to objects when creating objects [FMT_MSA.3].
O.Cryptographic	The SFRs [FCS_COP.1/RSA, FCS_COP.1/Verify, FCS_CKM.1] are sufficient to satisfy this objective because: • Data is appropriately encrypted, hashed, and signatures verified with DSD approved standards. [FCS_COP.1/Verify] • DSD approved standards apply to cryptographic protocols or modes of operation used to decrypt or sign data [FCS_COP.1/RSA] • The keys that the TOE generates are strong enough to resist cryptanalysis of likely attackers. [FCS_CKM.1]
O.Feedback	The SFRs [FDP_DAU.2, FDP_ACF.1/Keys, FDP_ACC.2/Keys, FIA_UAU.6] are sufficient to satisfy the objective because: • The TOE generates evidence that can be used as a guarantee of the validity of the received data to be signed with a non-repudiation key (the data to be signed is either displayed to the user, or saved to a file for inspection to provide the user with confidence in the information being signed). [FDP_DAU.2.1] • Users can verify the authenticity of the data to be signed and that they are the creators of the data to be signed. [FDP_DAU.2.2] • The only cryptographic operation with a non-repudiation key is to generate signatures for messages that the user authenticated. [FDP_ACF.1.2/Keys] • The TOE checks that the user has confirmed the operation by re-authenticating the user. [FIA_UAU.6] • All other operations that could result in a signature generated with a non-repudiation key are controlled. [FDP_ACC.2.2/Keys]
O.I&A	The SFRs [FCS_COP.1/Verify, FDP_RIP.1, FIA_AFL.1, FIA_UAU.1, FIA_UAU.4, FIA_UAU.6, FIA_UID.2, FMT_SMR.1, FMT_MSA.1, FMT_MSA.2, FMT_MSA.3] are sufficient to satisfy the objective because: • Users must be identified. [FIA_UID.2] • Unauthenticated users are only allowed operations on assets that do not affect the security of protected assets. [FIA_UAU.1] • The user can establish a password for authentication before protected assets are created for that user or are loaded with data, hence assets can be protected even before they are fully initialised. [FIA_UAU.1.1]

	• The users are allowed a limited number of consecutive attempts to enter a correct password. [FIA_UAU.1.1] • Invalid users are limited in the number of successive failed entry attempts of an authentication password. [FIA_AFL.1.1] • Exceeding that limit results in the blocking of access to assets protected by the password. [FIA_AFL.1.2] • Following authentication, the card holder is associated with user or issuer roles. [FMT_SMR.1] • Only the role of issuer is allowed to unblock a password. [FMT_MSA.1] • Stronger authentication is needed to unblock a password. [FIA_UAU.4] • Strong cryptography is used to implement a single use password mechanism and to store passwords. [FCS_COP.1/Verify] • Password attributes can be set to control the difficulty in subverting the authentication process and the protection applied to passwords can be verified by users. Only secure values are accepted for password attributes. [FMT_MSA.1, FMT_MSA.2, FMT_MSA.3] • Users are re-authenticated whenever an event occurs that would put the current authentication in doubt or where authentication is critical. [FIA_UAU.6] • Users have the ability to change a password should it be exposed. [FMT_MSA.1] • Passwords are protected from disclosure to other users upon reallocation of resources from the TOE. [FDP_RIP.1]
O.Key_Management	The SFRs [FCS_CKM.1, FCS_CKM.4, FCS_COP.1/RSA, FDP_ACF.1.2/Keys, FDP_ITC.1/Keys, FDP_ITC.1/Card data, FDP_RIP.1, UAU.1, FMT_MSA.1, FMT_MSA.2, FMT_MSA.3, FPT_FLS.1] are sufficient to satisfy the objective because: • Private keys have usage attributes that can be set according to the intended purpose of the key. [FMT_MSA.3.2, FMT_MSA.1] • The key usage attribute defines the cryptographic operations that are allowed on the private key. [FDP_ACF.1.2/Keys] • Only secure values can be set for security attributes while defaults are suitably restrictive. [FMT_MSA.3.1, FMT_MSA.2] • The TOE controls the objects to which key pairs and data related to keys can be imported, and these objects can have appropriate security attributes set at creation. [FDP_ITC.1/Keys, FDP_ITC.1/Card data, FMT_MSA.3.2] • Private keys can be imported into the TOE in split knowledge form using strong cryptography. [FDP_ITC.1.3/Keys, FCS_COP.1/RSA] • The TOE can generate secure key pairs on behalf of the user [FCS_CKM.1].

