Trusted Computing Platform Alliance (TCPA)
Trusted Platform Module
Protection Profile

Version 1.9.4

March 15, 2002

Prepared for: TCPA Membership

Foreword

This PP is written to support the development of trusted subsystems that may be integrated into a computing platform.

The base set of requirements used in this protection profile is taken from the “Common Criteria for Information Technology Security Evaluations, Version 2.1.

Comments on this PP should be sent to the Trusted Computing Platform Alliance at http://www.trustedpc.org/home/contact.htm.

Revision History

Version	Date	Description
0.5	January 17, 2001	Revision by CygnaCom Solutions under contract with the TCPA. Added requirements FPT_RVM.1, and augmented EAL 3 with AVA_VLA.2 and ALC_FLR.1. “Terminology” describing the security attributes was moved to TOE Description.
0.52	February 1, 2001	Reformatted sections 1 and 2: section 1 is only about the PP. Section 2 describes the TOE. Section 2.1 was modified to be more general; the previous version appeared to mandate certain hardware components. Additions and deletions of text are summarized in Attachment.
0.53	February 19, 2001	Deleted objectives that mapped only to assurance requirements or that were very general in nature. Modified mappings. Added text to Section 2 to further explain the function of the TOE.
0.54	February 21, 2001	Added a glossary. Added changes and suggestions from TCPA conference call.
0.55	March 5, 2001	Added explanatory diagrams. Cleaned up mapping. Added necessary details from the TCPA Specification.
0.56	April 11, 2001	Made corrections in response to reviews by working group members.
1.0	April 23, 2001	Final revision prior to submission for evaluation.
1.1	June 13, 2001	Revisions made to synchronize with TCPA Main Specification Version 1.1, RC 2.
1.2	August 16, 2001	Revisions made to synchronize with new version of TCPA Main Specification Version 1.1.
1.3	September 14, 2001	Updated PP to break out optional requirements and to correct errors in mapping.
1.4	September 24, 2001	Updated PP to provide more detailed refinements in functional requirements and more definition of TSF data, including security attributes.
1.5	October 15, 2001	Updated PP to delete audit requirements, augmentations to EAL3 assurance requirements and to delete T.SpecRef and O.SpecRef.
1.6	November 8, 2001	Updated PP to augment assurance requirements with testing specifications and to delete optional command annexes. Edited sections 1 and 2 to be more succinct.
1.7	January 7, 2002	Updated PP in response to Evaluator EORs.
1.8.2	February 11, 2002	Updated PP in response to Evaluator EORs.
1.9	February 22, 2002	Updated PP in response to Evaluator/Validator comments.
1.9.1	February 22, 2002	Updated PP in response to Evaluator comments.
1.9.2	February 26,2002	Updated PP in response to Evaluator/Validator comments.

1.9.3	March 9, 2002	Updated PP in response to Evaluator/Validator comments. Replaced AVA_VLA.2 with AVA_VLA.1 due to dependencies.
1.9.4	March 15, 2002	Updated PP in response to TCPA comment and upon Evaluator and Validator approval

Table of Contents

Page

List of Tables

Page

1 Introduction

This section contains document management and overview information. The PP Identification provides the labeling and descriptive information necessary to identify, catalogue, register, and cross-reference a PP. The PP Overview summarizes the profile in narrative form and provides sufficient information for a potential user to determine whether the PP is of interest. The overview can also be used as a standalone abstract for PP catalogues and registers.

1.1 Identification

Title: Trusted Computing Platform Alliance (TCPA) Trusted Platform Module Protection Profile (TPM PP)

Assurance Level: The assurance level for this protection profile is EAL3, augmented. The strength of function is SOF Basic.

Version Number: Version 1.9.4

Date: March 15, 2002

Sponsoring Organization: Trusted Computing Platform Alliance (TCPA)

Registration: <To be filled in upon registration>

Keywords: Smartcard, Trusted Platform Module, RSA

1.2 Protection Profile Overview

This PP describes the IT security requirements for a security module known as the Trusted Platform Module (TPM). The TPM provides security primitives in a secure environment. The primitives include digital signatures, random number generation, protected storage and binding information to the TPM. The TCPA TPM is described in detail in the TCPA Main Specification.

