IBM Global Security Kit Version 7.0.4.11 Security Target

Document Version: 1.7

Status: Final

Last Update: 2007-07-26

Document History

Version	Date	Changes	Summary	Author
1.7	2007-07-26	See Summary	Removed confidentiality labels and confidential parts of the document history. Clarified build number in section 1.1.	David Ochel

Table of Content

References

[CC]	Common Criteria for Information Technology Security Evaluation. Part 1-3. August 2005. Version 2.3.
[CEM]	Common Methodology for Information Technology Security Evaluation. August 2005. Version 2.3.
[DOD]	Global Security Kit Delivery Procedure Additional Guidance, Version 7.0 as of 2006-07-20
[PD-0108]	CCEVS Precedence PD-0108: FTP_ITC.1.3 Specifies The Functions For Which A Trusted Channel Is Provided, Effective Date: 2004-07-19, Last Modified Date: 2004-08-26.
[RI 58]	Final Interpretation for RI # 58 - Confusion over refinement, 07/31/2001
[TARGET]	Global Security Kit V7d Security Target (this document)
[EPRAND]	EUROPEAN PATENT APPLICATION EP 1 081 591 A2, Random number generator, Application number: 00114754.5, Date of publication: 07.03.2001 Bulletin 2001/10.
[FIPS46-3]	FIPS PUB 46-3: DATA ENCRYPTION STANDARD (DES), October 25, 1999.
[FIPS81]	FIPS PUB 81: DES MODES OF OPERATION, Issued December 2, 1980, including CHANGE NOTICES 2 and 3
[FIPS140-2]	FIPS PUB 140-2: SECURITY REQUIREMENTS FOR CRYPTOGRAPHIC MODULES, Issued May 25, 2001, including CHANGE NOTICES (12-03-2002)
[FIPS140IG]	Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, Initial Release: March 28, 2003, Last Update: July 26, 2004
[FIPS180-2]	FIPS PUB 180-2: Specification for the SECURE HASH STANDARD, including Change Notice to include SHA-224, August 1, 2002
[FIPS186-2]	FIPS PUB 186-2: DIGITAL SIGNATURE STANDARD (DSS), including Change Notice, January 27, 2000
[FIPS197]	FIPS PUB 197: Specification for the ADVANCED ENCRYPTION STANDARD (AES), November 26, 2001.
[FIPS198]	FIPS PUB 198: The Keyed-Hash Message Authentication Code (HMAC), March 6, 2002.
[GSKCAPI]	IBM Global Security Kit GSKCapiCmd User’s Guide GSKit Version 7d Edition March 4, 2005
[GSKCCMODE]	Global Security Kit Common Criteria Mode Operating Guidance for Version 7d Edition March 10, 2005
[GSKTRUST]	IBM Global Security Kit Certificate Validation and Trust Policy Design for version 7c, May 30, 2006
[GSKKEY]	IBM Global Security Kit Key Management for C Programmer's Guide Version 7c, Edition February 23, 2005.
[GSKINST]	IBM Global Security Kit Global Security Kit Install and Packaging Guide, Version 7c, March 6, 2005.
[GSKPLI]	Performing GSKit Local Installations, Version 7.0 as of 2006-08-03.
[GSKSSL]	IBM Global Security Kit, Secure Socket Layer for C Programmer’s Guide, Version 7c, Edition March 9, 2005
[ICC]	IBM Crypto for c (ICC) Version 1.2 Design Document Version 0.7 – December 29, 2004
[ICCDESIGN]	IBM Crypto for C (ICC) Version 1.4 Design Document Version 1.4 – March 10,2007
[ICCSEC]	IBM Crypto for C (ICC) Version 1.4.4 FIPS 140-2 Non-Proprietary Security Policy, version 0.7 October 4, 2006.
[GUIDE]	ISO/IEC PDTR 15446 Title: Information technology – Security techniques – Guide for the production of protection profiles and security targets, ISO/IEC JTC 1/SC 27 N 2449, 2000-01-04.
[PKCS#11-2.10]	PKCS #11 v2.10: Cryptographic Token Interface Standard. RSA Laboratories, December 1999.
[PKCS#11-2.20]	PKCS #11 v2.20: Cryptographic Token Interface Standard. RSA Laboratories, 28 June 2004.

[PKCS#12]	PKCS 12 v1.0: Personal Information Exchange Syntax, RSA Laboratories, June 24, 1999.
[RFC1319]	RFC 1319: The MD2 Message-Digest Algorithm, including the Erratum for RFC 1319, April 1992
[RFC1321]	RFC 1321: The MD5 Message-Digest Algorithm, including the Erratum for RFC 1321, April 1992
[RFC2313]	RFC 2313: PKCS#1: RSA Cryptography Specification, Version 1.5, March 1998.
[RFC2401]	RFC 2401: HMAC-Keyed-Hashing for Message Authentication, February 1997.
[RFC2437]	RFC 2437: PKCS #1: RSA Cryptography Specifications, Version 2.0, October 1998.
[RFC2560]	RFC 2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP. June 1999
[RFC2986]	RFC 2986: PKCS #10: Certification Request Syntax Specification, Version 1.7, November 2000
[RFC3280]	RFC 3280: Internet X.509 Public Key Infrastructure - Certificate and Certificate Revocation List (CRL), obsoletes RFC 2459, April 2002.
[SSLv3]	Alain O. Freier, Philip Karlton, Paul C. Kocher: The SSL Protocol, Version 3; IETF Memo, Internet Draft, November 1996.
[TLSv1]	T. Dierks, C.Allen: The TLS Protocol Version 1.0; RFC 2246, January 1999.
[TLS_AES]	P. Chown: Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS); RFC 3268, June 2002.
[X.509]	ITU-T RECOMMENDATION X.509 | ISO/IEC 9594-8: INFORMATION TECHNOLOGY - OPEN SYSTEMS INTERCONNECTION - THE DIRECTORY: PUBLIC-KEY AND ATTRIBUTE CERTIFICATE FRAMEWORKS.

1 Introduction

This document is the security target for the CC evaluation of Global Security Kit (GSKit). The version number 7d is used alternately throughout this document. 7d refers generically to all 7.0.4.x versions, where the x is a specific build number. For this evaluation, the complete version and build number is identified in section 1.1 below.

1.1 ST Identification

ST Title:	IBM Global Security Kit Version 7.0.4.11 Security Target
ST Version:	1.7
ST Date:	2007-07-26
TOE Identification:	The TOE comprises the Global Security Kit (GSKit) Version 7.0.4.11, including the SSL API and Key Management API and CLI. GSKit encapsulates the IBM Crypto for C (ICC) Version 1.4.5 as an algorithm factory.
Keywords:	GSKit, SSL, TLS, ICC, OCSP, PKCS#11, PKCS#12.

1.2 CC Conformance Claim

This ST is *CC Part 2 extended* [CC] and *CC Part 3 conformant* [CC], with a claimed Evaluation Assurance Level of EAL4.

No conformance to a Protection Profile is claimed.

1.3 Strength of Function

The overall strength of function claim for this TOE is SOF-high.

1.4 ST and TOE Overview

This security target documents the security characteristics of the Global Security Kit (GSKit) Version 7d. GSKit is a set of tools and C/C++ programming interfaces that can be used to add secure channels using the SSLv3 and TLSv1 protocols to TCP/IP applications (products). It provides the cryptographic functions, the protocol implementation and key generation and management functionality for this purpose. GSKit ships only compiled object code and header files.

The TOE consists of GSKit, containing

- SSL and TLS functionality which offers an API (called SSL API) for SSLv3 (as defined in [SSLv3]) and TLSv1 (as defined in [TLSv1] with [TLS_AES]) connections and
- key and certificate generation and management functionality which offers an API (called Key Management API), and a command line interface (CLI). Key data is stored in a so-called keystore. A keystore is implemented as access controlled files in the TOE environment. The TOE can be configured to be in Common Criteria (CC) mode which ensures that the integrity and, where appropriate, the confidentiality of the data stored in the keystore is protected. Alternatively, PKCS#11 devices can be used for key and certificate storage. CC-mode must be used in the evaluated configuration.

Furthermore, GSKit encapsulates the IBM Crypto for C (ICC) component, which provides cryptographic functions. The ICC module is validated under the Federal Information Processing Standard (FIPS) 140-2 for an overall Security level 1 [FIPS140-2].

It is possible to configure the TOE such that

- only SSLv3 and TLSv1 are allowed, other versions of SSL are disabled,
- only SSL/TLS ciphersuites whose cipherspec parts consist of cryptographic algorithms that are FIPS-approved (as listed in Annex 1 of FIPS 140-2 [FIPS140-2]) and/or NIST-recommended and
- the FIPS-approved and/or NIST-recommended random number generator

are used. This approved mode of configuration is called FIPS mode. Please note that the cryptographic algorithms used for key exchange are not affected by these restrictions.

The product that deploys the TOE must initialize and use the TOE in FIPS mode and in CC mode.

The underlying operating systems for the evaluated configuration are identified in section 2.4.3. Operating systems running the TOE have to be configured to prevent remote login (by disabling all services that offer remote login) since this is required by the FIPS 140-2 security policy for ICC [ICCSEC].

The environment must provide reliable timestamps for certificate and CRL verification and an OCSP responder, if the TOE is configured to make use of these services.

The processing resources of the TOE must be located within controlled access facilities.

The evaluation assumes the operation of the GSKit in an environment that controls the access to the TOE. The TOE services, the GSKit software, TSF data, and the keystore must be protected. This also includes an appropriate protection of backup copies.

The use of OCSP responders in the IT environment is supported. Alternatively, the use of certificate revocation lists (CRLs) for certificate validation is available. The TOE can retrieve a CRL using LDAP, flat files or HTTP, check that the signatures are valid, and, if they are, will use this CRL for certificate validation. If CRLs are to be retrieved via LDAP or HTTP, an LDAP client and an LDAP server or, alternatively, an HTTP server must be available in the TOE environment to provide a current CRL containing all revoked certificates.

The TOE in the evaluated configuration must not use

- non-FIPS approved cryptographic functions of ICC for the cipherspec part of SSL/TLS ciphersuites,
- SSL versions prior to 3 (due to known weaknesses),
- “total anonymity mode” for SSL/TLS (i.e. server authentication is mandatory),
- BSAFE cryptographic library (by RSA).

The optional iKeyman GUI that is installed along with GSKit is not part of the TOE. It may be used as a key and certificate management interface within the product using the TOE (see Figure 1). So if it is to be used, it will be part of the TOE environment.

1.5 Structure

The structure of this document is as defined in the following.

- Section 1 is the introduction.
- Section 2 is the TOE description.
- Section 3 provides the statement of the TOE security environment.
- Section 4 provides the statement of security objectives.
- Section 5 provides the statement of security requirements.
- Section 6 provides the TOE summary specification which includes the detailed specification of the IT Security Functions.
- Section 7 provides the rationale for the security objectives, security requirements, and the strength of function claim.
- Section 8 resolves the abbreviations used.

1.6 Terminology

This section contains definitions of technical terms that are used with a meaning specific to this document. Terms defined in the [CC] are not reiterated here, unless stated otherwise.

GSKit: This term serves as an abbreviation for Global Security Kit Version 7d, which is the target of this evaluation.

Key Management API and CLI: GSKit API and CLI used by programs that want to deploy the key/certificate generation and management functionality of GSKit.

ICC: This is an abbreviation for IBM Crypto for C, a library that is used within GSKit.

Keystore: A keystore is a collection of flat files in the IT environment (on the underlying system) managed by the TOE to provides secure storage for key and certificate data.

OCSP: Online Certificate Status Protocol, defined in [RFC2560].

SHA-2: Common denominator for the SHA-224, SHA-256, SHA-384 and SHA-512 hash algorithms.

SSL: Secure Sockets Layer; this protocol is implemented by the GSKit and available through the SSL API.

SSL API: GSKit API used by programs that want to deploy the functionality of SSL or TLS.

TLS: Transport Layer security; this protocol is implemented by the GSKit and available through the SSL API.

Target of Evaluation (TOE): The TOE is defined as the GSKit Version 7d, running and tested in the environment specified in this Security Target.

User: A user is a human or a product/application using the TOE or the TOE environment, e.g. a system user uses the underlying operating system, a TOE user the TOE.

2 TOE Description

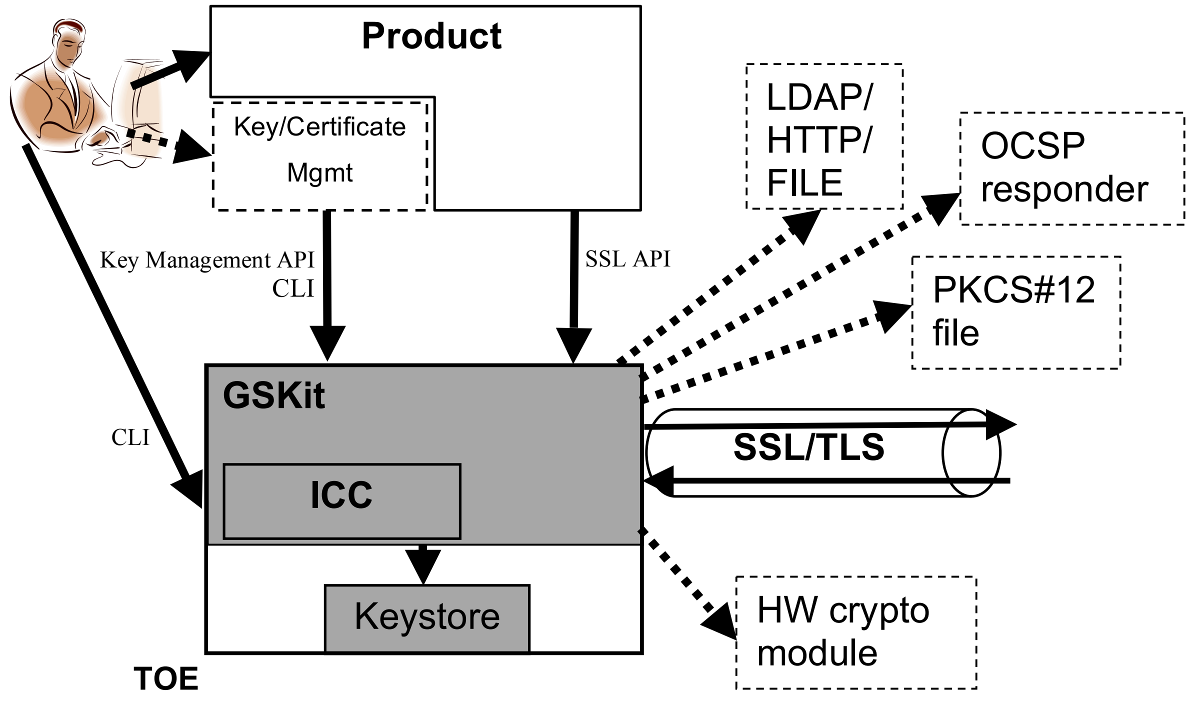

Figure 1: GSKit Overview

The TOE consists of the IBM Global Security Kit (GSKit).

The TOE's main function is to provide a secure channel to another trusted IT-product. This is shown in Figure 1. Optional components have been drawn with dotted lines and white components are part of the TOE environment. The TOE components have been colored dark grey and the box surrounding them shows the TOE boundary. The secure channel set up by the TOE is depicted as a pipe named SSL/TLS.

The following interfaces are offered to the user:

- SSL API: this is the main API for the product is the SSL API of the GSKit. For the evaluated configuration of the TOE, only secure connections using SSLv3 and TLSv1 are supported by the TOE in FIPS mode.
- Key management API and command line interface (CLI): through this API and the CLI, key and certificate generation and management functionality (like validation or deletion of certificates) can be accessed. A product that uses the TOE shall access the keystore through these interfaces only.

The TOE encapsulates ICC for software-based cryptographic functions and key generation.

Certificate validation may be done with the help of OCSP or CRLs. If CRLs are to be used for certificate validation, LDAP or HTTP resources or a flat file containing the revocation list must be provided by the TOE environment. Likewise, for the usage of OCSP, an OCSP responder must be provided by the environment.

Key management tools other than the CLI, e.g., iKeyman, are not part of the TOE.

The classification of the different data types is given in the following:

- **User data**: *User Data is information stored in TOE resources that can be operated upon by users in accordance with the TSP and upon which the TSF places no special meaning*. In this category falls all data that is not listed under TSF data since according to the CC, all data is either user or TSF data.
- **TSF data**: *TSF data is information used by the TSF in making TSP decisions*; for this TOE it is
  - critical security parameters,
    - key data,

        - random number seed,
        - random numbers used for cryptographic operations,
        - passwords
    - configuration parameters,
    - certificates,
    - CRLs,
    - OCSP responses, and
    - the **security attributes** given below for the objects they belong to:
        - for the TOE: the configuration parameters FIPS mode and CC mode,
        - for SSL/TLS connection: ciphersuite used, session keys, mode (client mode or server mode), in server mode: authentication type (whether client authentication is required or not), and whether CRLs are used for certificate verification,
        - for the keystore: the password and the hash value,
        - for hardware crypto modules accessed via PKCS#11: the PIN,
        - for PKCS#12 formatted files: the password,
        - for communication partners: the public key certificate,
        - for certificates: the trust status.

2.1 Product Type

GSKit is a library that can be used as a component within a larger product; it is intended to be integrated into such a product to provide this product with the capability to use an SSL- or TLS-protected communication channel to another product, providing authentication of the communication partners and protection against disclosure or undetected modification of the data transferred over this trusted channel. Furthermore, GSKit offers a command line interface which can be used for accessing key generation and management functions of GSKit, for example by a human user rather than by a program.

For (optional) client and (mandatory) server authentication, X.509 [X.509] certificates are used. These certificates can be imported from the environment. In addition to that, GSKit offers certificate generation and management functionality.

GSKit requires the product it is integrated into and the underlying operating system to provide the protection of the GSKit executables, security attributes and data, especially cryptographic keys and certificates. This includes a correct configuration of the TOE and that the processing resources of the TOE must be located within controlled access facilities.

2.2 Summary of Security Features

The security features provided by GSKit are described in the following paragraphs.

2.2.1 Secure Channel

The TOE offers a secure channel for the confidentiality and integrity protection of data transmitted over that channel.

The secure channel functionality of the TOE is available through the TOE's SSL API. The TOE environment is responsible to limit access to this API to the users and roles in the TOE environment authorized to use those functions.

When demanded by the product that integrates the TOE, the secure channel is established by GSKit using the SSLv3 (as defined in [SSLv3]) or TLSv1 (as defined in [TLSv1] with [TLS_AES]) protocols. Where these standards leave several options, the TOE does not always provide all possible options. These restrictions are described as follows.

Earlier versions of the SSL protocol have been excluded from the TOE since they have known weaknesses. The use of only SSLv3 and TLSv1 is ensured by initializing the TOE in FIPS mode. This has to be done by the TOE environment.

The SSL and TLS connections support mandatory server authentication and optional (configurable as mandatory) client authentication. Total anonymity mode as defined in SSLv3 or TLSv1, meaning that not even the server of the connection has to authenticate, cannot be used in the evaluated configuration.

For authentication, X.509 [X.509] certificates are used; version 1, 2, and 3 certificates are supported for root and end user certificates, and version 3 certificates are supported for intermediate CA certificates.

The following ciphersuites are supported by the TOE (see chapter 2.2.2 for further details on cryptographic algorithms):

a) SSL and TLS:

- CipherSuite SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA = { 0xFE,0xFF }

b) TLS:

- CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0A }
- CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA = { 0x00, 0x2F }
- CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA = { 0x00, 0x35 }

If no common ciphersuite is found, no SSL/TLS connection is set up, i.e. a session with the ciphersuite SSL_NULL_WITH_NULL_NULL is not supported.