	• Private keys loaded to, or generated in the TOE, cannot be read off the card, ensuring their confidentiality. [FDP_ACF.1.4/Keys] • The TOE only allows certificates to be imported if they match a stored key pair. This ensuring that certificates retrieved apply to protected keys stored in the TOE. [FDP_ITC.1.3/Card data] • The TOE allows unencumbered access to information which is needed to verify the correct configuration of the TOE allowing all interested parties to check, on behalf of the user, protection applied. [UAU.1.1, FMT_MSA.1] • Cryptographic keys in RAM are destroyed before the RAM can be reallocated to other users. [FDP_RIP.1] • The TOE has the ability to securely destroy stored private keys under user control. [FCS_CKM.4] • The system will go to a secure state when the smart card is removed at any stage during the operations so the protected data stored on the card is no longer used. [FPT_FLS.1]
O.No_Residual_Info	The SFR [FDP_RIP.1] is sufficient to satisfy the objective because: • Data which users require to be confidential is erased before the containers they are in are re-allocated to other users. This protection extends to numbers produced from the random number generator to which the TOE provides an interface, since these numbers are typically used for session keys. [FDP_RIP.1]
O.Secure_State	The SFRs [FDP_ACF.1/Keys, FIA_UAU.6, FPT_FLS.1] are sufficient to satisfy the objective because: • Private keys are stored and protected by the smart card [FDP_ACF.1/Keys]. • The smart card is kept in a secure state following a reset or power interruption [FPT_FLS.1.1]; while information in volatile memory is naturally lost when power is cut. • The user must re-authenticate following a reset of power interruption to the card [FIA_UAU.6]. • The system is in a secure state if the user has not been authenticated [FPT_FLS.1.1].

Table 7-6 – SFR Sufficiency for TOE Objectives

7.2.1 Security Requirements for IT Rationale

The MULTOS smart card environmental security requirements map to the OE.Crypto_Primitives Environmental Objective. The MULTOS smart card provides cryptographic primitives and a random number generator which are available to card applications running on the card. The MULTOS smart card has been evaluated to Assurance Level E6, which provides confidence that the card can satisfactorily uphold this objective.

Objectives	Requirements
OE.Crypto_Primitives	The SFRs [FCS_COP.1/Primitives, FCS_COP.1/Random] are sufficient to satisfy this objective because: • The cryptographic primitives correctly implement required cryptographic primitives. • The random number generator produces random numbers that meet requirements for distribution of generated numbers.

Table 7-7 – SFR Sufficiency for environmental objectives

7.2.2 TOE Security Functions Rationale

	TSF1	TSF2	TSF3	TSF4	TSF5	TSF6
FCS_CKM.1				✓		
FCS_CKM.4	✓			✓		
FCS_COP.1/RSA	✓					
FCS_COP.1/Verify	✓					
FDP_ACC.2/Keys			✓			
FDP_ACC.2/Card data			✓			
FDP_ACF.1/Keys			✓	✓		
FDP_ACF.1/Card data			✓			
FDP_ITC.1/Keys				✓		
FDP_ITC.1/Card data				✓		
FDP_DAU.2						✓
FDP_RIP.1	✓		✓			
FIA_AFL.1		✓				
FIA_UAU.1		✓	✓	✓	✓	
FIA_UAU.4		✓				
FIA_UAU.6		✓			✓	✓
FIA_UID.2		✓				
FMT_MSA.1				✓	✓	
FMT_MSA.2				✓	✓	
FMT_MSA.3				✓	✓	
FMT_SMR.1		✓				
FPT_FLS.1		✓	✓			

Table 7-8 - SFR - TSF Cross Reference

From Table 7-8, it is clear that every TSF contributes to at least one SFR so every TSF is necessary. Also, each SFR is supported by at least one TSF. The mapping of security functions to SFR is provided in the description of security functions in section 6.1, and as a whole, the security functions satisfy all SFRs. The justification for the suitability of TSF to satisfy SFRs is given below.