1.3 Related Documents

- Trusted Computing Platform Alliance (TCPA) Main Specification
- International Standard ISO/IEC 15408 Information technology — Security techniques — Evaluation criteria for IT security
- Common Methodology for Information Security Evaluation (CEM) Version 1.0, August 1999

1.4 PP Organization

The main sections of the PP are the TOE (target of evaluation) Description, TOE Security Environment, Security Objectives, IT Security Requirements, and Rationale.

Section 2, the TOE Description, provides general information about the TOE, serves as an aid to understanding its security requirements, and provides context for the PP's evaluation.

The TOE Security Environment in Section 3 describes security aspects of the environment in which the TOE is to be used and the manner in which it is to be employed. The TOE security environment includes:

a) Assumptions regarding the TOE's intended usage and environment of use

b) Threats relevant to secure TOE operation

c) Organizational security policies with which the TOE must comply

Section 4 contains the security objectives that reflect the stated intent of the PP. The objectives define how the TOE will counter identified threats and how it will cover identified organizational security policies and assumptions. Each security objective is categorized as being for the TOE or for the environment.

Section 5 contains the applicable Security Requirements taken from the Common Criteria, with appropriate refinements. The requirements are provided in separate subsections for the TOE and its environment. The IT security requirements are subdivided as follows:

a) TOE Security Functional Requirements

b) TOE Security Assurance Requirements

The Rationale in Section 6 presents evidence that the PP is a complete and cohesive set of requirements and that a conformant TOE would provide an effective set of IT security countermeasures within the security environment. The Rationale is in two main parts. First, a Security Objectives Rationale demonstrates that the stated security objectives are traceable to all of the aspects identified in the TOE security environment and are suitable to cover them. Then, a Security Requirements Rationale demonstrates that the security requirements (TOE and environment) are traceable to the security objectives and are suitable to meet them.

A glossary of acronyms and terms used in the protection profile (PP) is provided in the Appendix.

1.5 Common Criteria Conformance

This PP has been built with Common Criteria (CC) Version 2.1 (ISO/IEC 15408 Evaluation Criteria for Information Technology Security; Part 1: Introduction and general model, Part 2: Security functional requirements, and Part 3: Security assurance requirements).

The TCPA TPM PP is conformant with Common Criteria Version 2.1, Part 2, and Part 3 (Evaluation Assurance Level 3 with augmentation).

2 TOE Description

2.1 Overview

The target of evaluation (TOE) within this PP is the hardware, software, and firmware that comprise the TPM. The TPM may include, at the option of the manufacturer, integrated circuits, operating system software and/or firmware. The security requirements in this protection profile apply to the TPM.

2.2 Definition of TOE

The TPM is a collection of hardware, firmware and/or software that support the following protocols and algorithms:

- Algorithms: RSA, SHA-1, HMAC
- Random number generation
- Key generation
- Self Tests

The TPM may be used to provide secure storage for an unlimited number of private keys or other data by using RSA key technology to encrypt data and keys. The resulting encrypted file, which contains header information in addition to the data or key, is called a blob and is output by the TPM and can be loaded in the TPM when needed. The functionality of the TPM can also be used so that private keys generated on the TPM can be stored outside the TPM (encrypted) in a way that allows the TPM to use them later without ever exposing such keys in the clear outside the TPM.

The functionality used to provide secure storage is:

- Seal and Unseal, which perform RSA encrypt and decrypt, respectively, on data that is externally generated. The sealing operation encrypts not only the data, but also the platform configuration values that are stored in the platform configuration registers (PCRs) in the TPM and tpmProof, which is a unique identifier for that TPM. To unseal the data, three conditions must exist: 1) the appropriate key must be available for unseal, 2) the TPM PCRs must contain the same values that existed at the time of the seal operation, and 3) the value of tpmProof must be the same as that encrypted during the seal operation. By requiring the PCR values to be duplicated at unseal and the tpmProof value to be checked, the seal operation allows software to explicitly state the future “trusted” configuration that the platform must be in for the decrypted key to be used and for decrypt to only occur on the specified TPM.
- Unbind, which RSA decrypts a blob created outside the TPM that has been encrypted using a public key where the associated private key is stored in the TPM.