For key agreement and authentication, only RSA is supported (i.e. FORTEZZA and Diffie-Hellman are not supported).

The check that a certificate is valid and has not been revoked can be done with the help of CRLs or OCSP. CRLs have to be conformant to X.509 [X.509], version 1 and 2 CRLs are supported. The use of CRLs is not mandatory but recommended in [SSLv3] and [TLSv1]. If CRLs are used, the environment has to provide an LDAP client and LDAP server, or an HTTP server, or the CRL via a flat file available on the underlying system. Current and correct CRLs must be provided by the environment. If CRLs are used and no CRL can be retrieved, the validation check will fail and the certificate's revocation status will be considered undetermined.

As an alternate means to CRL validation, the TOE can use OCSP as defined in [RFC2560] to obtain the revocation status of a certificate. The TOE can be configured to use CRL checking as a fall-back if no valid OCSP response can be obtained from the IT environment.

The use of CRL checking and/or OCSP responses in CC mode is optional.

2.2.2 Cryptographic operations

The TOE offers generation of symmetric keys, generation of asymmetric key pairs, symmetric encryption/decryption, asymmetric encryption/decryption,generation/verification of digital signatures, data authentication, secure message digest algorithms, and random number generation. These cryptographic services are provided by the ICC component and are used by the TOE for SSL/TLS.

The ICC module is FIPS 140-2 level 1 [FIPS140-2] validated. A subset of its cryptographic algorithms used within the TOE are FIPS-Approved and/or NIST-recommended; these algorithms are listed in table 2-1 as FIPS approved algorithms.

The TOE uses only FIPS-Approved and/or NIST-recommended cryptographic algorithms for the cipherspec part of the SSL/TLS ciphersuites and uses non-FIPS approved algorithms for other functions, like key exchange for an SSL/TLS session, when operated in compliance with this Security Target.

The TOE's RSA encryption and decryption, MD5, and MD2 are not FIPS 140-2 [FIPS140-2] approved. RSA encryption/decryption and MD5 are required for key exchange during the setup of TLSv1/SSLv3 sessions. MD5 and MD2 are used for the verification of certificates that have not been generated by the TOE. None of these functions is part of the cipherspecs of the SSL/TLS ciphersuites.

The use of only FIPS-Approved and/or NIST-recommended cryptographic algorithms for the cipherspecs of SSL/TLS ciphersuites (which leads to the supported ciphersuites listed in chapter 2.2.1) is assured by initializing the TOE in FIPS mode. This initialization has to be done by the TOE environment. The TOE then only provides the ciphersuites listed above and initializes its ICC component also in FIPS mode which assures that only the FIPS approved random number generator is used.

The algorithms used by the TOE and whether they are FIPS-Approved and/or NIST-recommended is shown in the following table.

Alternatively to using the cryptographic mechanisms implemented in the TOE, the TOE is able to use cryptographic hardware modules in the TOE environment via PKCS#11 for cryptographic primitives.

Table 2-1: GSKit Cryptographic Algorithms implemented by the TOE

Cryptographic operation	**Algorithms implemented by the TOE**	
	FIPS 140-2 validated	**Non-FIPS 140-2 validated**
Asymmetric key generation		**RSA; 1024 and 2048 Bits** as defined in RFC 2313 [RFC2313]
Symmetric data encryption / decryption	**TDEA with three independent keys (also called Triple DES or 3 key Triple DES)** – **CBC mode** as defined in FIPS 46-3 [FIPS46-3] (TDEA) and FIPS 81 [FIPS81] (CBC mode) **AES – CBC mode** ; 128 and 256 Bits as defined in FIPS 197 [FIPS197]	
Asymmetric data encryption/decryption		**RSA** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313]; ***used by the TOE only for the key exchange of session keys as defined in SSLv3 and TLSv1***
Digital signature generation/ verification	**DSA with SHA-1** as defined in FIPS 186-2 [FIPS186-2] and FIPS 180-2 [FIPS180-2] (SHA-1) for signature verification only. **RSA with SHA-1** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1, SHA-2)	**RSA with MD5** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1321 [RFC1321]; ***only signature verification is used by the TOE (for the validation of certificates that are signed using RSA with MD5)*** **RSA with MD2** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1319 [RFC1319] (MD2); ***only signature verification is used by the TOE (for the validation of certificates that are signed using RSA with MD2)*** **RSA with SHA-224, SHA-256, SHA-384 or SHA-512** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1, SHA-2)
Digest (Hashfunction)	**SHA-1** as defined in FIPS 180-2 [FIPS180-2]	**MD5** as defined in RFC 1321 [RFC1321]
Data authentication	**RSA** as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) **HMAC SHA-1** as defined in FIPS 198 [FIPS198] (HMAC) and FIPS 180-2 [FIPS180-2] (SHA-1)	
Random number generation	**Universal Software Base True Random Number Generator algorithm** as invented by IBM, pseudo-randomizes in accordance with FIPS 186-2 [FIPS186-2]	

2.2.3 Self-tests

GSKit offers self-tests for the ICC component: some of ICC's cryptographic functions and the integrity of ICC can be tested. The self-tests have been analyzed as part of the FIPS 140-2 level 1 [FIPS140-2] validation.

The ICC self-tests are automatically executed upon the first call to the SSL environment initialization command of the TOE. This command (function) must be called by the application that deploys the TOE to initialize the environment before setting up a trusted channel.

2.2.4 Key Management

GSKit provides a local command line interface and an API for key generation and management of asymmetric key pairs as well as generation and management of certificates. For the generation of certificates, certificate requests conformant to PKCS #10 [RFC2986] are used.

The TOE environment is responsible for limiting access to the TOE functions (CLI and API) for key/certificate generation and management to the users and roles in the TOE environment authorized to perform those management functions. Furthermore, users must protect the password of the keystore and PINs for PKCS#11 devices.

GSKit uses a keystore to store key data, or alternatively a PKCS#11 device. The key data stored in the keystore consists of private keys, certificates, certificate request data, and a HMAC-SHA-1 hash of the keystore using the password as the HMAC Key. Stored with each certificate is its trust status, i.e. a security attribute that tells whether a certificate is trusted or not. If a certificate is not trusted, it is neither considered a valid CA nor a valid end user certificate. This security attribute is used for certificate management by the TOE users to validate or invalidate certificates.

GSKit may make use of hardware keystore and acceleration devices in the IT environment providing that the device being used is FIPS 140-2 certified. In CC mode GSKit may only use certificates held on PKCS#11 FIPS 140-2 certified devices that have the CKA_TRUSTED attribute set. If the PKCS#11 Device does not support this Certificate Object Attribute then the GSKit keystore is the only source of trusted certificates allowed. If the crypto module in the IT environment requires a PIN, GSKit can temporarily store the user-supplied PIN in memory during run-time and hand it down to the module when required.

The TOE supports the im- and export of private keys and public key certificates based on PKCS#12-formatted files.

In GSKit's own keystore, each certificate, certificate/private key pair, or request key pairs with PKCS #10 [RFC2986] certificate request data is stored in one so-called data record. Private keys are stored in encrypted form only; certificates and certificate request data are stored unencrypted for faster access. Encryption is done using TDEA (Triple DES) with three independent keys. For integrity protection, a HMAC-SHA-1 hash is generated by hashing all records using the password as the HMAC Key. This hash is also stored in the keystore, in a special record that exists once per keystore file, a so-called header record. In this header record, also the HMAC-SHA-1 hash of the keystore header record using the password as the HMAC Key is stored.

The TOE may be initialized in a mode that affects the behavior of the GSKit-provided keystore:

- Non Common Criteria (non-CC) mode: the keystore has optional password protection. An additional feature in this mode is the use of a "stashfile" which is a file where the password is stored in plain text to allow the unattended operation and restart of a product using GSKit.
- Common Criteria (CC) mode: the keystore is password protected and private keys are encrypted with the TDEA algorithm, using the password to generate the key. The password is checked against a defined password quality policy when entered the first time to ensure that a secure password has been chosen. No stashfile is allowed in this mode of operation.

The evaluated configuration has to initialize the TOE in CC mode; non-CC mode must not be used. This has to be set by the user deploying the TOE.

2.3 Software and Guidance

The Target of Evaluation is based on the following system software:

GSKit

The following documents form the TOE guidance:

- [GSKCAPI]: GSKCapiCmd User's Guide

- [GSKCCMODE]: Global Security Kit Common Criteria Mode Operating Guidance
- [GSKKEY]: IBM Global Security Kit Key Management for C Programmer's Guide
- [GSKINST]: Global Security Kit Install and Packaging Guide
- [GSKSSL]: IBM Global Security Kit – Secure Socket Layer for C Programmer's Guide
- [GSKTRUST]: IBM Global Security Kit Trust Policy Design for GSKit
- [GSKPLI]: Performing GSKit Local Installations
- [DOD]: Global Security Kit Delivery Procedure Additional Guidance

The following ICC documentation from IBM is not part of the TOE guidance:

- [ICC]: IBM Crypto for C Users Guide
- [ICCDESIGN]: IBM Crypto for c (ICC) Design
- [ICCSEC]: IBM Crypto for C (ICC) FIPS 140-2 Non-Proprietary Security Policy

Further referenced documents, such as the standards and RFCs, are not part of the TOE guidance either.

GSKit is only for IBM internal use and is not offered for sale as a standalone product, i.e., it is designed for the use in other IBM products. There is a formal process in place by which GSKit binaries, header files and documentation can be requested within IBM. After approval, permissions to access the TOE software and guidance over an IBM internal secured network connection are given and the TOE can be securely retrieved by the product team.

2.4 Security Environment TOE Boundary

2.4.1 Overview

The following figure provides an overview of the TOE within its intended environment. Dotted lines mark optional components which may be left out. White boxes show components that are part of the environment. The TOE components have been colored dark grey and the box surrounding them shows the TOE boundary. The secure channel set up by the TOE is the box named SSL/TLS. The TOE can use either its internal ICC software crypto module or an external hardware module in the IT environment for cryptographic operations, and in lieu of the TOE keystore again the external hardware module can be used for key and certificate storage.

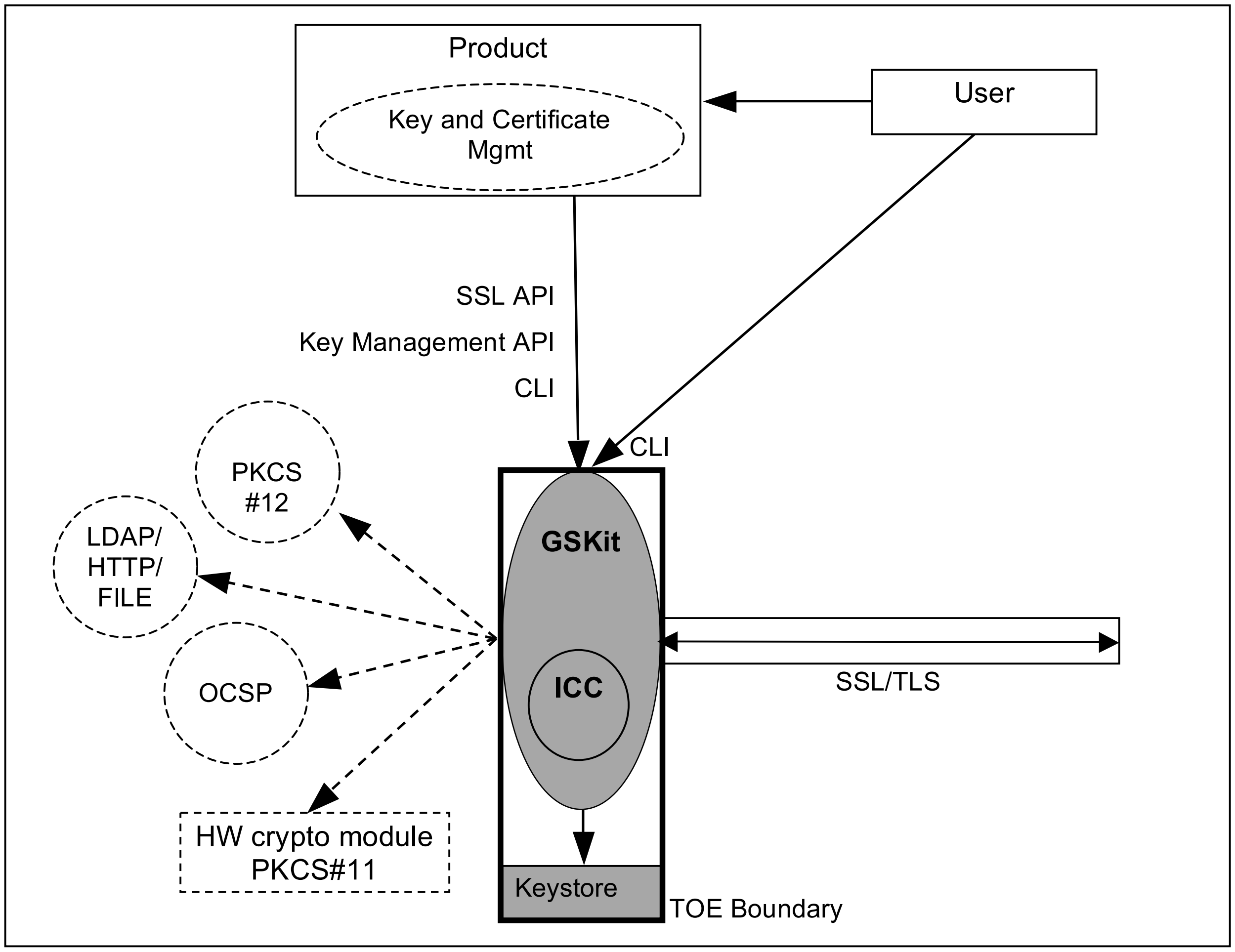

Figure 2: TOE Interfaces and Boundaries

2.4.2 TOE and User Interfaces

GSKit offers three interfaces to the user: the Key Management API and CLI for key and certificate management functions as well as the SSL API. The CLI is the only interface to human users. The APIs and also the CLI are used by IT products (applications) that want to integrate and use the functions of the TOE.

The SSL API is used by an IT product in the TOE environment to request the TOE to setup or to use a SSLv3 or TLSv1 connection. A SSL/TLS connection goes from the TOE to a remote trusted IT product allowing them to exchange data over this trusted channel. The setup of a SSL/TLS connection can be initiated either by the TOE (then the TOE is in client mode and the remote trusted IT product in server mode) or by the remote trusted IT product (then the TOE is in server mode and the remote trusted IT product in client mode).

The Key Management API and the CLI are for key and certificate generation and management.

GSKit implements the LDAP API to access a TOE external LDAP client as an optional part of its certificate management functions (to retrieve a current and correct CRL). The LDAP client used for this purpose is not part of the TOE but an optional part of the TOE environment. The interface to the LDAP client is not offered to the user; the information needed by the LDAP client to connect to the correct LDAP server is specified in the function calls of the SSL API. Likewise, GSKit can issue HTTP requests to web servers to retrieve CRLs or interpret a FILE URI to locate a CRL in the file system of its host system.

The TOE uses interfaces to the underlying operating system to store TSF data in the keystore (files in the operating system's filesystem), retrieve the system time and to access the TCP/IP stack. The TCP/IP stack is accessed using one of two methods controlled by the product using GSKit. In the first method, the application program passes a file descriptor directly to GSKit for use in writing and reading to the TCP/IP stack. In the second method, the application program passes a pointer to callback routines for reading and writing.

2.4.3 Operating Systems and TOE Security Architecture

The operating system is not part of the TOE. It provides identification and authorization for TOE users, and a time source to be used by the TOE for certificate validation. The operating system is also responsible to protect the access to TOE services, the GSKit software, TSF data, and the keystore. As such, it provides significant support for the security of the TSF and TSF data: the TOE itself has only limited ability to protect against the corruption of TSF (i.e. the ICC module implements basic self-checks to satisfy FIPS 140-2 requirements). The underlying operating system is responsible for preventing bypass of the TSF or compromise of the TOE and its data by restricting access to authorized users only.

The TOE is provided in form of multiple binaries that have been compiled for the different operating systems supported by GSKit and the hardware architectures these operating systems run on. The TOE does not interface hardware directly, but receives direct support for its security functions from the operating system. A direct dependency of the TSF on the underlying operating system is for the provision of a reliable time source for certificate validation. Indirect dependencies exist as elaborated above.

The table below identifies the GSKit binaries that are part of the evaluated configuration, and for each of them the operating system versions they have been tested on and the additional operating system versions that the evaluation results can be extended to. For each GSKit binary, the operating system versions listed for a binary in the "Compatibility" column have been found to provide the necessary support functions in the same way as the operating system version that is actually used for testing (column "Test Environment") during the evaluation. Compatibility is claimed in accordance with CCEVS Precedent PD-0084.

TOE binary version	**Test Environment**	**Compatibility**
Sun Solaris 32-bit, running on Sparc and Ultrasparc	Solaris 9 with 111712-08	Solaris 7, 8, 9, 10
Sun Solaris 64-bit, running on Ultrasparc	Solaris 9 with 111712-08	Solaris 7, 8, 9, 10
HPUX 32-bit, running on PA-RISC1 and PA_RISC2 architectures	HPUX 11i with PHSS26946, PHSS_33033	HPUX 11.0, 11i, 11iV2, 11iv3
HPUX 64-bit, running on PA-RISC2 architectures.	HPUX 11i with PHSS26946, PHSS_33033	HPUX 11.0, 11i, 11iV2, 11iv3
IBM AIX 32-bit, running on PPC32 & PPC64 architectures	AIX 5.2 patch level 5200-04 xlC.aix50.rte.6.0.0.3 or later	AIX 4.3.3, 5.1, 5.2, 5.3
IBM AIX 64-bit, running on PPC64 architectures	AIX 5.2 patch level 5200-04 xlC.aix50.rte.6.0.0.3 or later	AIX 5.2, 5.3
Microsoft Windows 32-bit, running on IA32 architectures	Windows Server 2003 SP1	Windows NT, XP, 2000, 2003, Vista, and future version
Microsoft Windows 64-bit, running on AMD x86_64 architectures	Windows Server 2003 SP 1	Windows 2003, Vista
RedHat Enterprise Linux 32-bit, running on IA32 architectures	RHEL 4.0 AS compat-gcc-32-c++-3.2.3-46.1.i386.rpm	UL 1.0 RHEL 2.1 AS, WS; RHEL 3.0 AS, ES, WS; RHEL 4.0 ES; RHEL 5.0

TOE binary version	Test Environment	Compatibility
	compat-gcc-32-3.2.3-46.1.i386.rpm compat-libstdc++-33-3.2.3-46.1.i386.rpm	
RedHat Enterprise Linux 32-bit, running on PPC32 & PPC64 architectures	RHEL 4.0 AS	UL 1.0 RHEL 3.0 AS, ES, WS; RHEL 4.0 AS RHEL 5.0
RedHat Enterprise Linux 32-bit, running on s390/zSeries architectures	RHEL 4.0 AS	UL 1.0 RHEL 3.0 AS; RHEL 4.0 AS RHEL 5.0
RedHat Enterprise Linux 64-bit, running on AMD x86_64 architectures	RHEL 4.0 AS	UL 1.0 RHEL 3.0 WS, AS; RHEL 4.0 WS, AS, ES; RHEL 5.0
RedHat Enterprise Linux 64-bit, running on PPC64 architectures	RHEL 4.0 AS	RHEL 3.0 AS, ES, WS; RHEL 4.0 AS RHEL 5.0
RedHat Enterprise Linux 64-bit, running on zSeries architectures	RHEL 4.0 AS	UL 1.0 RHEL 3.0 AS; RHEL 4.0 AS RHEL 5.0
SUSE Linux Enterprise Server 32-bit, running on IA32 architectures	SLES 9.1	UL 1.0 SLES 8, 9, 10
SUSE Linux Enterprise Server 32-bit, running on PPC32 & PPC64 architectures	SLES 9.1	UL 1.0 SLES 8, 9, 10
SUSE Linux Enterprise Server 32-bit, running on s390/zSeries architectures	SLES 9.0	UL 1.0 SLES 8, 9,10
SUSE Linux Enterprise Server 64-bit, running on AMD x86_64 architectures	SLES 8.0	UL 1.0 SLES 8, 9, 10
SUSE Linux Enterprise Server 64-bit, running on PPC64 architectures	SLES 9.1	SLES 8, 9, 10
SUSE Linux Enterprise Server 64-bit, running on zSeries architectures	SLES 9.0	SLES 8, 9, 10

The underlying operating system has to be configured to disallow remote login (by disabling all services that provide remote login). This is due to the FIPS 140-2 level 1 security policy [ICCSEC]. It is the consumer's responsibility to ensure that the evaluated configuration of GSKit is only executed on operating systems that have been covered by the FIPS 140-2 validation of GSKit's ICC module.