7.2.2.1 TSF1 – Cryptographic

FCS_CKM.4: Keys in RAM are destroyed as soon as operations on those keys are completed, while keys on the smart card can be overwritten with zeros.

FCS_COP.1/RSA: - Authenticated users can use the private keys stored on smart cards to perform cryptographic operations. The cryptographic operations comply with the stated standards. Encrypted private keys can be decrypted with a 56 bit DES key or a 112 bit triple DES key.

FCS_COP.1/Verify: TrustedNet Connect calculates digests of messages and passwords. Cryptographic operations can be performed on public keys in accordance with the stated standards.

FDP_RIP.1: Decrypted data and session-keys in RAM are overwritten with zeros as soon as cryptographic operations are complete.

7.2.2.2 TSF2 – Identity and Authentication

FIA_AFL.1: Should users fail to be adequately identified, or fail to authenticate themselves within the allowed number of authentication retries for a password, the password is blocked. Authentication is disallowed for blocked passwords and access to resources protected by a password that is blocked is prohibited. More than the allowed maximum of successive authentication failures with the unblock password causes the unblock password to be permanently blocked.

FIA_UAU.1: Before they are authenticated, users are allowed to present passwords for authentication until they exhaust the allowed number of authentication retries. Passwords can be handled securely within TrustedNet Connect.

FIA_UAU.4: Verification data for unblock passwords change after each successful authentication, and given that passwords have unique verification data, the value of the password must also changes after each successful authentication.

FIA_UAU.6: Passwords have an attribute that limits the validity of an authentication to a single operation necessitating re-authentication. Re-authentication of users is enforced when client application request services. Information about whether a user has been authenticated against a password is cleared upon card re-insertion, power-up and after resets. The user must be re-authenticated to regain access to operations protected with passwords.

FIA_UID.2: The user is identified through their smart card so no access to controlled user data is available until the user identifies themselves by inserting their smart card.

FMT_SMR.1: Users are assigned the roles of “Users” and “Issuers”.

FPT_FLS.1: Following an authentication failure against a password, access to resources protected by that password is denied. Information about the number of authentication failures is not lost following card removal, power-down or reset, and the restriction on the number of authentication attempts is enforced.

7.2.2.3 TSF3 – Data Protection

FDP_ACC.2/Keys, FDP_ACC.2/Card data: - All access to protected information can be gained only through TrustedNet Connect.

FDP_ACF.1/Keys: Access to private keys is prohibited if, for a cryptographic operation, the user is not authenticated or the access permission prohibits the operation. Stored private keys can be decrypted with the correct key, but private keys cannot be read off the card.

FDP_ACF.1/Card data Access permissions control access to public keys and certificates.

FDP_RIP.1: Passwords, including unblock passwords, private keys, key decrypting keys, random numbers, decrypted data and session-keys in RAM are overwritten with zeros before the memory is re-allocated to other users.

FIA_UAU.1: Access permissions can grant users access to objects before the user is authenticated. Other access to key pairs or certificates is prohibited or requires authentication. Users can also access the random number generator before they are authenticated.

FPT_FLS.1: Should the card be powered down or removed during operation, the TSF ensures that the state of the card remains secure. All access to protected data on the card is automatically removed since the data is no longer available.

7.2.2.4 TSF4 –Key Management

FCS_CKM.1: The TSF is capable of generating 1024 bit RSA keys.

FCS_CKM.4: The TSF is capable of destroying stored keys.

FDP_ACF.1/Keys: Cryptographic operations on private keys are only performed if the operation is compatible with the key usage attribute and the key is not encrypted. Signatures can be formed with a non-repudiation key on a message that the user has authenticated. Imported private keys can only be stored in smart cards.