A number of key types are defined within the TPM. Keys may be migratable or non-migratable. A migratable key is a key that may be transported outside the specific TPM. A non-migratable key is a key that cannot be transported outside a specific TPM. Key types include:

- The Storage Root key (SRK), which is the root key of a hierarchy of keys associated with a TPM; it is generated within a TPM and is a non-migratable key.

Each TPM contains a SRK, generated by the TPM at the request of the Owner. Under that SRK are two trees: one dealing with migratable data and the other dealing with non-migratable data

- Signing Keys, which must be a leaf of the Storage Root Key hierarchy. The private key of the key pair is used for signing operations only.
- Storage keys, which are used to RSA encrypt and RSA decrypt other keys in the Protected Storage hierarchy, only.
- Identity Keys, which are used for operations that require a TPM identity, only.
- Binding Keys, which are used for TPM_Unbind operations only. A bind operation (performed outside the TPM) associates identification and authentication data with a particular data set and the entire data blob is encrypted outside the TPM using a binding key, which is an RSA key. The TPM_Unbind operation uses a private key stored in the TPM to decrypt the blob so that the data (often a key pair) stored in the blob may be used.
- The Endorsement key pair, which is an asymmetric key pair generated by or inserted in a TPM that is used as proof that a TPM is a genuine TPM.

Each TPM is identified and validated by its Endorsement Key. A TPM has only one endorsement key pair. The Endorsement Key is transitively bound to the Platform via the TPM as follows:

1. An Endorsement Key is bound to one and only one TPM (i.e., that is a one to one correspondence between an Endorsement Key and a TPM.)
2. A TPM is bound to one and only one Platform, (i.e., there is a one to one correspondence between a TPM and a Platform.)
3. Therefore, an Endorsement Key is bound to a Platform, (i.e., there is a one to one correspondence between an Endorsement Key and a Platform.

TPM algorithms, protocols, identification and authentication, and access control functions are described in the subsections below.

2.2.1 Algorithms

The TPM supports the RSA algorithm and must use the RSA algorithm for encryption and digital signatures. The TPM supports RSA key sizes of 512, 1024, and 2048 bits. The RSA public exponent must be e, where $e = 2^{16}+1$. TPM devices that use the Chinese Remainder Theorem (CRT) as the RSA implementation must provide protection and detection of failures during the CRT process to avoid attacks on the private key. All TPM Storage keys are of strength equivalent to a 2048 bit RSA key or greater. The TPM does not load a Storage key whose strength is less than that of a 2048 bit RSA key. All TPM identity keys are of a strength equivalent to a 2048 bit RSA key or greater.

The TPM supports the Secure Hash Algorithm (SHA) -1 hash algorithm as defined by United States Federal Information Processing Standard 180-1. The output of SHA-1 is 160 bits and all areas that expect a hash value are required to support the full 160 bits. A SHA-1 digest is used in the early stages of a boot process, before more sophisticated computing resources are available. Secure Hash is also used in the process of preparing data for signature or signature verification.

The TPM uses the RSA algorithm for signature and verification operations. The TPM must use PKCS #1 V2 for the format and design of the signature output.

2.2.2 Random Number Generator (RNG)

The RNG capability is only accessible to valid TPM commands. Intermediate results from the RNG are not available to any user. When the data is for internal use by the TPM (e.g., asymmetric key generation) the data is held in a shielded location and is not accessible to any user.

2.2.3 Key Generation

The TPM generates asymmetric key pairs. The generate function is a protected capability and the private key is held in a shielded location.

The TPM generates the HMAC key by taking the next *n* bits from the TPM RNG.

The creation of all nonce values use the next *n* bits from the TPM RNG.

2.2.4 Self Tests

The TPM provides startup self-tests and a mechanism to allow the self-tests to be run on demand. The response from the self-tests is pass or fail. Self-tests include checks of the following:

- RNG functionality, as defined by United Stated Federal Information Processing Standard 140-1.
- Reading and extending the integrity registers. The self-test for the integrity registers will leave the integrity registers in a known state.
- Endorsement key pair integrity, if the key pair exists. This test will verify that the Endorsement key pair can sign and verify a known value. This test will also test the RSA sign and verify engine. If the Endorsement key has not yet been generated, the TPM action is manufacturer specific.
- Integrity of the protected capabilities of the TPM. This consists of checks that ensure that the TPM “microcode” or equivalent has not changed.
- Any tamper-resistance markers. The tests on the tamper-resistance or tamper-evident markers are under programmable control. There is no requirement to check tamper-evident tape or the status of epoxy surrounding the case.