2.4.4 Certificates

Certificates are either provided by the TOE environment or generated by the TOE. Even though the TOE is normally not used for implementing a PKI, using a self-generated certificate for testing or for an application in a closed environment is a needed functionality. Further needed certificate management functionality, especially the generation of CRLs if used for certificate validation or functions needed by the product deploying the TOE, has to be provided by the TOE environment.

2.4.5 iKeyman, and Keystore

With GSKit Version 7d, iKeyman can be used as a GUI for key/certificate generation and management. iKeyman is not part of the TOE; instead the Key Management API and CLI are offered to the user. If iKeyman is to be used, it will be as part of the TOE environment, i.e. of the product that integrates the TOE into itself.

The password-protected keystore can be used by the TOE for the storage of key data. The password used to protect the keystore must be generated by the TOE; user-defined passwords must not be used. The TOE checks the password when it is entered for the first time against a quality metric but that metric does not test the randomness of the password. The password must not be revealed to anyone. In order to ensure the secure operation of the keystore, the TOE has to be configured by the product deploying it to be in CC mode (see chapter 2.2.4 for an explanation of CC mode).

The files used to implement the keystore are located in the operating system. The TOE environment is responsible for ensuring that all access to these files is mediated via the GSKit Key Management API and for protecting these files from unauthorized access.

2.4.6 CRLs

CRLs can be used for certificate validation. If CRLs are used for certificate validation, it is the responsibility of the TOE environment to provide a current and correct CRL.

CRLs can be retrieved using an LDAP client and server, an HTTP server, or a flat file in the TOE environment. For LDAP or HTTP, an LDAP client and server, or an HTTP server, must be provided by the TOE environment. In case of LDAP, the TOE accesses the LDAP client via the LDAP client’s API. Then the LDAP client establishes a TCP/IP connection to the LDAP server, retrieves the CRL, and passes this CRL back to the TOE. In case of HTTP, the TOE issues HTTP requests to the web server. In case of flat files, the TOE retrieves the CRL from the underlying system’s file system. The retrieved CRL is finally used for certificate validation.

The connection between the LDAP client and the LDAP server, or between the TOE and the HTTP server, must be an internal communication link within a trusted network since this is done over an unprotected TCP/IP connection. This is needed in order to avoid DoS attacks or replay attacks.

2.4.7 OCSP

GSKit supports the Online Certificate Status Protocol as specified in RFC 2560 [RFC2560] to query a responder in the IT environment for the revocation status of a certificate. If OCSP is used for certificate validation, it is the responsibility of the TOE environment to provide a trusted OCSP responder.

2.4.8 Crypto Modules

In addition to its own cryptographic software module, ICC, GSKit also provides support for the use of hardware crypto modules (e. g. IBM 4758) via PKCS#11 for the execution of cryptographic primitives and the storage of keys and certificates. Such modules must be validated according to FIPS 140-1 or 140-2 and must be operated in their Approved mode of operation. The cryptographic operations accessed via PKCS#11 are the same as provided by ICC (see section 2.2.2).

GSKit implements PKCS#11 version 2.10 [PKCS#11-2.10] augmented with the AES mechanism definition and support specified in PKCS#11 version 2.20 [PKCS#11-2.20].

It is also possible to use the BSAFE library with GSKit; however this is not allowed for the evaluated configuration. Furthermore, the use of the Microsoft Crypto API is prohibited in the evaluated configuration.

3 TOE Security Environment

3.1 Introduction

The statement of TOE security environment 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 deployed.

To this end, the statement of TOE security environment identifies the list of assumptions made on the operational environment (including physical and procedural measures) and the intended method of use of the product, defines the threats that the product is designed to counter, and gives the organizational security policies with which the product is designed to comply.

3.2 Threats

The assumed security threats are listed below.

The **IT assets** to be protected comprise the information stored, processed or generated by the TOE. This information contains key data, data for cryptographic operations (like for example seeds for random data needed for key generation) as well as the data to be transferred over the trusted channel.

The purpose of the TOE is to provide a secure channel between itself and a remote trusted IT product. This secure channel is used by a product (application) of the TOE environment. Threats concerning information confidentiality and integrity of the TOE code and TSF data are mainly countered by the TOE environment; the TOE only has to provide a correct implementation of the secure channel such that the data transmitted over it is protected and to secure the data that is to be stored in the keystore.

The **threat agents** can be categorized as either:

1. A user (not a user of the TOE) who gets access to data communicated over unprotected networks ("remote attacker"). This person has no access to the TOE interfaces and no access to the system the TOE is integrated into.
2. A user (not a user of the TOE) who gets access to the TOE code, TSF data, or the keystore (e. g. on a backup tape or by getting access to the disk where it is stored on).
3. An authorized user of the TOE environment who has been granted the right to access the TOE using one or many of the interfaces (SSL API, command line interface, Key Management API).

In the first and second cases, the users are assumed to be potentially hostile with a clear motivation to get access to the data. The threat TE.BYPASS deals with threat agents of category 1, the threat TE.ACCESS with threat agents of category 2.

The authorized user is a user allowed to use the TOE. Authorized users (i.e. human users or products deploying the TOE) are assumed to be trustworthy and they will not attempt to subvert or bypass the TOE security functions. An administrator of the TOE or TOE environment is also included in this category. An administrator is assumed to be competent and trustworthy. Authorized users do not pose a threat.

3.2.1 Threats countered by the TOE

Due to the nature of the TOE, the security functionality is derived from organizational security policies the TOE enforces. Therefore no threats to be countered by the TOE are listed.

3.2.2 Threats to be countered by measures within the TOE environment

The following threats to the system need to be countered in the TOE environment:

TE.ACCESS	TOE functions, TOE code, and security attributes may be accessible for unauthorized modification and user data or TSF data may be accessible for unauthorized disclosure, access, and modification (for example modification of the TOE or cryptographic data) by users that have gained local access to the system the TOE is integrated to or to backup data.

TE.BYPASS	CRLs may be transmitted from LDAP or HTTP servers to the TOE over unprotected communication channels such that they may be modified or deleted by a remote attacker. No trusted channel can be set up if no CRL can be retrieved or the retrieved CRL is corrupted (i.e. not current or broken). If with a replay attack a technically still valid CRL that does not contain all revoked certificate information is resent, a trusted channel might be established for a revoked certificate.

3.3 Organizational Security Policies

The TOE complies with the following organizational security policies:

P.MODE	FIPS mode and CC mode must be set for the TOE. This also ensures that the TOE shall be operated in a configuration compliant to the FIPS validated configuration of the ICC component described in the FIPS 140-2 level 1 security policy for ICC [ICCSEC].
P.KEYSTORE	Private keys must be protected from unauthorized disclosure when they are stored in the keystore. In addition to that, a mechanism against undetected modification of the whole of the key data stored in the keystore must be provided.
P.SECCHANNEL	The TOE shall offer a product of the TOE environment the functionality to communicate with a remote trusted IT product using a cryptographically secured trusted channel that protects the confidentiality and integrity of data being transmitted over this channel and provides the ability to authenticate the communication partners before transmitting user data that requires protection. User data and secret TSF data of a session must be protected such that they are not available to other sessions.
P.SELFSEC	The TOE shall implement self-tests for cryptographic modules and failure detection with preservation of a secure state for the trusted channel component
P.TRANSFER	The TOE shall be able to exchange TSF data with entities in the TOE environment.

3.4 Assumptions

This section indicates the minimum physical and procedural measures required to maintain the security of the TOE.

A.ADMIN	Those responsible for the administration of the TOE and the TOE environment are competent and trustworthy individuals, capable of managing the TOE and the TOE environment and the security of the information it contains.
A.CERTIFICATES	The TOE environment is responsible for the management and generation of the certificates.
A.INTEGRATE	Those who integrate the TOE as part of a larger product or system are assumed to follow the guidance provided with the TOE with respect to the configuration of the TOE, the use of TOE functions and interfaces and the protection of the TOE code and data. They are trustworthy and do not try to subvert or bypass TOE security functions.
A.LDAP	If a CRL is required by the TOE, the TOE environment must provide a valid, current CRL.
A.OCSP	OCSP responders used by the TOE are trustworthy.
A.PASSWORD	Users knowing the password to protect the keystore will apply high protection measures that prohibit unauthorized access to or disclosure of password information.
A.TIMESTAMP	The environment will provide reliable timestamps.

4 Security Objectives

This section defines the security objectives of the TOE and its operational and development environment. Security objectives for the operational environment are either categorized as IT security objectives or as non-IT security objectives.

The intent of the security objectives is to counter all identified threats and to comply with the given assumptions and organizational security policies.

4.1 Security Objectives for the TOE

O.KEYSTORE	The TOE must protect private keys stored in the keystore from unauthorized disclosure. In addition to that, the whole of the data stored in the keystore must be protected against undetected modification.
O.SECCHANNEL	The TOE shall offer a product in the TOE environment the functionality to communicate with a remote trusted IT product using a trusted channel that protects the confidentiality and integrity of user data being transmitted over this channel and provides the ability to authenticate the communication partners before transmitting user data that requires protection.
O.SELFSEC	The TOE must implement self-tests for cryptographic modules and failure detection with preservation of a secure state for the trusted channel component.
O.SESSIONDATA	The TOE must protect secret TSF data (private/secret keys, session keys, and critical security parameters) of its individual trusted channel sessions and must ensure that no such data is made available to other trusted channel sessions also provided by the TOE.
O.TRANSFER	The TOE must be able to consistently interpret TSF data that is exchanged with entities in the IT environment via PKCS#11, PKCS#12.

4.2 Security Objectives for the TOE Environment

4.2.1 TOE Operational Environment

4.2.1.1 IT Security Objectives

OE.CRYPTO	If the TOE is configured to use a hardware crypto module in the IT environment for the storage of cryptographic keys and certificates and/or the provision of cryptographic operations, the environment must provide such a hardware module accessible via PKCS#11. The hardware module must authenticate GSKit before granting access to stored key material.
OE.DATA	The TOE environment (e.g. the operating system) must protect the TOE code from unauthorized modification and user data and TSF data from unauthorized disclosure, access, and modification. This includes also the protection of the TOE against other applications in the TOE environment.
OE.MODE	The TOE must be initialized to operate in FIPS mode and CC mode.
OE.OSLOGIN	Remote login to the operating system must be disabled (by disabling all services that offer remote login).
OE.TIMESTAMP	The environment must provide reliable timestamps.
OE.USER	The environment must only allow authorized users to access the TOE.

4.2.1.2 Non-IT Security Objectives

OE.ADMIN	Those responsible for the administration of the TOE must be trained such that they are capable of managing the TOE and the security of the information it contains; this includes making sure that any backup copies of TSF data are appropriately protected. Administrators are trustworthy.
OE.CERTIFICATES	The TOE environment is responsible for the management and generation of the certificates.

OE.FIPS140	If hardware crypto modules are employed in the IT environment, they must be FIPS 140-1 or 140-2 validated and operated in their Approved mode of operation.
OE.INTEGRATE	Those who integrate the TOE as part of a larger product or system are following the guidance provided with the TOE with respect to the configuration of the TOE, the use of TOE functions and interfaces and the protection of the TOE code and data. They are trustworthy and do not try to subvert or bypass TOE security functions.
OE.LDAP	If a CRL is required by the TOE, the TOE environment must provide a valid, current CRL. If CRLs are provided via LDAP client and server or HTTP server, then the TCP/IP connection to the server must be an internal communication link within a protected network. CRLs provided are valid, current CRLs conformant to [X.509].
OE.OCSP	If OCSP is to be used, those responsible for the TOE operations must ensure that only trustworthy OCSP responders are used to provide certificate status to the TOE.
OE.OS	On the server the TOE is running, only operating systems that are covered by the FIPS 140-2 level 1 certificate for the ICC component may be used.
OE.PASSWORD	Users knowing the password to protect the keystore must apply protection measures that prohibit unauthorized access to or disclosure of password information.
OE.PHYSEC	The processing resources of the TOE must be located within controlled access facilities such that they are protected against unauthorized physical access.

5 Security Requirements

5.1 TOE Security Functional Requirements

This section gives the security functional requirements (SFRs) of the TOE. The SFRs FPT_TST_WEAK.1 and FPT_ETT_GSK.1 have been defined in the style of [CC] part 2 and the SFR FTP_ITC.1 has been replaced by an explicitly stated SFR, FTP_ITC_PD-0108.1, as described in [PD-0108]. All other SFRs are taken from [CC] part 2.

The following formatting styles have been used for the SFRs:

- **Assignments and selections**: bold text
- **Refinements**: bold and italic text
- **Explicit SFRs**: italic text; to identify assignments and selections in explicit SFRs, bold and italic text is used; to identify refinements, bold and non-italic text is used.

Table 5-1: Security Functional Requirements of the TOE

Security Functional Class	Security Functional Components
Cryptographic Support (FCS)	FCS_CKM.1: Cryptographic key generation FCS_COP.1: Cryptographic operation
User Data Protection (FDP)	FDP_ACC.1: Subset access control FDP_ACF.1: Security attribute based access control FDP_DAU.1: Basic data authentication FDP_RIP.2: Full residual information protection FDP_UCT.1: Basic data exchange confidentiality FDP_UIT.1: Data exchange integrity
Identification and Authentication (FIA)	FIA_SOS.1: Verification of secrets FIA_SOS.2: TSF Generation of secrets FIA_UAU.2: User authentication before any action
Security Management (FMT)	FMT_MSA.3: Static attribute initialization
Protection of the TSF (FPT)	*FPT_ETT_GSK.1 Basic external TSF data transfer protection* FPT_FLS.1: Failure with preservation of secure state FPT_TDC.1: Inter-TSF basic TSF data consistency *FPT_TST_WEAK.1: Partial TSF testing*
Trusted Path/Channels (FTP)	*FTP_ITC_PD-0108.1: Inter-TSF trusted channel*

Three explicit security functional requirements are defined in this Security Target:

FPT_ETT_GSK.1 Basic external TSF data transfer protection

FPT_TST_WEAK.1 Partial TSF testing

FTP_ITC_PD-0108.1 Inter-TSF trusted channel

The following section provides the definitions. The rationale for those explicit security functional requirements is given in chapter 7.2.1.

FPT_ETT_GSK.1 Basic external TSF data self-protection

FPT_ETT_GSK.1 Basic external TSF data self-protection

Hierarchical to: No other components.

FPT_ETT_GSK.1.1 The TSF shall protect [assignment: *list of TSF data*] from [selection: *disclosure, undetected modification*] when it is transmitted outside of the TSC.

FPT_ETT_GSK.1.2 The TSF shall use [assignment: *list of standards that are applicable*].

Dependencies: No dependencies

There are no actions identified that should be auditable if FAU_GEN Security audit data generation is included in the PP/ST.

The following actions could be considered for the management functions in FMT:

a) The management of the list of types of TSF data that must be protected when it is transmitted outside of the TSF;
b) management of the types of modification against which the TSF should protect;
c) management of the mechanism used to provide the protection of the data transmitted outside of the TSC.

FPT_TST_WEAK.1 Partial TSF testing

FPT_TST_WEAK.1 Partial TSF testing

Hierarchical to: No other components.

FPT_TST_WEAK.1.1 The TSF shall run a suite of self tests [selection: *during initial start-up, periodically during normal operation, at the request of the authorised user, at the conditions* [assignment: *conditions under which self test should occur*]] to demonstrate the correct operation of [selection: [assignment: *parts of TSF*], the TSF].

FPT_TST_WEAK.1.2 The TSF shall provide authorised users with the capability to verify the integrity of [selection: [assignment: *parts of the TSF data*], *the TSF data*].

Dependencies: FPT_AMT.1 Abstract machine testing

The following actions should be audited if FAU_GEN Security audit data generation is included in the PP/ST:

a) Basic: Execution of the TSF self tests and the results of the tests.

The following actions could be considered for the management functions in FMT:

a) Management of the conditions under which TSF self testing occurs, such as during initial start-up, regular interval, or under specified conditions;
b) management of the time interval if appropriate.

FTP_ITC_PD-0108.1 Inter-TSF trusted channel

FTP_ITC_PD-0108.1 Inter-TSF trusted channel

Hierarchical to: No other components.

FTP_ITC_PD-0108.1.1 The TSF shall provide a communication channel between itself and a remote trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC_PD-0108.1.2 The TSF shall permit [selection: *the TSF, the remote trusted IT product*] to initiate communication via the trusted channel.

FTP_ITC_PD-0108.1.3 The TSF shall use a trusted channel for the following functions [assignment: *list of functions for which a trusted channel is required*].

Dependencies: No dependencies

The following actions should be audited if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: Failure of the trusted channel functions.

b) Minimal: Identification of the initiator and target of failed trusted channel functions.

c) Basic: All attempted uses of the trusted channel functions.

d) Basic: Identification of the initiator and target of all trusted channel functions.

The following actions could be considered for the management functions in FMT:

a) Configuring the actions that require trusted channel, if supported.

Note: Use of the explicitly specified requirement replacement does not imply a responsibility for the TSF under evaluation to ensure that a remote TSF performs a particular action; rather, the TSF under evaluation is only required to use the channel for the indicated purpose if the channel is initiated. There is no requirement for the evaluator to verify that the remote TSF initiates the channel.

Note: When this modified version of FTP_ITC.1.3 is used, there should also be an accompanying note that explains that the rationale for this explicit requirement is that it corrects an error identified by CCEVS in the requirement and an interpretation is being created by NIAP to correct the offending wording.

5.1.1 Cryptographic Support (FCS)

5.1.1.1 Cryptographic key generation (FCS_CKM.1(a))

FCS_CKM.1.1 The TSF shall generate ***symmetric*** cryptographic keys in accordance with a specified cryptographic key generation algorithm

a) SSLv3 symmetric key and secret generation,

b) TLSv1 symmetric key and secret generation

and specified cryptographic key sizes

a) 168 Bits (three independent TDEA keys),

b) 128 and 256 Bits (AES key),

c) 160 Bits (HMAC SHA-1 secret)

that meet the following:

a) conformant to SSLv3 [SSLv3] (SSLv3 symmetric key and secret generation),

b) conformant to RFC 2246 [TLSv1] with RFC 3268 [TLS_AES] (TLSv1 symmetric key and secret generation).

5.1.1.2 Cryptographic key generation (FCS_CKM.1(b))

FCS_CKM.1.1 The TSF shall generate ***asymmetric*** cryptographic keys in accordance with a specified cryptographic key generation algorithm **RSA** and specified cryptographic key sizes **1024 and 2048 Bits** that meet the following:

conformant to RFC 2313 [RFC2313].