FDP_ITC.1/Keys: Private keys or encrypted private keys can be imported into objects that allow the operation. Any attributes associated with imported private keys are ignored as the security attributes of the target object apply.

FDP_ITC.1/Card data: Certificates and key pairs can be imported only if access permissions for the target object are met. Any attributes associated with imported private keys are ignored as the security attributes of the target object apply. Certificates are only imported if a matching key pair can be found on the smart card.

FIA_UAU.1: Users can read the configuration settings and status of keys before they are authenticated.

FMT_MSA.1: Operations on security attributes are appropriately restricted to the specified roles. Key configuration and status details can be read.

FMT_MSA.2: Restrictions are placed on security attributes to ensure secure operations.

FMT_MSA.3: Users and Issuers can set initial values for security attributes when objects are created. These defaults restrict access to operations that effect stored data to authenticated users or prohibit those operations, ensure that the private key is protected with a password, and the default key usage does not allow access to the entire block of RSA decrypted data.

7.2.2.5 TSF5 – Password Management

FIA_UAU.1: Users can set the initial value of passwords before they are authenticated. Users can read the configuration settings and status of passwords before they are authenticated.

FIA_UAU.6: Users need to be re-authenticated for password changes and unblocking of passwords.

FMT_MSA.1: Operations on security attributes are appropriately restricted to the specified roles. Password configuration and status details can be read.

FMT_MSA.2: Restrictions are placed on security attributes to ensure secure operations.

FMT_MSA.3: Issuers and users can set initial values for security attributes when objects are created. By default, passwords have restrictions placed on their possible values and the number of authentication retries is set to a small value.

7.2.2.6 TSF6 – Non-Repudiation Key

FDP_DAU.2: Messages to be signed can be either displayed or saved for later inspection as evidence of valid messages. Users can inspect the messages to ensure the messages are ones that they created and intend to sign.

FIA_UAU.6: Users must authenticate themselves before a signature can be generated with a non-repudiation key.

7.2.3 Satisfaction of SFR Dependencies

Table 7.9 below illustrates the TOE SFRs their dependencies and the SFRs that satisfy those dependencies. 'None' following an SFR indicates that it has no dependencies and is included for the sake of completeness.

SFR	Dependencies	Satisfied by	Notes
FCS_CKM.1	FCS_CKM.2 or FCS_COP.1	FCS_COP.1/RSA	
	FCS_CKM.4	FCS_CKM.4	
	FMT_MSA.2	FMT_MSA.2	
FCS_CKM.4	FDP_ITC.1 or FCS_CKM.1	FDP_ITC.1/keys and FCS_CKM.1	Both present
	FMT_MSA.2	FMT_MSA.2	
FCS_COP.1/RSA	FDP_ITC.1 or FCS_CKM.1	FDP_ITC.1/keys and FCS_CKM.1	Both present
	FCS_CKM.4	FCS_CKM.4	
	FMT_MSA.2	FMT_MSA.2	
	FCS_COP.1	FCS_COP/Primitives	Cryptographic protocols are dependent on implementation of primitives
FCS_COP.1/Verify	FDP_ITC.1 or FCS_CKM.1		See section 7.2.4
	FCS_CKM.4		See section 7.2.4
	FMT_MSA.2		See section 7.2.4