5.1.1.3 Cryptographic operation (FCS_COP.1 (a))

FCS_COP.1.1 The TSF shall perform **symmetric encryption and symmetric decryption** in accordance with a specified cryptographic algorithm

a) TDEA with three independent keys (CBC mode) ,

b) AES (CBC mode)

and cryptographic key sizes

a) 168 Bits (TDEA),

b) 128, 256 Bits (AES)

that meet the following:

a) conformant to FIPS 46-3 [FIPS46-3] (TDEA), conformant to FIPS 81 [FIPS81] (CBC mode),

b) conformant to FIPS 197 [FIPS197] (AES, CBC mode),

and FIPS 140-2 [FIPS140-2] approved.

5.1.1.4 Cryptographic operation (FCS_COP.1(b))

FCS_COP.1.1 The TSF shall perform **digest generation and verification** in accordance with a specified cryptographic algorithm

a) SHA-1

b) MD5

and cryptographic key sizes **none** that meet the following:

a) conformant to the Secure Hash Standard (SHS) as defined in FIPS 180-2 [FIPS180-2] and FIPS 140-2 [FIPS140-2] approved (SHA-1),

b) conformant to RFC 1321 [RFC1321] (MD5).

5.1.1.5 Cryptographic operation (FCS_COP.1(c))

FCS_COP.1.1 The TSF shall perform **data authentication** in accordance with a specified cryptographic algorithm **HMAC SHA-1** and cryptographic key sizes **160 Bits (HMAC SHA-1)** that meet the following

conformant to FIPS 198 [FIPS198] (HMAC) and FIPS 180-2 [FIPS180-2] (SHA-1).

5.1.1.6 Cryptographic operation (FCS_COP.1(d))

FCS_COP.1.1 The TSF shall perform **encryption and decryption of session key related data** in accordance with a specified cryptographic algorithm **RSA** and cryptographic key sizes **1024 or 2048 Bits** that meet the following:

conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and encryption/decryption of session key related data as defined in The SSL Protocol, Version 3 [SSLv3] and RFC 2246 [TLSv1].

5.1.1.7 Cryptographic operation (FCS_COP.1(e))

FCS_COP.1.1 The TSF shall perform **generation of random numbers** in accordance with a specified cryptographic algorithm **Universal Software Base True Random Number Generator algorithm** and cryptographic key sizes **none** that meet the following:

the requirements in FIPS 186-2 [FIPS186-2], Appendix 3.2 as required in FIPS 140-2 annex C [FIPS140-2] and FIPS 140-2 level 1 [FIPS140-2] approved.

Application note: The Universal Software Base True Random Number Generator (Universal Base TRNG) algorithm has been invented by IBM. It is patented in Europe [EPRAND] and the patent is pending in the US.

Application note: The random number generator uses timing jitter as entropy source.

5.1.1.8 Cryptographic operation (FCS_COP.1(f))

FCS_COP.1.1 The TSF shall perform **digital signature generation** in accordance with a specified cryptographic algorithm **RSA with SHA-1 message digest** and cryptographic key sizes **1024 or 2048 Bits (RSA)** that meet the following:

conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1).

Application note: The TOE provides a function to generate digital signatures that may be used by the application that integrates the TOE. The TOE environment is responsible to ensure that the user data signed is correct and is intended to be signed by the TOE. The TOE is neither able to verify the correctness of the user data provided for signing. This function has to be performed by the TOE environment.

5.1.1.9 Cryptographic operation (FCS_COP.1(g))

FCS_COP.1.1 The TSF shall perform **digital signature verification** in accordance with a specified cryptographic algorithm

a) RSA with MD2 message digest,

b) RSA with MD5 message digest,

c) RSA with SHA-224, SHA-256, SHA-384 or SHA-512 message digest,

d) RSA with SHA-1 message digest,

e) DSA with SHA-1 message digest,

and cryptographic key sizes

a) 1024 or 2048 or 4096 Bits (RSA),

b) 1024 or 2048 or 4096 Bits (RSA),

c) 1024 or 2048 or 4096 Bits (RSA),

d) 1024 or 2048 or 4096 Bits (RSA),

e) 1024 Bits (DSA),

that meet the following:

a) conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1319 [RFC1319] (MD2),

b) conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1321 [RFC1321] (MD5),

c) conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-224, SHA-256, SHA-384, SHA-512),

d) conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1),

e) conformant to FIPS 186-2 [FIPS186-2] (DSA) and FIPS 180-2 [FIPS180-2] (SHA-1).

Application note: Digital signature verification using RSA with MD2/MD5, RSA with SHA-2 and DSA with SHA-1is used by the TOE to verify X.509 [X.509] conformant certificates. The TOE itself does not generate any certificates using DSA, MD2, MD5 or SHA-2 but needs to be able to verify such certificates.

5.1.2 User Data Protection (FDP)

5.1.2.1 FDP_ACC.1 Subset access control

FDP_ACC.1.1 The TSF shall enforce the **SSL/TLS connection policy** on **external entities as subjects, SSLv3 (as defined in [SSLv3]) /TLSv1 (as defined in [TLSv1] with [TLS_AES]) session as object and session establishment as the operation**.

Application note: The SSL/TLS connection policy enforced by the TOE handles both cases where the TOE operates as SSL/TLS client and server. The TOE will “allow access”, i. e. set up a connection, when the authentication steps required by the protocol have been successfully performed and the TOE has identified a common cipher suite with the communication partner. It is not part of the SSL/TLS connection policy enforced by the TOE to decide when it is going to setup a SSL/TLS connection and when it requires the client to authenticate itself using a valid certificate. This is defined by the product that uses the TOE in accordance to the policies it is going to enforce. The TOE (when operating in server mode) will enforce client authentication if this is requested by the product using the TOE.

5.1.2.2 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the **SSL/TLS connection policy** to objects based on the following:

- **the public key certificate (as defined in ITU-T Recommendation X.509 [X.509]) as security attribute associated with the communication partner as subject,**
- **on SSL/TLS connections as objects with the associated security attributes of ciphersuite used, mode (client mode or server mode), in server mode: authentication type (whether client authentication is required or not), and whether a CRL and/or OCSP responses are used for certificate validation**.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

a) **if operated in client mode, the session is established only if the certificate presented by the communication partner can be validated by the TOE and when the communication partner proves that it has the associated private key**.

b) **If operated in server mode and the TOE is not configured for client authentication, no rule applies.**

c) **If operated in server mode and the TOE is configured for client authentication, the session is established if the certificate presented by the communication partner can be validated by the TOE and when the communication partner proves that it has the associated private key. If the client presents a certificate that can not be validated by the TOE or that has expired, the connection is refused. If the client does not present any certificate an indicator is set that allows the application using the TOE to decide if the connection is established.**

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **none**.

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

a) **The TOE will not establish a SSL or TLS session if the communication partner does not support a cipher suite provided by the TOE. The following ciphersuites are provided as defined in the following standards: [SSLv3], [TLSv1] and [TLS_AES]:**

- **For TLS v1.0 (RFC 2246 Appendix A.5)**
  - **CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0A };**
- **For TLS v1.0 (RFC 3268 the AES):**
  - **CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA = { 0x00, 0x2F };**
  - **CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA = { 0x00, 0x35 };**
- **For the SSL v3.0 and TLS v1.0**
  - **CipherSuite SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA = {0xFE,0xFF };**

b) **If OCSP usage has been configured for certificate validation:**
if the TOE is in client mode or
if the TOE is in server mode and configured for client authentication,
the connection is denied if the OCSP response indicates that the certificate has been revoked.

c) **If CRL usage has been configured for certificate validation, and OCSP has not been configured for usage or the revocation status is undetermined after using OCSP:**
if the TOE is in client mode or
if the TOE is in server mode and configured for client authentication,
the connection is denied if the CRL indicates that the certificate has been revoked.

Application note: The rules for the communication partner connecting to the channel set up by the TOE are not laid out in FDP_ACF.1.2 since it is not within the scope of the TOE to care about the connection policy of the remote program. Nevertheless, the default rules can easily be derived from FDP_ACF.1.2 by choosing the alternate role (SSL/TLS client or server) for the communication partner.

Application note: Both the SSLv3 and the TLSv1 standard do not require that the connection establishment is terminated in the case where a server is configured for client authentication and the client does not present any certificate. This allows applications to handle this case individually e. g. by presenting an error message to the client or offering the client a reduced set of services. The TOE allows the application to check if the client has been successfully authenticated and then lets the application decide whether to shut down the connection, inform the client about the required authentication or offer the client a reduced set of services. Quite a number of web based applications want to adapt their services depending on the fact if the client has been authenticated or not.

Application note: It is possible but not mandatory to use OCSP and/or CRLs for certificate validation. Please refer to chpt. 6 for a detailed discussion of the path validation mechanism.

5.1.2.3 Basic data authentication (FDP_DAU.1)

FDP_DAU.1.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **the X.509 [X.509] conformant public key certificate**.

FDP_DAU.1.2 The TSF shall provide **all subjects** with the ability to verify evidence of the validity of the indicated information.

Application note: FDP_DAU.1.1 provides evidence rather for the authenticity of the public key certificate than for its validity. This is the actual certificate generation by signing an X.509-formatted certificate.

5.1.2.4 Full residual information protection (FDP_RIP.2 (a))

FDP_RIP.2.1 The TSF shall ensure that ***the state information*** of ***the random number generator*** is made unavailable upon the **deallocation of the *random number generator* from** all objects.

Application note: The refinement states that the resource this SFR applies to is the random number generator. Since the random number generator’s state is considered security critical, it will be cleared when it is removed from memory.

5.1.2.5 Full residual information protection (FDP_RIP.2(b))

FDP_RIP.2.1 The TSF shall ensure that any previous ***sensitive*** information content of ***the buffer space memory*** is made unavailable upon the **deallocation of the *buffer space memory* from** all objects.

Application note: The refinement states that the resources this SFR applies to are sensitive information in the buffer space memory. The TOE considers all cleartext critical security parameters, random numbers and cryptographic keys as such sensitive information.

5.1.2.6 Basic data exchange confidentiality (FDP_UCT.1)

FDP_UCT.1.1 The TSF shall enforce the **SSL/TLS connection policy** to be able to **transmit and receive** objects in a manner protected from unauthorized disclosure.

5.1.2.7 Data exchange integrity (FDP_UIT.1)

FDP_UIT.1.1 The TSF shall enforce the **SSL/TLS connection policy** to be able to **transmit and receive** user data in a manner protected from **modification, insertion and replay** errors.

FDP_UIT.1.2 The TSF shall be able to determine on receipt of user data, whether **modification, insertion or replay** has occurred.

5.1.3 Identification and Authentication

5.1.3.1 Verification of secrets (FIA_SOS.1)

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets ***used to protect the keystore*** meet the following metric:

- **The minimum password length is 14 characters**
- **A password needs to have at least one lower case character, one upper case character, and one digit or special character.**
- **Each character should not occur more than three times in a password.**
- **No more than two consecutive characters of the password should be identical.**

5.1.3.2 TSF Generation of secrets (FIA_SOS.2)

FIA_SOS.2.1 The TSF shall provide a mechanism to generate secrets that meet

the following metric:

- **the password length is at least 14 characters**
- **the characters are chosen from the set of upper case alpha, lower case alpha, numeric and punctuation characters i.e. 95 characters of the 7-bit ASCII set,**
- **the password is cryptographically random, i.e. generated with the FIPS 140-2 [FIPS140-2] approved Universal Software Base True Random Number Generator random number generator**.

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

no TSF functions.

Application note: No TSF functions are listed in FIA_SOS.2.2 since the TOE does not enforce the use of TOE generated passwords but the user is advised to use only such passwords. To help avoiding user errors, the entered password is checked against the quality metric (which is of lower strength) given in FIA_SOS.1.

5.1.3.3 User authentication before any action (FIA_UAU.2)

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing ***any TSF-mediated access to the keystore*** on behalf of that user.

Application note: The authentication is not combined with identification since no identity is associated with the password used to protect the keystore. For further information on the password see FPT_ETT_GSK.1(a), FPT_ETT_GSK.1(b), FIA_SOS.2, and FIA_SOS.1.

5.1.4 Security Management (FMT)

5.1.4.1 Static attribute initialization (FMT_MSA.3)

FMT_MSA.3.1 The TSF shall enforce the **SSL/TLS connection policy** to provide **permissive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **Crypto-Officer** to specify alternative initial values to override the default values when an object or information is created.

Application note: The TOE will allow a connection when the rules of the SSL/TLS connection policy are satisfied. It is up to the policy enforced by the TOE environment to define additional restrictions. The TOE environment is also responsible to enforce those additional restrictions.

5.1.5 Protection of the TSF (FPT)

5.1.5.1 Basic external TSF data self-protection (FPT_ETT_GSK.1 (a))

FPT_ETT_GSK.1.1 *The TSF shall protect **a private key** from **disclosure and undetected modification** when it is transmitted outside of the TSC.*

FPT_ETT_GSK.1.2 *The TSF shall use **the password based encryption scheme defined in PKCS #12 [PKCS#12] (SHA-1 and 3-key TDEA CBC mode)**.*

Application Note: This SFR applies to the private key when stored in the keystore, which consists of files in the operating system, protecting the private key from disclosure by other users who do not have the password. It should be noted that when the encrypted private key is stored in memory, the information necessary to decrypt it is also in memory and available for the application to obtain the plain text private key. The clearance of the buffer space memory is described in FDP_RIP.2 (b).

5.1.5.2 Basic external TSF data self-protection (FPT_ETT_GSK.1 (b))

FPT_ETT_GSK.1.1 *The TSF shall protect **a certificate with the associated trust status and certificate request data** from **undetected modification** when it is transmitted outside of the TSC.*

FPT_ETT_GSK.1.2 *The TSF shall use **password based HMAC SHA-1 RFC 2104 [RFC2104] (SHA-1).***

5.1.5.3 FPT_FLS.1 Failure with preservation of secure state

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

a) **SSL / TLS handshake failure,**

b) **failing to correctly decrypt SSL / TLS messages received from the communication partner,**

c) **data integrity failure for SSL / TLS messages received from the communication partner,**

d) **SSL/TLS protocol detected buffer sequence errors,**

e) **integrity errors detected when reading keys from the key file,**

f) **internal failure intercepted by the TOE,**

g) **failure of the self tests**.

5.1.5.4 Inter-TSF basic TSF data consistency (FPT_TDC.1 (a))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret

a. **public key certificates,**

b. **and certificate requests**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use

a. **the standards X.509 as defined in the ITU-T Recommendation X.509 [X.509] and RFC 3280 [RFC3280];**
 certificates of the versions 1, 2, and 3 are supported (public key certificates),

b. **the standard PKCS #10 as defined in [RFC2986] (certificate requests)**

when interpreting the TSF data from another trusted IT product.

Application Note: This SFR applies to certificates managed by the key management API.

5.1.5.5 Inter-TSF basic TSF data consistency (FPT_TDC.1 (b))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret

a. **public key certificates**

b. **certificate revocation lists (CRLs),**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use

a. **the standards X.509 as defined in the ITU-T Recommendation X.509 [X.509] and RFC 3280 [RFC3280];**
certificates of the versions 1, 2, and 3 are supported (public key certificates),

b. **the standards X.509 as defined in the ITU-T Recommendation X.509 [X.509] and RFC 3280 [RFC3280] following ITU-T Recommendation X.509 [X.509] for the handling of unknown critical extensions in a CRL;**
CRLs of the versions 1 and 2 are supported (CRLs),

when interpreting the TSF data from another trusted IT product.

Application Note: This SFR applies to certificates used during secure SSL/TLS sessions.

Application Note: Please refer to chpt. 6 for a discussion on the specifics of certificate path validation with optional CRLs.

5.1.5.6 Inter-TSF basic TSF data consistency (FPT_TDC.1 (c))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret

a. **public key certificates**

b. **private keys**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use **PKCS#12 [PKCS#12]** when interpreting the TSF data from another trusted IT product.

Application Note: This SFR applies to import and export of keys and certificates from/to PKCS#12-formatted files.

5.1.5.7 Inter-TSF basic TSF data consistency (FPT_TDC.1 (d))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret

a. **the revocation status of a certificate**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use **the Online Certificate Status Protocol [RFC2560]** when interpreting the TSF data from another trusted IT product.

5.1.5.8 Inter-TSF basic TSF data consistency (FPT_TDC.1 (e))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret

a. **public key certificates**

b. **private keys**

c. **security attributes (such as the trust status for certificates)**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use **PKCS#11 v2.10 [PKCS#11-2.10] augmented by the AES parts of PKCS#11 v2.20 [PKCS#11-2.20]** when interpreting the TSF data from another trusted IT product.

5.1.5.9 Partial TSF testing (FPT_TST_WEAK.1)

FPT_TST_WEAK.1.1 *The TSF shall run a suite of self tests* ***upon the first call to the SSL environment initialization command of the TOE*** *to demonstrate the correct operation of* ***the TOE ICC component****.*

FPT_TST_WEAK.1.2	*The TSF shall provide authorised users with the capability to verify the integrity of* ***the TOE ICC component****.*

5.1.6 Trusted Path/Channels (FTP)

5.1.6.1 Inter-TSF trusted channel (FTP_ITC_PD-0108.1)

FTP_ITC_PD-0108.1.1	*The TSF shall provide a communication channel between itself and a remote trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.*
FTP_ITC_PD-0108.1.2	*The TSF shall permit* ***the TSF*** **(when operated in client mode)** ***or the remote IT product*** **(when operated in server mode)** *to initiate communication via the trusted channel.*
FTP_ITC_PD-0108.1.3	*The TSF shall use a trusted channel for the following functions:* ***data transfer****.*
Application Note:	*The rationale for this explicit requirement is that it corrects an error identified by CCEVS in the requirement and an interpretation is being created by NIAP to correct the offending wording.*

5.2 Security Functional Requirements for the Operational Environment

The SFRs for the TOE environment have to be satisfied by the TOE environment. They have all been taken from CC [CC] part 2. Where appropriate, the SFRs have been rephrased to clearly indicate that the TOE environment (not the TOE) must meet the requirements. This is allowed and described as “a special kind of refinement and not subject to the assessment requirements associated with modified CC components” in the final interpretation 58 for CC 2.1 [RI 58].

The security functional requirement for the IT environment to a large part address the operating system the evaluated version of GSKit is installed. In addition, FMT_MSA.1 requires to restrict the access to some GSKit functions and objects to the Crypto Officer. This needs to be done by the application using GSKit, which in turn can use the access control functions of the underlying operating system to implement this. FMT_MSA.2 defines the requirement to initialize GSKit in FIPS 140 and CC mode. This requirement also needs to be addressed by the application that initializes and uses GSKit and is therefore not targeted at the operating system as part of the TOE environment. FTA_TSE.1 is derived from the Security Policy of ICC used for the FIPS 140-2 validation.

The following formatting styles have been used for the SFRs:

- **Assignments and selections**: bold text,
- **Refinements**: bold and italic text,
- **Special kind of refinements (SFR rephrasing, see above)**: italic text.

Table 5-2: Security Functional Requirements of the TOE Operational Environment

Security Functional Class	Security Functional Components
Cryptographic Support (FCS)	FCS_CKM.1 Cryptographic key generation FCS_COP.1 Cryptographic operation
User Data Protection (FDP)	FDP_ACC.1: Subset access control FDP_ACF.1: Security attribute based access control FDP_RIP.1:Subset residual information protection
Identification and Authentication (FIA)	FIA_UAU.1: Timing of authentication FIA_UID.1: Timing of identification
Security Management (FMT)	FMT_MSA.1: Management of security attributes

	FMT_MSA.2: Secure Security Attributes FMT_MSA.3: Static attribute initialization FMT_SMF.1: Specification of management functions FMT_SMR.1: Security roles
Protection of the TSF (FPT)	FPT_STM.1: Reliable time stamps FPT_SEP.1: TSF domain separation
TOE Access (FTA)	FTA_TSE.1: TOE session establishment

5.2.1 Cryptographic Support (FCS)

5.2.1.1 Cryptographic key generation (FCS_CKM.1(a))

FCS_CKM.1.1 The *IT environment* shall generate ***symmetric*** cryptographic keys in accordance with a specified cryptographic key generation algorithm

a) **SSLv3 symmetric key and secret generation,**

b) **TLSv1 symmetric key and secret generation**

and specified cryptographic key sizes

a) **168 Bits (three independent TDEA keys),**

b) **128 and 256 Bits (AES key),**

c) **160 Bits (HMAC SHA-1 secret)**

that meet the following:

a) **conformant to SSLv3 [SSLv3] (SSLv3 symmetric key and secret generation),**

b) **conformant to RFC 2246 [TLSv1] with RFC 3268 [TLS_AES] (TLSv1 symmetric key and secret generation).**

5.2.1.2 Cryptographic key generation (FCS_CKM.1(b))

FCS_CKM.1.1 The *IT environment* shall generate ***asymmetric*** cryptographic keys in accordance with a specified cryptographic key generation algorithm **RSA** and specified cryptographic key sizes **1024 and 2048 Bits** that meet the following:

conformant to RFC 2313 [RFC2313]..

5.2.1.3 Cryptographic operation (FCS_COP.1 (a))

FCS_COP.1.1 The *IT environment* shall perform **symmetric encryption and symmetric decryption** in accordance with a specified cryptographic algorithm

a) **TDEA with three independent keys (CBC mode) ,**

b) **AES (CBC mode)**

and cryptographic key sizes

a) **168 Bits (TDEA),**

b) **128, 256 Bits (AES)**

that meet the following:

a) **conformant to FIPS 46-3 [FIPS46-3] (TDEA), conformant to FIPS 81 [FIPS81] (CBC mode),**

b) **conformant to FIPS 197 [FIPS197] (AES, CBC mode),**

and FIPS 140-2 [FIPS140-2] approved.

5.2.1.4 Cryptographic operation (FCS_COP.1(b))

FCS_COP.1.1 The *IT environment* shall perform **digest generation and verification** in accordance with a specified cryptographic algorithm

a) SHA-1

b) MD5

and cryptographic key sizes **none** that meet the following:

a) conformant to the Secure Hash Standard (SHS) as defined in FIPS 180-2 [FIPS180-2] and FIPS 140-2 [FIPS140-2] approved (SHA-1),

b) conformant to RFC 1321 [RFC1321] (MD5).

5.2.1.5 Cryptographic operation (FCS_COP.1(c))

FCS_COP.1.1 The *IT environment* shall perform **data authentication** in accordance with a specified cryptographic algorithm **HMAC SHA-1** and cryptographic key sizes **160 Bits (HMAC SHA-1)** that meet the following

conformant to FIPS 198 [FIPS198] (HMAC) and FIPS 180-2 [FIPS180-2] (SHA-1).

5.2.1.6 Cryptographic operation (FCS_COP.1(d))

FCS_COP.1.1 The *IT environment* shall perform **encryption and decryption of session key related data** in accordance with a specified cryptographic algorithm **RSA** and cryptographic key sizes **1024 or 2048 Bits** that meet the following:

conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and encryption/decryption of session key related data as defined in The SSL Protocol, Version 3 [SSLv3] and RFC 2246 [TLSv1].

5.2.1.7 Cryptographic operation (FCS_COP.1(e))

FCS_COP.1.1 The *IT environment* shall perform **generation of random numbers** in accordance with a specified cryptographic algorithm **compliant with FIPS 186-2 Appendix 3.1 or 3.2** and cryptographic key sizes **none** that meet the following:

the requirements in FIPS 186-2 [FIPS186-2], Appendix 3.1 or 3.2 as required in FIPS 140-2 annex C [FIPS140-2] and FIPS 140-2 level 1 [FIPS140-2] approved.

5.2.1.8 Cryptographic operation (FCS_COP.1(f))

FCS_COP.1.1 The *IT environment* shall perform **digital signature generation and verification** in accordance with a specified cryptographic algorithm **RSA with SHA-1 message digest** and cryptographic key sizes **1024 or 2048 Bits (RSA)** that meet the following:

conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1) and FIPS 140-2 [FIPS140-2] approved.

5.2.1.9 Cryptographic operation (FCS_COP.1(g))

FCS_COP.1.1 The *IT environment* shall perform **digital signature verification** in accordance with a specified cryptographic algorithm

a) RSA with MD2 message digest,

b) RSA with MD5 message digest

and cryptographic key sizes

a) **1024 or 2048 Bits (RSA),**

b) **1024 or 2048 Bits (RSA)**

that meet the following:

a) **conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1319 [RFC1319] (MD2),**

b) **conformant to RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1321 [RFC1321] (MD5).**

5.2.2 User Data Protection (FDP)

5.2.2.1 Subset access control (FDP_ACC.1)

FDP_ACC.1.1 The *IT environment* shall enforce the **user access control SFP** on **user access**.

Application note: This is the protection provided by the operating system (part of TOE environment) assuring that only authorized users may access the TOE and that the buffer space memory of one user is only available to that user.

5.2.2.2 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The *IT environment* shall enforce the **user access control SFP** to objects based on the following:

access rights as security attributes for users as subject and TOE code, TSF data, TOE services, buffer space memory, and the keystore as objects.

FDP_ACF.1.2 The *IT environment* shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

By default, access to the objects must be denied. Only explicitly authorized subjects may access the objects.

FDP_ACF.1.3 The *IT environment* shall explicitly authorize access of subjects to objects based on the following additional rules:

none.

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

none.

Application note: The IT environment is left freedom on how to fulfill the needed security functionality since it is not the responsibility of the ST to define such technical details.

5.2.2.3 Subset residual information protection (FDP_RIP.1)

FDP_RIP.1.1 The *IT environment* shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to** the following objects: **memory objects.**

Application note: The IT environment is left freedom on how to fulfill the needed security functionality since it is not the responsibility of the ST to define such technical details. However, this requirement entails that all storage objects holding TSF data that are visible to user space processes, for example in the case of memory objects temporarily swapped out onto hard disk drives and process context information such as CPU registers upon process context switch are subject to this residual information protection.

5.2.3 Identification and Authentication (FIA)

5.2.3.1 Timing of authentication (FIA_UAU.1 (a))

FIA_UAU.1.1 The *IT environment* shall allow ***no TSF mediated actions*** on behalf of the user to be performed before the user is authenticated.

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

Application Note: Please note that the SFR deals only with TSF mediated actions vs. actions in the IT environment. An operating system may allow some actions to be performed before successful authentication as long as those actions do not interfere with the security functions provided by the TOE as defined in this Security Target.

5.2.3.2 Timing of authentication (FIA_UAU.1 (b))

FIA_UAU.1.1 The *IT environment* shall allow ***no actions providing access to cryptographic material*** on behalf of the user to be performed before the user is authenticated.

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

Application Note: The hardware cryptographic module is expected to deny access to TSF data without proper authentication based on a PIN.

5.2.3.3 Timing of identification (FIA_UID.1)

FIA_UID.1.1 The *IT environment* shall allow ***no TSF-mediated actions*** on behalf of the user to be performed before the user is identified.

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

Application Note: See application note for FIA_UAU.1.

5.2.3.4 Management of Security Attributes (FMT_MSA.1(a))

FMT_MSA.1.1 The *IT environment* shall enforce the **user access control SFP** to restrict the ability to **modify, delete, add or invalidate** the security attributes **public key certificates** to **the Crypto-Officer**.

5.2.3.5 Management of Security Attributes (FMT_MSA.1(b))

FMT_MSA.1.1 The *IT environment* shall enforce the **user access control SFP** to restrict the ability to **set or modify** the security attributes

- **operation modes for the TOE**,
- **allowed ciphersuites for the session,**
- **authentication type (whether client authentication is required or not), and**
- **whether CRLs and/or OCSP are used for certificate validation**

to **the Crypto-Officer**.

5.2.3.6 Secure Security Attributes (FMT_MSA.2)

FMT_MSA.2.1 The *IT environment* shall ensure that only secure values are accepted for security attributes.

Application note: The TOE must be initialized in FIPS mode and CC mode.

5.2.3.7 Static attribute initialization (FMT_MSA.3)

FMT_MSA.3.1 The *IT environment* shall enforce the **user access control SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The *IT environment* shall allow the **Crypto-Officer** to specify alternative initial values to override the default values when an object or information is created.

5.2.3.8 Specification of management functions (FMT_SMF.1)

FMT_SMF.1.1 The *IT environment* shall be capable of performing the following security management functions:

- **management of access rights of the user access control SFP,**
- **management of access to the TOE functions to manage cryptographic keys and certificates**.

5.2.3.9 Security roles (FMT_SMR.1)

FMT_SMR.1.1 The *IT environment* shall maintain the roles **User and Crypto-Officer**.

FMT_SMR.1.2 The *IT environment* shall be able to associate users with roles.

Application note: These roles are required by the FIPS 140-2 security policy for ICC [ICCSEC].

5.2.4 Protection of the TSF (FPT)

5.2.4.1 TSF domain separation (FPT_SEP.1)

FPT_SEP.1.1 The *IT environment* shall maintain a security domain for *the TSF's* execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The *IT environment* shall enforce separation between the security domains of subjects in the TSC.

Application note: The second refinement in FPT_SEP.1.1 from "its own" to "the TSF's" is needed because of the SFR rephrasing which refines "TSF" to "IT environment". The SFR states that the IT environment (namely the product deploying the TOE) is required to protect the TOE.

5.2.4.2 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The *IT environment* shall be able to provide reliable time stamps for *the TSF's* use.

Application note: The second refinement in FPT_STM.1.1 from "its own" to "the TSF's" is needed because of the SFR rephrasing which refines "TSF" to "IT environment". The IT environment provides the reliable timestamps the TOE needs for certificate, OCSP response and CRL verification.

5.2.5 TOE Access (FTA)

5.2.5.1 TOE session establishment (FTA_TSE.1)

FTA_TSE.1.1 The *IT environment* shall be able to deny session establishment based on

- **the type of connection for the login: remote login to the operating system is disabled; for this, all services offering remote login are disabled.**

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are the Evaluation Assurance Level 4 components as specified in [CC] part 3. No operations are applied to the assurance components.

The following table lists the SARs for EAL4.

Table 5-3: TOE Security Assurance Requirements and Measures

Assurance Component	Assurance Measure
ACM_AUT.1 Partial CM automation ACM_CAP.4 Generation support and acceptance procedures ACM_SCP.2 Problem tracking CM coverage	The developer provides a configuration management plan describing the implemented procedures fulfilling the requirements.

ADO_DEL.2 Detection of modification ADO_IGS.1 Installation, generation, and start-up procedures	The developer provides documentation of the integrity measure for product delivery as well as installation, generation and start-up procedures for the TOE.
ADV_FSP.2 Fully defined external interfaces ADV_HLD.2 Security enforcing high-level design ADV_IMP.1 Subset of the implementation of the TSF ADV_LLD.1 Descriptive low-level design ADV_RCR.1 Informal correspondence demonstration ADV_SPM.1 Informal TOE security policy model	The developer provides a functional specification, high-level design, low-level design, security policy model, correspondence analysis as well as requested code samples for evaluation.
AGD_ADM.1 Administrator guidance AGD_USR.1 User guidance	The developer provides guidance documents as part of the TOE.
ALC_DVS.1 Identification of security measures ALC_LCD.1 Developer defined life-cycle model ALC_TAT.1 Well-defined development tools	The developer provides a documentation of the security measures for TOE development, the product life-cycle and the tools used to develop the TOE.
ATE_COV.2 Analysis of Coverage ATE_DPT.1 Testing: high-level design ATE_FUN.1 Functional testing ATE_IND.2 Independent testing – sample	The developer provides test plan, test procedures, test cases, test results and further test documentation as required.
AVA_MSU.2 Validation of analysis AVA_SOF.1 Strength of TOE security function evaluation AVA_VLA.2 Independent vulnerability analysis	The developer provides a vulnerability analysis and a misuse analysis.

6 TOE Summary Specification

6.1 TOE Security Functions

6.1.1 Introduction

This chapter describes the security functions of the GSKit that are subject to this evaluation.

6.1.2 Key Management (KEYMAN)

KEYMAN.1: The following services are provided for key generation:

- symmetric key and secret generation: a symmetric key or a secret for MAC computation is generated. The supported keys are 168 Bits TDEA keys, 128 or 256 Bits AES keys, and 160 Bits secrets for HMAC SHA-1.

  The keys are generated conformant to SSLv3 ([SSLv3]) and TLSv1 ([TLSv1] with [TLS_AES]). This is done as follows: first, random numbers as exchanged in the ClientHello and ServerHello messages and in a premaster secret generated by the client (which consists of the protocol version number and a random number). Then, a master secret is computed from the premaster secret and the random numbers of the ClientHello and ServerHello messages. This master secret is finally used for the generation of keys and secrets for MAC computation.

  The operations used are (see CRYPTO.6, CRYPTO.2):

  - random number generation for the random numbers used in the ClientHello message, the ServerHello messages and the premaster secret,
  - RSA encryption/decryption for the encryption/decryption of the premaster secret (see CRYPTO.4),
  - MD5 and SHA-1 for the computation of the master secret, and
  - MD5 and SHA-1 for the generation of keys and secrets for MAC computation, using the master secret.

- asymmetric key generation: an asymmetric key pair is generated. The supported keys are 1024 or 2048 Bits RSA keys (conformant to RFC 2313 [RFC2313]). Each prime is about half the size of the modulus; two independently generated random numbers are used to ensure that the probability of using close primes is negligible; checks include Miller-Rabin tests (3 rounds for 2048 bit modulus, 6 rounds for 1024 bit modulus) and small-factor tests.

KEYMAN.2: KEYMAN.2 is intentionally not defined.

KEYMAN.3: Access to the keystore is password-protected: the password must be entered correctly before any access is granted to the keystore. Note that this password is also used to generate the keys for the encryption and signing of data records as described in KEYMAN.4 and is enforced by a password policy as described in KEYMAN.5.

KEYMAN.4: Private keys that are to be stored in the keystore are encrypted prior to storage and decrypted after retrieval. This ensures their confidentiality. The encryption is done using TDEA. Certificates together with their trust status and certificate requests are stored unencrypted for faster access.

Each certificate, certificate/private key pair, or requested key pair together with PKCS #10 [RFC2986] certificate request data is stored in the keystore in a so-called data record. For integrity protection, the TOE generates a HMAC-SHA-1 hash over all data records by hashing all records using the password as the HMAC Key. This hash is then stored in the keystore in a special record that exists once in every keystore file, a so-called header record.

A HMAC-SHA-1 hash of the header using the password as the HMAC Key is also stored in the header record in the keystore.

The encryption standard used is PKCS #12 Password Based Encryption with SHA-1 and TDEA with three independent keys in CBC mode as described in PKCS #12 [PKCS#12], see also KEYMAN.5.

KEYMAN.5: The TOE enforces a password policy, describing the quality of the passwords that are used for the password based encryption scheme to encrypt private keys prior to storage in the keystore provided by the TOE environment and to decrypt them after retrieval.

The rules of the password policy are the following:

- The minimum password length is 14 characters.

- A password needs to have at least one lower case character, one upper case character, and one digit or special character.
- Each character of the password should not occur more than three times in a password.
- No more than two consecutive characters of the password should be identical.

Furthermore, the password characters must be randomly chosen.

The TOE generates random passwords that are to be used for as keystore passwords, i.e. the passwords that are used for the password based encryption scheme PKCS #12. The passwords are generated using the TOE's cryptographic random number generator, and consist of alphanumeric and punctuation characters.

Users are required to only use TOE generated passwords. In order to help avoiding user errors, the TOE enforces a password policy that checks the entered password against a quality metric. This metric does not check the randomness of the password but ensures at least the correct length and used characters.

The TOE Key Management Command Line Interface (CLI) tests the generated random passwords to insure these meet the password policy (FIA_SOS.1), however the Key Management API does not automatically check that the randomly generated password meet the policy. All passwords are checked against the requirements of FIA_SOS.1 when they are used as a Key Management API call in order to use keystores.

KEYMAN.6: KEYMAN.6 is intentionally not defined.

KEYMAN.7: Public key certificates that are conformant to the Standard X.509 [X.509] are used for SSL/TLS connections; certificates of the versions 1, 2, and 3 are supported. For intermediate CAs, only version 3 certificates are supported.

These certificates can be imported and exported using the Key Management API and the CLI. Furthermore, public key certificates can be imported during the setup of an SSL/TLS connection. This allows the TOE to use a larger external public key infrastructure to establish trust in the authenticity of public keys received from external parties.

Several well-know CA certificates (IBM, Thawte, and Verisign) are always contained in a newly created keystore and renewal certificates can be imported as described above.

Public key certificates can be generated using RSA and SHA-1; the generated certificates are conformant to X.509 [X.509]; certificates of versions 1, 2, and 3 are supported. To generate such a certificate, the public key is digitally signed such that this digital signature can afterwards be used for verifying the authenticity of that key.

Certificate request can be generated and used to request the signing of the public key used to create a certificate. These certificate requests are conformant to PKCS #10 [RFC2986]. They can be stored in or retrieved from the keystore.

The certification path validation algorithm is done as described in [RFC3280]. In order to be able to deal with certificates that are signed using RSA with MD2, RSA with MD5, RSA with SHA-2, RSA with 4096 bit keys, and DSA with SHA-1 the TOE is able to verify such signatures, but MD2, MD5, SHA-2, DSA, and 4096 bit-RSA keys are not used by the TOE for certificate generation.

This is supported by CRYPTO.5.

KEYMAN.8: PKCS#12 file im- and export.

The TOE imports cryptographic material into its keystore from PKCS#12-formatted files [PKCS#12] employing password privacy mode and password integrity mode using the same password and pbeWithSHAAnd3-KeyTriple-DES-CBC, supporting KeyBags, PKCS-8ShroudedKeyBags, and/or CertBags.

The TOE exports cryptographic material from its keystore into PKCS#12-formatted files employing privacy mode and password integrity mode using the same password and pbeWithSHAAnd3-KeyTriple-DES-CBC, supporting KeyBags, PKCS-8ShroudedKeyBags, and/or CertBags.

KEYMAN.9: PKCS#11 support.

The TOE implements the PKCS#11 protocol specified in [PKCS#11-2.10] with AES augmentations from [PKCS#11-2.20] to access hardware cryptographic modules in the IT environment in order to retrieve and/or store public key certificates, private keys and security attributes, and to request the execution of cryptographic primitives.

In CC mode GSKit uses only certificates held on PKCS#11 FIPS 140-2 certified devices that have the CKA_TRUSTED attribute set. If the PKCS#11 Device does not support this Certificate Object Attribute then the GSKit keystore is the only source of trusted certificates allowed.

If the crypto module in the IT environment requires a PIN, GSKit can temporarily store the user-supplied PIN in memory during run-time and hand it down to the module when required.