	FDP_RIP.1	FDP_RIP.1	See section 7.2.4
FDP_ACC.2/Keys	FDP_ACF.1	FDP_ACF.1/Keys	
FDP_ACC.2/Card data	FDP_ACF.1	FDP_ACF.1/Card data	
FDP_ACF.1/Keys	FDP_ACC.1	FDP_ACC.2/Keys	hierarchical to FDP_ACC.1
	FMT_MSA.3	FMT_MSA.3	
FDP_ACF.1/Card data	FDP_ACC.1	FDP_ACC.2/Card data	hierarchical to FDP_ACC.1
	FMT_MSA.3	FMT_MSA.3	
FDP_ITC.1/Keys	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.2/Keys	hierarchical to FDP_ACC.1
	FMT_MSA.3	FMT_MSA.3	
FDP_ITC.1/Card data	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.2/Card data	hierarchical to FDP_ACC.1
	FMT_MSA.3	FMT_MSA.3	
FDP_DAU.2	FIA_UID.1	FIA_UID.2	Hierarchical to FIA_UID.1
FDP_RIP.1	None		
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1	
FIA_UAU.1	FIA_UID.1	FIA_UID.2	Hierarchical to FIA_UID.1
FIA_UAU.4	None		
FIA_UAU.6	None		
FIA_UID.2	None		
FMT_MSA.1	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.2/Keys & FDP_ACC.2/Card data	hierarchical to FDP_ACC.1
	FMT_SMR.1	FMT_SMR.1	
FMT_MSA.2	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.2/Keys & FDP_ACC.2/Card data	hierarchical to FDP_ACC.1
	FMT_MSA.1	FMT_MSA.1	
	FMT_SMR.1	FMT_SMR.1	
	ADV_SPM.1	EAL4	Part of EAL4
FMT_MSA.3	FMT_MSA.1	FMT_MSA.1	
	FMT_SMR.1	FMT_SMR.1	
FMT_SMR.1	FIA_UID.1	FIA_UID.2	Hierarchical to FIA_UID.1

FPT_FLS.1	ADV_SPM.1	EAL4	Part of EAL4
FCS_COP/Primitives	FDP_ITC.1 or FCS_CKM.1		See section 7.2.4
	FCS_CKM.4		See section 7.2.4
	FMT_MSA.2		See section 7.2.4
FCS_COP/Random	FDP_ITC.1 or FCS_CKM.1		See section 7.2.4
	FCS_CKM.4		See section 7.2.4
	FMT_MSA.2		See section 7.2.4

Table 7.9 – Satisfaction of SFR Dependencies

7.2.4 Justification for Unsupported Dependencies

The security functional dependencies for the TOE and its environment are not completely fulfilled by security functional requirements in sections 5.1 and 5.2

SFR	Rationale
FCS_COP/Verify	The SFRs [FDP_ITC.1, FCS_CKM.1, FCS_CKM.4, FMT_MSA.2] are not required because: • These operations are performed on data entirely provided by the user specifically for the operation and does not involve secret keys. Hence, these operations do not require any secret keys to be generated, imported, or deleted, or security management. Session keys which are passed in for encryption are treated as user data and protected under FDP_RIP.1
FCS_COP/Primitives	The SFRs [FDP_ITC.1, FCS_CKM.1, FCS_CKM.4, FMT_MSA.2] are not required because: • The TOE uses the operations to implement its cryptographic operations. • The TOE handles all keys and security management for these operations, and dependencies are met for the FCS_COP/RSA SFR that uses these primitives.
FCS_COP/Random	The SFRs [FDP_ITC.1, FCS_CKM.1, FCS_CKM.4, FMT_MSA.2] are not required because: • The random number generator does not require any keys or security management.

Table 7.10 – Justification of Unsupported Dependencies

7.2.5 Assurance Security Requirements Rationale

EAL4 was chosen to provide a moderate level of independently assured security. The chosen assurance level is consistent with the postulated threat environment. The threat of malicious attacks in the intended environment is not considered to be greater than moderate, and the product undergoes a search for obvious flaws.

EAL4 provides objective analysis of the design of the TOE through high level and low level assurance requirements and also through a subset of the source code. This will provide the evaluators with sufficient information to understand the security functions.

EAL4 enforces independent confirmation of the developers test results and independent testing of the security functions, which should provide confidence in the correctness of the functions.

The development life-cycle of the TOE is examined to demonstrate that the TOE is developed in a controlled and secure manner.

The Security Assurance requirements relevant to this security target are drawn from CC Part 3 EAL4 assurance requirements.

7.2.6 Strength of Function Claims

Strength of function claims are subject to proper generation, installation and operation of the TOE in accordance with provided guidance.

The TOE mechanisms on the smart card will resist technical attacks by unauthorised users. The password policy (for card-access) enforced by the TOE:

- allows alphanumeric mixed-case characters for passwords which are verified against a cryptographic digest of the password stored on the card;
- enforces minimum lengths for passwords;
- protects against the selection of 'weak' passwords;
- allows only a pre-set limited number of incorrect attempts before blocking the password, and
- has mechanisms in place to ensure that the password protection mechanisms cannot be de-activated.