This functionality satisfies the following SFRs:

Ÿ KEYMAN.1:

a. FCS_CKM.1 (Cryptographic key generation)

Ÿ FCS_CKM.1(a) applies to the generation of symmetric keys

Ÿ FCS_CKM.1(b) applies to the generation of asymmetric keys

Ÿ KEYMAN.3: FIA_UAU.2 (User authentication before any action)

Ÿ KEYMAN.4:

a. FCS_COP.1 (Cryptographic operation)

Ÿ FCS_COP.1(a) applies to TDEA encryption

Ÿ FCS_COP.1(c) applies to HMAC SHA-1

b. FPT_ETT_GSK.1 (Basic external TSF data transfer protection)

Ÿ FPT_ETT_GSK.1(a) applies to private keys

Ÿ FPT_ETT_GSK.1(b) applies to certificates

Ÿ KEYMAN.5:

a. FIA_SOS.2 (TSF Generation of secrets)

b. FIA_SOS.1 (Verification of secrets)

Ÿ KEYMAN.7:

a. FPT_TDC.1(a) (Inter-TSF basic TSF data consistency)

b. FCS_COP.1 (Cryptographic operation)

Ÿ FCS_COP.1(f) for RSA/SHA-1

Ÿ FCS_COP.1(g) for RSA/MD2 and RSA/MD5

c. FDP_DAU.1 (Basic data authentication)

Ÿ KEYMAN.8:

a. FCS_COP.1 (Cryptographic operation)

Ÿ FCS_COP.1(a) for TDEA

Ÿ FCS_COP.1(b) for SHA-1

b. FPT_TDC.1(c) (Inter-TSF basic TSF data consistency)

 KEYMAN.9:

a. FPT_TDC.1(e) (Inter-TSF basic TSF data consistency)

6.1.3 Cryptographic Algorithms (CRYPTO)

CRYPTO.1: The algorithms for symmetric encryption and decryption are:

Ÿ TDEA (Triple DES) with three independent keys – CBC mode; 168 Bits as defined in FIPS 46-3 [FIPS46-3] (TDEA) and FIPS 81 [FIPS81] (CBC mode)

Ÿ AES - CBC mode; 128 & 256 Bits as defined in FIPS 197 [FIPS197] (AES, CBC mode)

All algorithms are FIPS 140-2 [FIPS140-2] approved.

CRYPTO.2: The algorithms for digest generation and verification are:

Ÿ SHA-1 as defined in FIPS 180-2 [FIPS180-2]

Ÿ MD5 as defined in RFC 1321 [RFC1321]

The SHA-1 algorithm is FIPS 140-2 [FIPS140-2] approved.

MD5 is not used within the cipherspec parts of the SSL/TLS ciphersuites since it is not FIPS approved but it is used together with HMAC for the setup of SSL/TLS connections.

CRYPTO.3: The algorithm for data authentication is HMAC SHA-1; 160 Bits as defined in FIPS 198 [FIPS198] (HMAC) and FIPS 180-2 [FIPS180-2] (SHA-1).

CRYPTO.4: The algorithm used for the encryption/decryption needed for exchange of the premaster secret used to generate the session key for the SSL/TLS connection is:

Ÿ RSA; 1024 or 2048 Bits as defined in The SSL Protocol, Version 3 [SSLv3] and RFC 2246 [TLSv1]

RSA encryption/decryption which is not FIPS 140-2 [FIPS140-2] approved is used for this purpose; the generation of the session key is an operation covered by the key exchange algorithm parts (not the cipherspec parts) given in SSL/TLS ciphersuites.

The TOE supports only cipher suites for SSLv3 and TLSv1 that use RSA for the session key exchange. Other methods for key exchange defined in the SSLv3 [SSLv3] and TLSv1 [TLSv1] standard are not supported by the TOE in its evaluated configuration.

CRYPTO.5: RSA signatures can be generated and verified, the algorithms are RSA with SHA-1 message digests; 1024 or 2048 Bits as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1).

Signatures using other algorithms or key sizes are not generated by the TOE but they can be verified, the algorithms are

- RSA with MD5 message digests; 1024 or 2048 or 4096 Bits as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1321 [RFC1321] (MD5), and
- RSA with MD2 message digests, 1024 or 2048 or 4096 Bits as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and RFC 1319 [RFC1319] (MD2).
- RSA with SHA-2 message digests, 1024 or 2048 or 4096 Bits as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-224, SHA-256, SHA-384, SHA-512),
- RSA with SHA-1 message digests, 4096 Bits as defined in RFC 2437 [RFC2437] and RFC 2313 [RFC2313] (RSA) and FIPS 180-2 [FIPS180-2] (SHA-1) and FIPS 140-2 [FIPS140-2] approved,
- DSA with SHA-1 message digests, 1024 Bits as defined in FIPS 186-2 [FIPS186-2] and FIPS 180-2 [FIPS180-2] (SHA-1) and FIPS 186-2 [FIPS186-2] approved.

The RSA signature generation/verification with SHA-1 is FIPS 140-2 [FIPS140-2] approved, as is the DSA with SHA-1 signature verification; the RSA signature verification with MD2 ,MD5 or SHA-2 is not.

CRYPTO.6: Random number generation:

A random number is generated using the Universal Software Base TRNG algorithm invented by IBM [EPRAND]. The TRNG operates by detecting timing jitter across repeated identical sequences of operations. This jitter serves as an entropy source. The mechanism to do this uses timing jitter in the internal CPU timebase register (TBR) measured over a fixed number of machine instruction execution. This raw jitter source is then subjected to a Shannon distribution test before being used as the source in the hashing function SHA-1 to produce the final random data vales. SHA-1 is used here as a compression function. The design also protects against time base drift synchronization..

The Universal Software Base TRNG, the RSA key generation, and SHA-1 are FIPS 140-2 [FIPS140-2] approved; RSA encryption/decryption and MD5 are not.

This functionality satisfies the following SFRs:

Ÿ CRYPTO.1: FCS_COP.1(a) (Cryptographic operation)

Ÿ CRYPTO.2: FCS_COP.1(b) (Cryptographic operation)

- CRYPTO.3: FCS_COP.1(c) (Cryptographic operation)
- CRYPTO 4: FCS_COP.1(d) (Cryptographic operation)
- CRYPTO.5: FCS_COP.1 (Cryptographic operation)
    a. FCS_COP.1(f) for RSA/SHA-1
    b. FCS_COP.1(g) for RSA/MD2 and RSA/MD5
- CRYPTO.6
    a. FCS_COP.1(e) for random number generation

6.1.4 Secure Channels (SECCHAN)

SECCHAN.1: GSKit offers the possibility to set up secure/trusted channels using the SSLv3 or TLSv1 protocol, earlier versions of SSL are disabled by initializing the TOE in FIPS mode. These channels offer confidentiality and integrity protection of the data transmitted over them. This functionality is accessible through the SSL API of the GSKit. It is both possible to write client- and server-applications using this API and the libraries are capable of multi-threading.

The TOE implements the SSL protocol as defined in The SSL Protocol, Version 3 [SSLv3] and the TLS protocol as defined in RFC 2246 [TLSv1] with the extension of the AES ciphersuites as defined in RFC 3268 [TLS_AES]. When there are multiple options, not all options are implemented. This applies to the supplied ciphersuites (not all possible ciphersuites are supported), the authentication (total anonymity mode must not be used, i.e. the server always has to authenticate), and the key agreement using RSA (FORTEZZA and Diffie-Hellman are not supported).

The cipherspec parts of SSL/TLS ciphersuites provided by the TOE in FIPS mode contain only the algorithms that are FIPS 140-2 [FIPS140-2] approved (see the security functions of CRYPTO). The following ciphersuites are supported:

a) For TLS v1.0 (RFC 2246 Appendix A.5)
    - CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0A };

b) For TLS v1.0 (RFC 3268 the AES):
    - CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA = { 0x00, 0x2F };
    - CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA = { 0x00, 0x35 };

c) For the SSL v3.0 and TLS v1.0
    - CipherSuite SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA = { 0xFE,0xFF };

SECCHAN.2: For authentication, X.509 certificates as defined in the X.509 standard [X.509] are used.

The TOE may operate in server or client mode.

When the TOE is in client mode, server authentication (with a valid X.509 certificate and the associated private key) is required for the successful completion of the SSLv3 and TLSv1 channels. This means that the session is only established if the certificate of the communication partner can be validated by the TOE and the communication partner proves that it has the associated private key.

When the TOE is in server mode, it can be configured whether the TOE requires client authentication. If the TOE is configured to require client authentication, client authentication (with a valid X.509 certificate and the associated private key) is needed for the successful establishment of the SSLv3 and TLSv1 channels without restrictions, i.e. the SSL/TLS session is established if the certificate of the communication partner can be validated by the TOE and the communication partner proves that it has the associated private key. If the communication partner presents a certificate the TOE can not validate or a certificate the TOE verifies as invalid (due to expiration or due to being revoked), the session is terminated. If the TOE is configured to require client authentication and the client does not present any certificate the TOE establishes the session and sets a flag indicating that the communication partner did not present a certificate. The application using the TOE can check this flag and decide to terminate the session, send an error message to the communication partner and then terminate the session or to continue the session with a set of services adapted to the fact that the communication partner did not authenticate itself. If the TOE requires no client authentication, no authentication of the communication partner is performed.

The validity of a certificate stored in the TOE's keystore is managed by setting its trust status. The trust status of a certificate is a security attribute that tells whether a certificate is trusted or not. If a certificate is not trusted, it is not considered a valid certificate (i.e. it is invalid).

The path validation algorithm implemented by the TOE is compliant to ITU-T Technical Recommendation X.509 [X.509] and also follows the PKIX path validation logic as specified in [RFC3280].

The TOE does the path validation with respect to the current date and time and for the validation of a certificate, the entire chain is followed to the root of the signing tree, thus any intermediate certificates are considered in the chain validation.

The TOE can be configured to use an OCSP responder in the IT environment to query the revocation status of a certificate. The OCSP responder is located either by being pre-configured or via the AIA extension. Responses will be verified and must be signed by an instance that is an OCSP signing authority for the certificate in question known to the TOE, the CA that issued the certificate in question, or an instance who has the id-ad-ocspSigning extension in its certificate and has been issued by the CA that issued the certificate in question.

The TOE supports the following three trust models, as specified in [RFC2560], for the OCSP responder:

- The Responder could be explicitly trusted (i.e., as a trust anchor).
- The CA that issued the certificate signs the OCSP response. In this case the CA uses the same key to sign the OCSP response and the certificate in question.
- The CA issues a certificate to the OCSP responder (Delegated OCSP Responder Model). In this case, the CA uses the same key to sign the OCSP Responder certificate and the certificate in question.

Revocation checking of an OCSP responder is performed per [RFC2560]. The AIA or CDP extensions are used to check the revocation status except in the case where the 'id-pkix-ocsp-nocheck' extension is set.

The thisUpdate, nextUpdate and producedAt time elements in the OCSP response are used as follows in determining the validity of the OCSP response:

- The producedAt time is ignored.
- The thisUpdate time is ignored.
- If the nextUpdate time is less than the current time, the response is considered old but still usable as long as no other non-stale information is found by all forms and sources of revocation checking/information. If out of date information is used, GSKit retains the certificates' status as UNDETERMINED for the application. If the nextUpdate time is empty, the response is used as if the nextUpdate time was greater than the current time.

In all cases, GSKit never caches responses and always requests fresh information from the OCSP responder. The TOE offers the option of nonce checking to allow or disallow the use of responses that have been cached by the responder and had been previously received, and to prevent replay attacks on the network. GSKit supports nonce checking for requests that it issues and/or for responses received from the responder – it is the application's choice whether or not to enable these.

If the status of a certificate is undetermined due to an unsuccessful attempt to query the OCSP responder, and if CRL checking is enabled, the TOE will attempt to verify the validity of the certificate by using CRLs as described below. If CRL checking is not configured and the certificate status is undetermined, the TOE sets a corresponding marker that can be queried by the applications using the TOE.

The TOE can be configured to do certificate validation with the help of X.509 [X.509] conformant CRLs; CRLs of versions 1 and 2 are supported. The use of CRLs is not mandatory. CRLs can be used both in server mode with client authentication enabled, and in client mode.

If CRLs are to be used for certificate validation, an LDAP client is used to query an LDAP server for CRLs, an HTTP server is queried, or a file is retrieved from the underlying file system, depending on the URI provided in the certificate to be checked. LDAP client and server, or HTTP server, must be provided by the environment and the environment must provide current and correct CRLs. In case of LDAP, the TOE accesses the LDAP client via the LDAP client's API. Then the LDAP client establishes a TCP/IP connection to the LDAP server, retrieves one or more CRLs as result of its query, and passes the CRLs back to the TOE, which uses them as part of the certificate path validation.

If CRLs are configured to be part of the certificate path validation, the following behavior is of particular relevance to their evaluation:

- If the query to the application-specified distribution point fails, the revocation status for the certificate is "revoked".

- If the query to the application-specified distribution point succeeds and no CRL is returned, the revocation status for the certificate is “undetermined”.
- If multiple CRLs are returned as result of the query, all of them are evaluated until a valid and correctly issued CRL revokes the certificate in question or all CRLs have been processed.
- A CRL is rejected if its validation fails, for example due to a bad signature.
- An indirect CRL is rejected.
- The following cases do not automatically lead to a CRL validation failure:
  - If an unrecognized critical extension is encountered in the crlExtensions field and the certificate is listed in the CRL, the certificate revocation status is “revoked”. If the certificate is not listed in the CRL, the CRL is rejected.
  - If an unrecognized critical entry extension is encountered in the crlEntryExtensions field and the certificate is listed in the CRL, the certificate revocation status is “revoked”. If the certificate is not listed in the CRL, the CRL is rejected.
  - If a CRL is encountered containing an out of date ‘NextUpdate’ value and the certificate is listed in the CRL, the certificate revocation status is “revoked”. If the certificate is not listed in the CRL, the CRL is rejected.

A rejected CRL yields a revocation status of “undetermined”.

Please note that the TOE, being an SSL/TLS implementation, does not differentiate between the status “undetermined” and “not revoked” for certificates and establishes a trusted channel if the certificate itself is determined to be valid.

A certificate revocation status of “revoked” results in termination of the SSL/TLS session with the entity that provided the certificate.

Distribution points in certificates are evaluated as follows: if a CDP is specified it will only be used if it is an X.500 Directory, FILE or HTTP name since only LDAP, files and HTTP are supported for CRLs. If the CDP is in a format other than an X.500 directory , file URI or HTTP name, the Issuer Name will be used unless the CDP is marked critical in which case GSKit will fail the validation.

The supported extensions are specified in the certificate, OCSP and CRL policies, which comply with X.509 [X.509]. [GSKTRUST] describes which attributes are evaluated as follows:

Standard Certificate Policy)

Supported Fields:

- *OuterSigAlgID*
- *Signature*
- *Version*
- *SerialNumber*
- *InnerSigAlgID*
- *Issuer*
- *Validity*
- *SubjectName*
- *SubjectPublicKeyInfo*
- *IssuerUniqueID*
- *SubjectUniqueID*

Supported Extensions:

- *AuthorityKeyID*
- *AuthorityInfoAccess*
- *SubjectKeyID*

- *IssuerAltName*
- *SubjectAltName*
- *KeyUsage*
- *BasicConstraints*
- *PrivateKeyUsage*
- *CRLDistributionPoints*
    - *DistributionPoint*
        - *DistributionPointName*
        - *NameRelativeToCRLIssuer -- (not supported)*
        - *Reasons -- (ignored)*
        - *CRLIssuer fields -- (not supported - the CRL issuer name defaults to the certificate issuer name)*
- *Extended KeyUsage*
- *NameConstraints*
- *CertificatePolicies*
    - *PolicyInformation*
        - *PolicyIdentifier*
        - *PolicyQualifiers – (not supported)*
- *PolicyMappings*
- *PolicyConstraints*

Standard CRL Policy

Supported Fields:

- *OuterSigAlgID*
- *Signature*
- *Version*
- *InnerSigAlgID*
- *Issuer*
- *ThisUpdate*
- *NextUpdate*
- *RevokedCertificate*
    - *UserCertificate*
    - *RevocationDate*

Supported CRLEntry Extensions:

- *(none)*

Supported CRL Extensions:

- *AuthorityKeyID*
- *IssuerAltName*
- *CRLNumber*
- *IssuingDistributionPoint*

- *DistributionPoint*
- *DistributionPointName*
    - *FullName (X.500 Name and LDAP/HTTP/FILE Format URI only)*
    - *NameRelativeToCRLIssuer -- (not supported)*
- *Reasons -- (ignored)*
- *CRLIssuer*
- *OnlyContainsUserCerts -- (not supported)*
- *OnlyContainsCACerts -- (not supported)*
- *OnlySomeReasons -- (not supported)*
- *IndirectCRL -- (rejected)*

Standard OCSP Policy

Request Supported Fields:

- Signature (*Optional*)
- Version (Version 1 Only)
- RequesterName (Optional)
- RequestList (single request only)
    - *CertID*
    - *singleRequestExtensions* (not supported)
- RequestExtensions
    - Nonce (if *enabled*)

Response Supported Fields:

- ResponseStatus
- Response
    - *responseType* (id-pkix-ocsp-basic)
    - *BasicOCSPResponse*
        - *Signature*
        - *Certs*
            - *Extensions*
            - *extendedKeyUsage*
                - *id-kp-OCSPSigning*
            - *id-pkix-ocsp-nocheck*
        - *ResponseData*
            - *Version (Version 1 Only)*
            - *ResponderID (by name or by hash)*
            - *ProducedAt (ignored)*
            - *Responses (multiple responses suppoted)*
                - *SingleResponse*
                    - *certID*
                    - *certStatus*

            - *RevokedInfo* (ignored)
        - *thisUpdate (ignored)*
        - *nextUpdate*
        - *singleExtensions (ignored)*
    - *responseExtensions*
        - *Nonce (if enabled)*

A SSL/TLS connection is not set up if the communication partners cannot find a common cipher suite supported by the TOE. This ensures that there cannot be any unprotected connection in the case that the remote application does not support the cipher suites provided by the TOE in FIPS mode.

This is supported by CRYPTO.5.

SECCHAN.3: Sensitive data must be protected between different sessions. Unprotected (i.e. cleartext) critical security parameters and random numbers are considered such sensitive data and are cleared before the buffer is released. The mechanism for this is that the TSF marks all critical data objects. Before releasing the buffers, the TSF clears all the objects that are marked as critical.

The random number generator clears its state automatically when the module is removed from memory.

This functionality satisfies the following SFRs:

Ÿ SECCHAN.1:

a. FDP_UCT.1 (Basic data exchange confidentiality)

b. FDP_UIT.1 (Data exchange integrity)

c. FTP_ITC_PD-0108.1 (Inter-TSF trusted channel)

d. FDP_ACF.1 (Security attribute based access control)

Ÿ SECCHAN.2:

a. FDP_ACC.1 (Subset access control)

b. FDP_ACF.1 (Security attribute based access control)

c. FMT_MSA.3 (Static attribute initialization)

d. FPT_TDC.1(b) (Inter-TSF basic TSF data consistency)

e. FPT_TDC.1(d) (Inter-TSF basic TSF data consistency) – OCSP

Ÿ SECCHAN.3:

a. FDP_RIP.2 (Full residual information protection)

Ÿ FDP_RIP.2(a) for the random number generation

Ÿ FDP_RIP.2(b) for the buffers

6.1.5 Self-tests and Failure Handling (STATE)

STATE.1: The ICC component is able to do self-tests upon initialization. This is not directly available to the user or program (since the ICC API has no external interface) but the self-tests are called automatically with the first call to the SSL environment initialization of the TOE; calling this environment initialization function does the initialization that is needed for setting up an SSL/TLS connection. So the self-tests of the ICC component are invoked and conducted before the first use of the cryptographic algorithms of the ICC component for setting up an SSL/TLS connection.

The self-tests check cryptographic operations and the software integrity (an RSA signature verification). The following self-tests are done:

- **Integrity test** of a digital signature (a SHA-1 hash signed with RSA) that has been computed over the dynamic parts of the ICC component and that is provided with the static parts of the ICC component.
- **Known answer tests** for encryption/decryption of
  - TDEA in CBC mode
  - AES in CBC mode
  - SHA-1 message digest generation (signed with RSA)
  - RSA
  - HMAC-SHA-1
  - SHA-1
  - PRNG
- **Pairwise consistency tests** for
  - RSA signature generation and verification
  - DSA signature generation and verification
- **Pairwise consistency tests** for public/private key generation: the consistency of the keys is tested by the generation and verification of a digital signature. If the signature cannot be verified, the test will fail.
- **Continuous RNG tests** for the random number generator: generated random number blocks or random number bits sequences are compared. The test will fail if an equal block or bit sequence is generated.

STATE.2: The TOE will cause a termination of the SSL/TLS session (in case of 1, 2, 3, 4 and 6 below), Inability to establish a SSL/TLS session (in case of 5 and 6 below), or a disabling of all ICC cryptographic functions (in case 7), thus entering a secure state, if one of the following events occurs:

1. SSL or TLS handshake failure, including decryption of encrypted handshake messages and data integrity
2. failing to correctly decrypt SSL or TLS application data messages received from the communication partner
3. data integrity failure for SSL or TLS application data messages received from the communication partner (including attempted replay of messages)
4. SSL/TLS protocol detected buffer sequence errors
5. integrity failures of private keys and certificates read from the keystore
6. internal failure intercepted by the TOE
7. failure of the ICC component's self tests

This functionality satisfies the following SFRs:

Ÿ STATE.1: FPT_TST_WEAK.1 (Partial TSF testing)

Ÿ STATE.2: FPT_FLS.1 (Failure with preservation of secure state)

7 Rationale

The rationale section provides additional information and demonstrates that the security objectives and the security functions defined in the previous chapter are consistent and sufficient to counter the threats defined in chapter 2.

7.1 Security Objectives Rationale

7.1.1 Security Objectives Coverage

The following table provides a mapping of TOE objectives to threats and policies, showing that each objective is at least covered by one threat or policy.

Table 7-1: Mapping Objectives to Threats and Policies

Objective	Threat / Policy
O.KEYSTORE	P.KEYSTORE
O.SECCHANNEL	P.SECCHANNEL
O.SELFSEC	P.SELFSEC
O.SESSIONDATA	P.SECCHANNEL
O.TRANSFER	P.TRANSFER

The following table provides a mapping of the objectives for the TOE environment to assumptions, threats and policies, showing that each objective is at least covered by one assumption, threat or policy.

Table 7-2: Mapping Objectives for the Environment to Threats, Assumptions and Policies

Env. Objective	Threat / Assumption / Policy
OE.CRYPTO	P.SECCHANNEL
OE.DATA	TE.ACCESS
OE.MODE	P.MODE
OE.OSLOGIN	P.MODE
OE.USER	TE.ACCESS
OE.ADMIN	A.ADMIN, TE.ACCESS
OE.CERTIFICATES	A.CERTIFICATES
OE.FIPS140	P.KEYSTORE
OE.INTEGRATE	A.INTEGRATE
OE.LDAP	TE.BYPASS, A.LDAP
OE.OCSP	A.OCSP
OE.OS	P.MODE
OE.PASSWORD	A.PASSWORD
OE.PHYSEC	TE.ACCESS
OE.TIMESTAMP	A.TIMESTAMP

7.1.2 Security Objectives Sufficiency

7.1.2.1 A.ADMIN

Those responsible for the administration of the TOE and the TOE environment are competent and trustworthy individuals, capable of managing the TOE and the TOE environment and the security of the information it contains.

This assumption is addressed by the environment objective that states that those responsible for the administration of the TOE must be trained such that they are capable of managing the TOE and the security of the information it contains; this includes making sure that backup copies of TSF data are appropriately protected. Administrators are trustworthy. (OE.ADMIN). Please note that in addition to fulfilling the assumption A.ADMIN, the objective OE.ADMIN covers the protection of backup data which helps to counter the threat TE.ACCESS.

7.1.2.2 A.CERTIFICATES

The TOE environment is responsible for the management and generation of the certificates.

This assumption is addressed by the environment objective that states that the TOE environment is responsible for the management and generation of the certificates (OE.CERTIFICATES).

7.1.2.3 A.INTEGRATE

Those who integrate the TOE as part of a larger product or system are assumed to follow the guidance provided with the TOE with respect to the configuration of the TOE, the use of TOE functions and interfaces and the protection of the TOE code and data. They are trustworthy and do not try to subvert or bypass TOE security functions.

This assumption is addressed by the environment objective that states that those who integrate the TOE as part of a larger product or system are following the guidance provided with the TOE with respect to the configuration of the TOE, the use of TOE functions and interfaces and the protection of the TOE code and data. They are trustworthy and do not try to subvert or bypass TOE security functions (OE.INTEGRATE).

7.1.2.4 A.LDAP

If a CRL is required by the TOE, the TOE environment must provide a valid, current CRL

This assumption is addressed by the environment objective that states that if a CRL is required by the TOE, the TOE environment must provide a valid, current CRL, and TCP/IP connections used for retrieving CRLs must be an internal communication link within a protected network. The environment provides a valid, current CRL conformant to [X.509] (OE.LDAP).

Please note that the part dealing with the protection of the communication link is not derived from A.LDAP but from the threat TE.BYPASS.

7.1.2.5 A.OCSP

OCSP responders used by the TOE are trustworthy.

The objective OE.OCSP requires consumers to ensure that only trustworthy OCSP responders are used by the TOE.

7.1.2.6 A.PASSWORD

Users knowing the password to protect the keystore will apply protection measures that prohibit unauthorized access to or disclosure of password information.

This assumption is addressed by the environment objective that states that users knowing the password to protect the keystore must apply protection measures that prohibit unauthorized access to or disclosure of password information (OE.PASSWORD).

7.1.2.7 A.TIMESTAMP

The environment will provide reliable timestamps.

This assumption is addressed by the environment objective that states that the environment must provide reliable timestamps (OE.TIMESTAMP). These timestamps are needed for certificate, OCSP response and CRL verification.

7.1.2.8 TE.ACCESS

TOE functions, TOE code, and security attributes may be accessible for unauthorized modification and user data or TSF data may be accessible for unauthorized disclosure, access, and modification (for example modification of the TOE or cryptographic data) by users that have gained local access to the system the TOE is integrated to or to backup data.

This threat is countered by

- the environment objective that states that the TOE environment (e.g. the operating system) must protect the TOE code from unauthorized modification and user data and TSF data from unauthorized disclosure, access, and modification. This includes also the protection of the TOE against other applications in the TOE environment. (OE.DATA).
- the environment objective that states that the processing resources of the TOE must be located within controlled access facilities such that they are protected against unauthorized physical access (OE.PHYSEC).
- the environment objective that states that the environment must only allow authorized users to access the TOE (OE.USER).
- the environment objective that states that those responsible for the administration of the TOE must be trained such that they are capable of managing the TOE and the security of the information it contains; this includes making sure that backup copies of TSF data are appropriately protected. Administrators are trustworthy (OE.ADMIN).

7.1.2.9 TE.BYPASS

CRLs may be transmitted from LDAP or HTTP servers to the TOE over unprotected communication channels such that they may be modified or deleted by a remote attacker. No trusted channel can be set up if no CRL can be retrieved or the retrieved CRL is corrupted (i.e. not current or broken). If with a replay attack a technically still valid CRL that does not contain all revoked certificate information is resent, a trusted channel might be established for a revoked certificate

This threat is countered by the environment objective that states that if a CRL is required by the TOE, the IT environment must provide a valid, current CRLand the TCP/IP connection to the server must be an internal communication link within a protected network. The server provides a valid, current CRL conformant to [X.509] (OE.LDAP).

Please note that only the part about the protection of the communication link is required to counter the threat TE.BYPASS.

7.1.2.10 P.MODE

FIPS mode and CC mode must be set for the TOE. This also ensures that the TOE shall be operated in a configuration compliant to the FIPS validated configuration of the ICC component described in the FIPS 140-2 level 1 security policy for ICC [ICCSEC].

This policy is fulfilled by

- the environment objective that states that the TOE must be initialized to operate in FIPS mode and CC mode (OE.MODE).
- the environment objective that states that remote login to the operating system must be disabled (by disabling all services that offer remote login) (OE.OSLOGIN).
- choosing one of the operating systems that have been used for testing of the ICC FIPS component during the FIPS 140-2 level 1 [FIPS140-2] validation, AIX V5.2 (OE.OS).

7.1.2.11 P.KEYSTORE

Private keys must be protected from unauthorized disclosure when they are stored in the keystore. In addition to that, a mechanism against undetected modification of the whole of the key data stored in the keystore must be provided.

This policy is fulfilled by the TOE objective that states that the TOE must protect private keys stored in the keystore from unauthorized disclosure. In addition to that, the whole of the key data stored in the keystore must be protected against undetected modification. (O.KEYSTORE).

In case that a hardware crypto module in the IT environment is used as key and certificate store, OE.CRYPTO provides for the authentication of users before granting access to key material, and OE.FIPS140 requires consumers to ensure that the module is operated in a FIPS 140-1 or -2 validated mode, providing for the necessary tamper resistance.

7.1.2.12 P.SECCHANNEL

The TOE shall offer a product of the TOE environment the functionality to communicate with a remote trusted IT product using a cryptographically secured trusted channel that protects the confidentiality and integrity of data being transmitted over this channel and provides the ability to authenticate the communication partners before transmitting user data that requires protection. User data and secret TSF data of a session must be protected such that they are not available to other sessions.

This policy is fulfilled by

Ÿ the TOE objective that states that the TOE shall offer a product of the TOE environment the functionality to communicate with a remote trusted IT product using a trusted channel that protects the confidentiality and integrity of user data being transmitted over this channel and provides the ability to authenticate the communication partners before transmitting user data that requires protection (O.SECCHANNEL)

Ÿ the TOE objective that states that the TOE must protect secret TSF data (private/secret keys, session keys, and critical security parameters) of its individual trusted channel sessions and must ensure that no such data is made available to other trusted channel sessions also provided by the TOE. (O.SESSIONDATA).

Ÿ Alternatively, the cryptographic operations needed to establish secure channels are provided by a cryptographic module in the IT environment rather than by the TOE itself. In this case, OE.CRYPTO contributes to the fulfillment of the policy.

7.1.2.13 P.SELFSEC

The TOE shall implement self-tests for cryptographic modules and failure detection with preservation of a secure state for the trusted channel component.

This policy is fulfilled by the TOE objective that states that the TOE must implement self-tests for cryptographic modules and failure detection with preservation of a secure state for the trusted channel component (O.SELFSEC).

7.1.2.14 P.TRANSFER

The TOE shall be able to exchange TSF data with entities in the TOE environment.

The objective O.TRANSFER requires the TOE to be able to interpret data that is exchanged via interfaces following the specifications of PKCS#11 and PKCS#12.

7.2 Security Requirements Rationale

This section describes the mutual support and internal consistency of the components (requirements) selected for this Security Target.

7.2.1 Explicit Security Functional Requirements Rationale

Two explicit security functional requirements are defined in this Security Target:

FPT_ETT_GSK.1 Basic external TSF data transfer protection

FPT_TST_WEAK.1 Partial TSF testing.

In addition to that, the security functional requirement FTP_ITC.1 has been replaced by FTP_ITC_PD-0108.1 according to [PD-0108].

The reasons for using these explicit SFRs are given in the following.

7.2.1.1 FPT_ETT_GSK.1 Basic external TSF data self-protection

This security functional requirement has been defined as an extension to part 2 of the Common Criteria to address a situation where a TOE needs to export (and later re-import and use) TSF data outside of the TOE scope of control. The TOE needs to ensure that the data is protected by itself when it is not under the control of the TOE. A typical example is critical TSF data that needs to be stored for later retrieval outside the TOE scope of control (even though the keystore is part of the TOE, the files that are used to implement it are located in the TOE environment). Requirements may exist to protect the TSF data from disclosure and/or to allow the TOE to detect (upon re-import) unauthorized modifications made to the TSF data.

Although cryptographic methods will usually be used to provide this functionality, other methods (e. g. a non-cryptographic transformation using a secret only known to the TOE) are not excluded and therefore no dependency to cryptographic functions is defined.

There has been found no SFR in [CCv2] part 2 that is appropriate for this functionality. There are FPT_ITC.1 for confidentiality protection and FPT_ITI.1 for integrity protection but these SFRs have been found to be not appropriate since they deal with transmission of data between the TOE and other entities whereas in this context, data has to be stored outside the TSC. Furthermore, SFRs of class user data protection (FDP) do not apply since FPT_ETT_GSK.1 aims to protect TSF data rather than user data.

7.2.1.2 FPT_TST_WEAK.1 Partial TSF testing

This security functional requirement has been derived from the SFR FPT_TST.1 as defined in part 2 of the Common Criteria. FPT_TST_WEAK.1 is identical to the first two functional elements of FPT_TST.1 while the third functional element of FPT_TST.1 (“FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of stored TSF executable code”) is omitted. In the case of the TOE defined in this Security Target, the TOE is not able to satisfy FPT_TST.1.3.

7.2.1.3 FTP_ITC_PD-0108.1 Inter-TSF trusted channel

The SFR FTP_ITC.1 has been replaced by FTP_ITC_PD-0108.1 as described in PD-0108 [PD-0108]. PD-0108 addresses the element FTP_ITC.1.3. A problem arises when the functions for which either the TSF or a remote trusted IT product must use a trusted channel would have to be specified since FTP_ITC.1.3 only refers to the local TOE and specifically refers to the initiation and not to the use of the trusted channel.

PD-0108 corrects this error identified by CCEVS in the requirement and provides an interpretation created by NIAP to correct the offending wording.

7.2.2 Security Functional Requirements Rationale

The following table provides a mapping of objectives to security functional requirements, showing that that each security functional requirement covers at least one objective and that each objective is covered by at least one security functional requirement.

Table 7-4: Mapping Objectives to Security Functional Requirements

Objective	Security Functional Requirement
O.KEYSTORE	FCS_COP.1(c), FIA_SOS.1, FIA_SOS.2, FIA_UAU.2, FPT_ETT_GSK.1(a), FPT_ETT_GSK.1(b)
O.SECCHANNEL	FCS_CKM.1(a), FCS_CKM.1(b), FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), FCS_COP.1(g), FDP_ACC.1, FDP.ACF.1, FDP_DAU.1, FDP_UCT.1, FDP_UIT.1, FMT_MSA.3, FPT_TDC.1(a), FPT_TDC.1(b), FPT_TDC.1(d), FTP_ITC_PD-0108.1
O.SELFSEC	FPT_FLS.1, FPT_TST_WEAK.1
O.SESSIONDATA	FDP_RIP.2(a), FDP_RIP.2(b)
O.TRANSFER	FCS_COP.1(a), FCS_COP.1(b), FPT_TDC.1(c), FPT_TDC.1(e)

The following table provides a mapping of the IT objectives for the TOE operational environment to security functional requirements for the environment.

Non IT objectives for the TOE operational environment must be accomplished by procedural or administrative measures such that no SFRs are formulated for them in this ST.

Table 7-5: Mapping Objectives for the Operational Environment to SFRs for the Environment

Objective for the IT Environment	Security Functional Requirement for the Environment
OE.CRYPTO	FCS_CKM.1(a), FCS_CKM.1(b), FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), FCS_COP.1(g), FIA_UAU.1(b)
OE.DATA	FDP_ACC.1, FDP_ACF.1, FMT_MSA.1(a), FMT_MSA.1(b), FMT_MSA.3, FMT_SMF.1, FPT_SEP.1, FDP_RIP.1
OE.MODE	FMT_MSA.2
OE.OSLOGIN	FTA_TSE.1
OE.TIMESTAMP	FPT_STM.1
OE.USER	FIA_UAU.1(a), FIA_UID.1, FMT_SMR.1

7.2.2.1 O.KEYSTORE

The TOE must protect private keys stored in the keystore from unauthorized disclosure. In addition to that, the whole of the data stored in the keystore must be protected against undetected modification.

Before the keystore can be accessed, the password must be provided (FIA_UAU.2). This password is generated by the TOE (FIA_SOS.2); users are advised to only use TOE generated passwords as keystore passwords. Furthermore, a quality metric on the passwords is given by a password policy (this password policy cannot guarantee the same randomness of the passwords as the generated passwords) and a newly entered password is checked against that password policy (FIA_SOS.1).

Private keys are encrypted using TDEA before being stored in the keystore and are decrypted after retrieval. The password based encryption scheme PKCS #12 Password Based Encryption with SHA-1 and TDEA with three independent keys in CBC mode as described in PKCS #12 [PKCS#12] is used to generate a key from the given password (FPT_ETT_GSK.1(a)). A HMAC-SHA-1 hash is generated over all data records in the keystore using the password as the HMAC Key. This hash is stored in the keystore in a special record, called the header record, that exists once per keystore file. A HMAC-SHA-1 hash of the header using the password as the HMAC Key is also stored in the keystore, in the header record (FCS_COP.1(c), FPT_ETT_GSK.1(b)).

7.2.2.2 O.SECCHANNEL

The TOE shall offer a product of the TOE environment the functionality to communicate with a remote trusted IT product using a trusted channel that protects the confidentiality and integrity of user data being transmitted over this channel and provides the ability to authenticate the communication partners before transmitting user data that requires protection.

SSLv3 and TLSv1 are implemented to be used to set up a trusted channel which provides confidentiality and integrity protection of transferred data using cryptographic methods (FDP_UCT.1, FDP_UIT.1, FTP_ITC_PD-0108.1).

X.509 certificates as defined in [X.509] are needed for authentication during setup of the trusted channels. For the setup of the trusted channel, server authentication is mandatory and client authentication optional, meaning that it can be configured as mandatory. Furthermore, the channel is only set up if the communication partners can agree on a ciphersuite. The TOE can use LDAP (if this is provided by the TOE environment) to obtain a current CRL compliant to [X.509] during the SSL-/TLS handshake that it then uses for verification. Several well-know CA certificates are always contained in a newly created keystore and renewal certificates can be received during the setup of the SSL/TLS session (FDP_ACC.1, FDP_ACF.1, FMT_MSA.3, FPT_TDC.1(a) , FPT_TDC.1(b)).

The implementation of the cryptographic algorithms needed for the generation of symmetric keys and secrets for MAC computation, for the generation of asymmetric key pairs (FCS_CKM.1(a), FCS_CKM.1(b)), for the cryptographic operations and for the random number generation are provided by the ICC component (FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), FCS_COP.1(g)). All cryptographic algorithms given in the cipherspecs of the SSL/TLS ciphersuites and the random number generator are FIPS 140-2 [FIPS140-2] approved.

Certificates can be imported using API calls, the command line interface or during the setup of an SSL/TLS connection (FPT_TDC.1(a) , FPT_TDC.1(b)). In addition to that, certificates can be generated by the TOE itself by digitally signing a certificate for a public key such that this digital signature can afterwards be used for verifying the authenticity of that key. The algorithms for that are RSA/SHA-1. Certificate request compliant to PKCS #10 [RFC2986] can be generated and used to request the generation of certificates. These requests may be stored in and retrieved from the keystore (FCS_COP.1(f), FDP_DAU.1, FPT_TDC.1(b)). Certificates signed with RSA/MD2 or RSA/MD5 can be verified but are not generated (FCS_COP.1(g)).

7.2.2.3 O.SELFSEC

The TOE must implement self-tests for cryptographic modules and failure detection with preservation of a secure state for the trusted channel component

Self-tests for the cryptographic module ICC are implemented; they are called upon the first call to the SSL environment initialization command of the TOE. This is a function a product using the TOE has to call for the initialization of the SSL/TLS environment; this initialization is needed for setting up an SSL/TLS connection. Since the self-tests of the ICC component are requested when this function is called for the first time, these tests are conducted before the first use of the cryptographic algorithms supplied by the ICC for SSL/TLS (FPT_TST_WEAK.1).