Correct decryption of stored encrypted private keys is verified through a check of recovered redundancy. The amount of redundancy is such that in practice only the correct decryption key will result in this redundancy being uncovered correctly.

Consequently, a level of high function strength (SoF-High) applies which indicates that a function provides adequate protection against sophisticated and intentional breaches of TOE security by attackers possessing a high attack potential. This is necessary since smart cards can be lost or stolen and attacked outside the normal IT environment of the TOE. The SoF-High is also consistent with the assumed physical protection of the smart card and smart card operating system security features. The smart card security features ensure that data cannot be read from the card and that the code on the card cannot be modified to bypass or deactivate security mechanisms.

This claim of SoF-High is also appropriate for the FIA_UAU family, and TSF relating to the identification and authentication of users of the TOE (TSF2 – Identity and Authentication).

It is not appropriate for cryptographic SoF claims to be made in this ST, as evaluation of strength of cryptographic functions is the responsibility of DSD (as the National Comsec Authority). This claim corresponds to the FCS class of SFR, and the cryptographic IT Security Functions (TSF1 – Cryptographic Services).

7.2.7 Mutually Supportive Security Requirements

The purpose of this rationale is to show that the IT security requirements (and the SFRs in particular) are complete and internally consistent by demonstrating that they are mutually supportive and provide an ‘integrated and effective whole’.

Dependency helps in showing mutual support because if SFR-A is dependent on SFR-B then by definition, SFR-B is supportive of SFR-A. Table 7.9 shows the dependencies of the Security Functional Requirements. The table along with Section 7.2.4 shows that all necessary dependencies are satisfied.

This ST is targeting a standard EAL4 assurance package and so the dependency and mutual support of the assurance requirements is self-evident as the EAL is taken from the CC.

For those SFRs not directly related by dependency, mutual support can be provided by SFRs which address the following issues.

7.2.7.1 Help prevent bypassing of other SFRs

FIA_UID.1 and FIA_UAU.1 support other functions which rely on authentication to restrict user’s access to assets. FIA_UAU.6 supports authentication since users are re-authenticated at critical operations and instances where authentication may no longer be valid.

FMT_MSA.2 and FMT_MSA.3 limit the acceptable values for secure data, protecting the SFRs dependent on those values from being bypassed.

FPT_FLS.1 ensures that certain failures will not lower the level of security provided.

FDP_RIP.1 prevent inadvertent disclosure of passwords between users.

The FDP_ACC.2 SFRs ensure that there is no uncontrolled alternative for performing an operation.

7.2.7.2 Help prevent tampering of other SFRs

The cryptographic functions FCS_CKM.1, FCS_CKM.4 and FCS_COP.1 provide for the secure generation, destruction and operation of keys, and therefore support those SFRs which may rely on the use of those keys.

FIA_UID.2 and FIA_UAU.1 support other functions which allow the user access to the assets by restricting the actions the user can take before being authorised.

FMT_MSA.2 and FMT_MSA.3 limit the acceptable values for secure data, protecting the SFRs dependent on those values from being tampered with. FMT.MSA.1 limits the ability to modify security attributes and lower the level of security provided.

7.2.7.3 Help prevent de-activation of other SFRs

The Access Control policy detailed in FDP_ACC.2 and FDP_ACF.1 SFRs along with the other SFRs involved in access control, provide for rigorous control of allowed access, preventing unauthorised deactivation of SFRs.

ADO_IGS.1 included as part of EAL4, ensures that any data loaded into the smart card at generation will not de-activate SFRs.

FMT_MSA.2 and FMT_MSA.3 limit the acceptable values for secure data, protecting the SFRs dependent on those values from being de-activated.

7.2.8 Mutually Supportive Security Functions

The information provided in Section 6 does not undermine the mutual support of the SFRs in any way. The information presented in the TSFs does not introduce any potential security weaknesses.