Failures during the operation of the trusted channel are detected and a secure state is reached by either terminating the SSL/TLS session or the TOE (FPT_FLS.1).

7.2.2.4 O.SESSIONDATA

The TOE must protect secret TSF data (private/secret keys, session keys, and critical security parameters) of its individual trusted channel sessions and must ensure that no such data is made available to other trusted channel sessions also provided by the TOE.

The random number generator's state is reset when the module is unloaded and sensitive information in the buffers of a session is cleared before the buffer is released. Neither the buffer content nor the random number generator state are then any longer available (FDP_RIP.2(a), FDP_RIP.2(b)).

Cryptographic keys (symmetric keys and asymmetric key pairs) are destroyed when no longer used (FDP_RIP.2(b)).

7.2.2.5 O.TRANSFER

The TOE must be able to consistently interpret TSF data that is exchanged with entities in the IT environment via PKCS#11, PKCS#12.

This policy is modeled in FPT_TDC.1(c), which requires that the TOE is able to exchange public key certificates and private keys based on the PKCS#12 format, supported by cryptographic operations modeled in FCS_COP.1(a) and (b), and in FPT_TDC.1(e), which requires the TOE to be able to interpret public key certificates, private keys and security attributes exchanged via PKCS#11.

7.2.2.6 OE.CRYPTO

If the TOE is configured to use a hardware crypto module in the IT environment for the storage of cryptographic keys and certificates and/or the provision of cryptographic operations, the environment must provide such a hardware module accessible via PKCS#11. The hardware module must authenticate GSKit before granting access to stored key material.

The implementation of cryptographic primitives in the IT environment provided by hardware crypto modules includes key generation (FCS_CKM.1(a), FCS_CKM.1(b)) and cryptographic operations (FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), FCS_COP.1(g)). The objective of user authentication is met by FIA_UAU.1(b).

7.2.2.7 OE.DATA

The TOE environment (e.g. the operating system) must protect the TOE code from unauthorized modification and user data and TSF data from unauthorized disclosure, access, and modification. This includes also the protection of the TOE against other applications in the TOE environment.

The underlying operating system protecting the TOE provides access control mechanisms called out in FDP_ACC.1 and FDP_ACF.1, supported by FMT_MSA.1(a) and FMT_MSA.1(b), FMT_MSA.3 and the requirement for corresponding management functionality in FMT_SMF.1. Domain separation (FPT_SEP.1) and residual information protection (FDP_RIP.1) for data processed by the TOE using the resources provided by the operating system provide protection against interference of potentially hostile applications on the same machine.

7.2.2.8 OE.MODE

The TOE must be initialized to operate in FIPS mode and CC mode.

FMT_MSA.2 requires that only secure values are used to initialize the TOE.

7.2.2.9 OE.OSLOGIN

Remote login to the operating system must be disabled (by disabling all services that offer remote login).

The requirements of the FIPS policy for ICC are reflected in the login policy modeled in FTA_TSE.1.

7.2.2.10 OE.TIMESTAMP

The environment must provide reliable timestamps.

The provision of reliable time stamps by the OS is modeled in FPT_STM.1.

7.2.2.11 OE.USER

The environment must only allow authorized users to access the TOE.

The underlying system identifies and authenticates users (FIA_UAU.1(a), FIA_UID.1) and is able to relate user identities to different roles (FMT_SMR.1). Further access control mechanisms are implemented as described for OE.DATA.

7.2.3 Security Requirements Dependency Rationale

The following table shows the dependencies between the security functional requirements for the TOE and their resolution in this Security Target. SFRs in italic type setting show dependent SFRs that have not been resolved.

Table 7-7: Dependencies between TOE Security Functional Requirements

Security Functional Requirement	Dependencies	Resolved
FCS_CKM.1(a)	[*FCS_CKM.2 Cryptographic key distribution* or FCS_COP.1 Cryptographic operation: FCS_COP.1(a), FCS_COP.1(c)] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No

Security Functional Requirement	**Dependencies**	**Resolved**
FCS_CKM.1(b)	[*FCS_CKM.2 Cryptographic key distribution* or FCS_COP.1 Cryptographic operation: FCS_COP.1(d), FCS_COP.1(f), FCS_COP.1(g)] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(a)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(a) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(b)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation*] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(c)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(a) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(d)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(b) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(e)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation*] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No

Security Functional Requirement	**Dependencies**	**Resolved**
FCS_COP.1(f)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(b) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(g)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation]* *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FDP_ACC.1	FDP_ACF.1 Security attribute based access control	Yes
FDP_ACF.1	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialization	Yes
FDP_DAU.1	No dependencies	Yes
FDP_RIP.2(a)	No dependencies	Yes
FDP_RIP.2(b)	No dependencies	Yes
FDP_UCT.1	[FTP_ITC.1 Inter-TSF trusted channel: FTP_ITC_PD-0108.1, or *FTP_TRP.1 Trusted path*] [FDP_ACC.1 Subset access control, or *FDP_IFC.1 Subset information flow control*]	Yes
FDP_UIT.1	[FTP_ITC.1 Inter-TSF trusted channel: FTP_ITC_PD-0108.1, or *FTP_TRP.1 Trusted path*] [FDP_ACC.1 Subset access control, or *FDP_IFC.1 Subset information flow control*]	Yes
FIA_SOS.1	No dependencies	Yes
FIA_SOS.2	No dependencies	Yes
FIA_UAU.2	*FIA_UID.1 Timing of identification*	No

Security Functional Requirement	Dependencies	Resolved
FMT_MSA.3	*FMT_MSA.1 Management of security attributes* *FMT_SMR.1 Security roles*	No
FPT_ETT_GSK.1(a)	No dependencies	Yes
FPT_ETT_GSK.1(b)	No dependencies	Yes
FPT_FLS.1	ADV_SPM.1 Informal TOE security policy model	Yes
FPT_TDC.1 (a)	No dependencies	Yes
FPT_TDC.1 (b)	No dependencies	Yes
FPT_TDC.1 (c)	No dependencies	Yes
FPT_TDC.1 (d)	No dependencies	Yes
FPT_TDC.1 (e)	No dependencies	Yes
FPT_TST_WEAK.1	*FPT_AMT.1 Abstract machine testing*	No
FTP_ITC_PD-0108.1	No dependencies	Yes

For the TOE, dependencies to FCS_CKM.1, FCS_CKM.4, FDP_ITC.1, FIA_UID.1, FMT_MSA.1, FMT_MSA.2, FMT_SMR.1, and FPT_AMT.1 have not been resolved.

The dependencies of FCS_CKM.1, FCS_CKM.2 and FCS_COP.1 on FCS_CKM.4 (Cryptographic key destruction) have not been resolved since cryptographic session keys for the SSL session are protected by the TOE against unauthorized access and are destroyed by the object re-use functions of the TOE. Long living private keys of a public/private key pair will also be destroyed by the object re-use function of the TOE when they are kept in memory.

There are unresolved dependencies to FMT_MSA.2 from FCS_CKM.1(a), FCS_CKM.1(b), FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), and FCS_COP.1(g). All these SFRs describe functions that deal with cryptographic algorithms or the keystore. FMT_MSA.2 has to ensure that secure security attribute values are chosen. This applies to the initialization of the TOE in FIPS-mode and CC mode. The initialization of the TOE in FIPS mode and CC mode cannot be done by the TOE itself but must rather be done in the environment. So FMT_MSA.2 is a SFR for the TOE environment.

The unresolved dependencies from FCS_COP.1(b) to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4 come from the fact that FCS_COP.1(b) deals with digest generation and verification using SHA-1 or MD5. Since no cryptographic keys are needed for this cryptographic operation, the dependencies dealing with keys (FCS_CKM.1 and FCS_CKM.4) or import of user data without security attributes (FDP_ITC.1) cannot be applied.

FCS_COP.1(c) has an unresolved dependency to FCS_CKM.4. The reason for that is that the secret for MAC computation is not covered by key deletion but is cleared as part of FDP_RIP.2(b). So the dependency to FCS_CKM.4 does not apply here.

FCS_COP.1(e) has unresolved dependencies to SFRs concerning key generation, FCS_CKM.1, import of user data without security attributes (FDP_ITC.1), and key destruction, FCS_CKM.4. This is due to the fact that the random number generator (Universal Software Base TRN algorithm) needs no key. So the dependencies to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4 cannot be applied.

FCS_COP.1(g) has unresolved dependencies to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4. FCS_COP.1(g) deals with certificate verification using RSA with MD2/MD5. Since the TOE itself does not generate such certificate (but may need to verify them), no key generation (FCS_COP.1) is needed, furthermore, the import of user data without security attributes as given by FTP_ITC.1 is not used since certificates are security attributes for communication partners. Instead the import of such certificates is covered by FDP_CKM.3(b) and FTP_TDC.1. Key destruction as provided by FCS_CKM.4 is not needed since certificates only contain public keys, so no secure way to delete them is needed. For that reason, FCS_CKM.4 does not apply here.

There is an unresolved dependency from FIA_UAU.2 to FIA_UID.1 FIA_UAU.2 states that authentication is required before access to the keystore is granted, i.e. the password to access the keystore must be known. This authentication is not combined with identification since no identity is associated with the password used to protect the keystore. The TOE implements no knowledge of identities; it relies on the TOE environment to provide this where needed. So the TOE grants everyone who enters the correct password access to the keystore. The dependency to FIA_UID.1 cannot be applied for this reason.

The unresolved dependencies from FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 deal with user access and user roles. The TOE itself has no knowledge of user roles but relies on the environment to provide this. The TOE environment is responsible to restrict access to functions related to the management of cryptographic keys and certificates to a role Crypto-Officer. The TOE environment may also define additional roles (e. g. the role of a system administrator), but this is dependent on the specific requirements the TOE environment needs to satisfy. The definition of the roles is therefore left to the TOE environment and therefore dependencies to FMT_MSA.1 and FMT_SMR.1 are not resolved within the security functional requirements for the TOE but need to be resolved within the environment the TOE is integrated into (see the SFRs FMT_MSA.1(a), FMT_MSA.1(b), and FMT_SMR.1 of the TOE environment).

The dependency from FPT_TST_WEAK.1 to FPT_AMT.1 has not been resolved since the ICC component only tests itself and there are no assumptions concerning that matter in the TOE environment. So the dependency does not apply here.

The following table shows the dependencies between the different security functional requirements for the TOE environment and whether they are resolved in this Security Target. SFRs in italic type setting show dependent SFRs that have not been resolved.

Table 7-8: Dependencies between Security Functional Requirements for the TOE Environment

Security Functional Requirement	**Dependencies**	**Resolved**
FCS_CKM.1(a)	[*FCS_CKM.2 Cryptographic key distribution* or FCS_COP.1 Cryptographic operation: FCS_COP.1(a), FCS_COP.1(c)] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_CKM.1(b)	[*FCS_CKM.2 Cryptographic key distribution* or FCS_COP.1 Cryptographic operation: FCS_COP.1(d), FCS_COP.1(f), FCS_COP.1(g)] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(a)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(a) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(b)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation*] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No

Security Functional Requirement	**Dependencies**	**Resolved**
FCS_COP.1(c)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(a) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(d)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(b) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(e)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation*] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(f)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or FCS_CKM.1(b) Cryptographic key generation] *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No
FCS_COP.1(g)	[*FDP_ITC.1 Import of user data without security attributes* or *FDP_ITC.2 Import of user data with security attributes* or *FCS_CKM.1 Cryptographic key generation]* *FCS_CKM.4 Cryptographic key destruction* *FMT_MSA.2 Secure security attributes*	No

Security Functional Requirement	Dependencies	Resolved
FDP_ACC.1	FDP_ACF.1 Security attribute based access control	Yes
FDP_ACF.1	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation	Yes
FDP_RIP.1	No dependencies	Yes
FIA_UAU.1(a)	FIA_UID.1 Timing of identification	Yes
FIA_UAU.1(b)	FIA_UID.1 Timing of identification	No
FIA_UID.1	No dependencies	Yes
FMT_MSA.1(a)	[FDP_ACC.1 Subset access control or *FDP_IFC.1 Subset information flow control*] FMT_SMF.1 Specification of management functions FMT_SMR.1 Security roles	Yes
FMT_MSA.1(b)	[FDP_ACC.1 Subset access control or *FDP_IFC.1 Subset information flow control*] FMT_SMF.1 Specification of management functions FMT_SMR.1 Security roles	Yes
FMT_MSA.2	*ADV_SPM.1 Informal TOE security policy model* [FDP_ACC.1 Subset access control or *FDP_IFC.1 Subset information flow control*] FMT_MSA.1(b) Management of security attributes FMT_SMR.1 Security roles	No
FMT_MSA.3	FMT_MSA.1 Management of security attributes: FMT_MSA.1(a) and FMT_MSA.1(b) FMT_SMR.1 Security roles	Yes
FMT_SMF.1	No dependencies	Yes
FMT_SMR.1	FIA_UID.1 Timing of identification	Yes
FPT_SEP.1	No dependencies	Yes
FPT_STM.1	No dependencies	Yes
FTA_TSE.1	No dependencies	Yes

The dependencies of FCS_CKM.1 and FCS_COP.1 on FCS_CKM.4 (Cryptographic key destruction) have not been resolved since cryptographic session keys for the SSL session are protected by the IT environment against unauthorized access. Long living private keys of a public/private key pair will also be protected by the IT environment.

There are unresolved dependencies to FMT_MSA.2 from FCS_CKM.1(a), FCS_CKM.1(b), FCS_COP.1(a), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e), FCS_COP.1(f), and FCS_COP.1(g). All these SFRs describe functions that deal with cryptographic algorithms or the key material stored in a hardware crypto module. FMT_MSA.2 has to ensure that secure security attribute values are chosen. This is achieved by the organizational requirement in OE.FIPS140 to use only FIPS 140-validated crypto modules.

The unresolved dependencies from FCS_COP.1(b) to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4 come from the fact that FCS_COP.1(b) deals with digest generation and verification using SHA-1 or MD5. Since no cryptographic keys are needed for this cryptographic operation, the dependencies dealing with keys (FCS_CKM.1 and FCS_CKM.4) or import of user data without security attributes (FDP_ITC.1) cannot be applied.

FCS_COP.1(e) has unresolved dependencies to SFRs concerning key generation, FCS_CKM.1, import of user data without security attributes (FDP_ITC.1), and key destruction, FCS_CKM.4. This is due to the fact that the random number generator (Universal Software Base TRN algorithm) needs no key. So the dependencies to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4 cannot be applied.

FCS_COP.1(g) has unresolved dependencies to FCS_CKM.1, FDP_ITC.1, and FCS_CKM.4. FCS_COP.1(g) deals with certificate verification using RSA with MD2/MD5. Since the IT environment itself does not generate such certificate (but provides support to the TOE for verifying them), no key generation (FCS_COP.1) is needed, furthermore, the import of user data without security attributes as given by FTP_ITC.1 is not used since certificates are security attributes for communication partners.

The dependency from FMT_MSA.2 to ADV_SPM.1 demands the definition of an informal security policy model. Since the values that are to be set for the security attributes are clearly defined (FIPS mode and CC mode for the TOE) and it is also described why they are to be chosen (see [TARGET] chapter 2), this dependency need not to be applied (see also [CC] appendix H.2 paragraph 1015).

The dependency of FIA_UAU.1(b) on FIA_UID.1 is not resolved, since the cryptographic module in the IT environment uses a PIN for the authentication of the TOE without requiring additional identification from its users.

7.2.4 Appropriateness of TOE security assurance requirements

The chosen EAL, EAL4, is sufficient to address the security problems defined in chapter 3 and able to support the security objectives stated in chapter 4. The TOE does not counter any direct threats, but implements a security policy derived from organizational demands. The choice of EAL4 has in consequence been driven chiefly by consumer demands – the TOE is a component to be used in other IBM products which are typically evaluated at EAL3 or EAL4.

The explicitly defined security functional requirements do not introduce dependencies on assurance requirements that are not covered by the chosen EAL.

7.3 Strength of function Rationale

The following functions that use probabilistic or permutational mechanisms were identified:

- Cryptographic support (FCS):
  - key generation of symmetric keys and secrets for MAC computation (TDEA, AES, HMAC SHA-1)
  - key generation of asymmetric key pairs (RSA)
  - symmetric encryption/decryption (TDEA, AES),
  - asymmetric encryption/decryption (RSA)
  - digest generation (SHA-1, MD5)
  - data authentication (RSA and SHA-1, HMAC SHA-1)
  - digital signature generation/verification (RSA with SHA-1, SHA-2, MD2 or MD5, DSA with SHA-1)
  - random number generation (Universal Software Base TRNG algorithm)
- Protection of the TSF (FPT):
  - confidentiality protection of private keys prior to storing them in the keystore
  - integrity protection of private keys and certificate/private key prior to storing them in the keystore

The strength of cryptographic algorithms is outside the scope of CC ([CEMv1] ASE_REQ.1-15, paragraph 424: *The strength of cryptographic algorithms is outside the scope of the CC*), so no SOF-claims are made for key generation of symmetric keys and secrets for MAC computation (TDEA, AES, HMAC SHA-1), key generation of asymmetric key pairs (RSA), symmetric encryption/decryption (TDEA, AES), asymmetric encryption/decryption (RSA), digest generation (SHA-1, MD5), data authentication (RSA and SHA-1, HMAC SHA-1), digital signature generation/verification (RSA with SHA-1, SHA-2, MD2 or MD5, DSA with SHA-1), random number generation (Universal Software Base TRNG algorithm), the cryptographic functions used for confidentiality protection of private keys prior to storing them in the keystore, and the cryptographic functions used for integrity protection of private keys and certificate/private key prior storing them in the keystore.

For the encryption of private keys before storing them in the keystore (confidentiality protection) and for the generation of a hash over all data records of the keystore (integrity protection), a password is used. On this password, SOF-high is claimed. The password must be entered before the keystore can be accessed and is used for the generation of cryptographic keys used to protect key data in the keystore.

The corresponding SFRs are FIA_UAU.2 (access to the keystore), FPT_ETT_GSK.1(a) (encryption of private keys in the keystore), FPT_ETT_GSK.1(b) and FCS_COP.1(c) (a HMAC-SHA-1 hash over all data records in the keystore), FIA_SOS.2 (generation of passwords), and FIA_SOS.1 (password strength).

The corresponding parts of the TSS are KEYMAN.3 (password protection of the keystore), KEYMAN.4 (protection of key data), and KEYMAN.5 (generation and quality of the passwords).

The claim of SOF-high on the password quality is considered appropriate since the password must be a password generated by the TOE. Such passwords are very hard to guess since they are random, consist of alphanumeric and punctuation characters, and have a length of at least 14 characters.

Please note that the cryptographic module, ICC, is FIPS 140-2 level 1 [FIPS140-2] validated and that the TOE in the evaluated configuration uses only algorithms that are FIPS 140-2 [FIPS140-2] approved for the cipherspec part of the SSL/TLS ciphersuites.

8 Abbreviations

AES	Advanced Encryption Standard
AIX	Advanced Interactive Executive
ANSI	American National Standards Institute
API	Application Programming Interface
CC	Common Criteria
CD	Compact Disc
CLI	Command Line Interface
CRL	Certificate Revocation List
DES	Data Encryption Standard
EE	End-entiry
FIPS	Federal Information Processing Standard
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISO	International Standards Organization
OSP	Organizational Security Policy
PDF	Portable Data Format
PP	Protection Profile
RSA	Rivest-Shamir-Adleman
SHA	Secure Hash Algorithm
SFR	Security Functional Requirement
SOF	Strength of Function
SSL	Secure Sockets Layer
ST	Security Target
TDEA	Triple Data Encryption Algorithm
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOE	Target of Evaluation
TRNG	True random number generator
TSF	TOE Security Functions