When the TPM detects a failure during any self-test, the part experiencing the failure will enter a shutdown mode and an error code is returned. Registers are written with a fixed set of data as defined by the manufacturer and described in the applicable Security Target.

2.2.5 Identification and Authentication

The TPM identification and authentication capability is used to authenticate an entity owner and to authorize use of an entity. The basic premise is to prove knowledge of a shared secret. This shared secret is the identification and authentication data. The TCPA Specification calls the identification and authentication process and this data authorization.

The identification and authentication data for the TPM Owner and the owner of the Storage Root Key are held within the TPM itself. The identification and authentication data for other owners of entities are held and protected with the entity.

The identification and authentication protocols use a random nonce. This requires that a nonce from one side be in use only for a message and its reply. For instance, the TPM would create a nonce and send that on a reply. The requestor would receive that nonce and then include it in the next request. The TPM would validate that the correct nonce was in the request and then create a new nonce for the reply. This mechanism is in place to prevent replay attacks and man-in-the-middle attacks.

2.2.6 Access Control

Access control is enforced in the TPM on all data and operations performed on that data. The TPM provides access control by denying access to some data and operations and allowing access to other data and operations based on the value of the TCPA_AUTH_DATA_USAGE flag TCPA_KEY_FLAGS and the TCPA_KEY_USAGE flag. The TCPA_AUTH_DATA_USAGE flag defines access as either owner or world. Owner must be authenticated with a shared secret as described in Section 2.2.5, above. World means that usage of the key is permitted by anyone without authentication. The TCPA_KEY_FLAGS define whether a key is migratable or non-migratable and whether the key is stored in volatile storage and must be unloaded at TPM startup. The TCPA_KEY_USAGE flag identifies the key type, as defined in Section 2.2 above. Depending on the key type, certain operations may or may not be allowed using the particular key, as described above.

Upon appropriate identification and authentication associated with the keys, users can use the key for the purposes permitted by the TCPA_KEY_USAGE flag.

2.3 Security Attributes and Data

All data, including user key pairs, user data, and TSF data, have associated security attributes, stored as flags in the TPM or associated with the data in an encrypted blob. The following security attributes are defined:

- Migration attribute, which determines if the data (or key pair) can migrate from one TPM to another. This security attribute is stored in TCPA_KEY_FLAGS.
- TCPA_AUTHDATA_USAGE flag is used to define whether the data can be access only by the owner or by the world.
- Attribute key type, stored in TCPA_ KEY_USAGE, which indicates if the data is a key or key pair and the type of key (e.g., storage, binding, etc., as defined in section 2.2, above).
- Volatility attribute, which defines whether the data must be stored in volatile or non-volatile storage and if it is cleared at TPM startup. This security attribute is stored in TCPA_KEY_FLAGS.

Within the TPM, for the purposes of Common Criteria evaluation, TSF data is defined as:

- The Endorsement Key Pair,
- The Storage Root Key (SRK),
- TPMProof, i.e., the random number (nonce) that each TPM maintains to validate that the data originated at this TPM.
- PCR values,
- TPM Identity Keys,

- TPM Owner Key,
- TPM owner identification and authentication data,
- Entity owner identification and authentication data,
- Migration authorization data, which is used in creating migratable key blobs,
- Security attributes as defined above.

User data is defined as all user keys and other data that may be passed to the TPM for signature, decryption, etc.

3 TOE Security Environment

3.1 Secure Usage Assumptions

TOE secure usage assumptions are defined in Table 3.1, below.

Table 3.1 – Secure Usage Assumptions

#	Assumption	Description
1	A.Configuration	The TOE will be properly installed and configured.

Table 3.2 lists the Secure Usage Assumptions for the IT environment.

Table 3.2 – Assumptions for the IT Environment

#	Assumption Name	Description
1	AE.Physical_Protection	The TOE provides tamper evidence only. It provides no protection against physical threats such as simple power analysis, differential power analysis, external signals, or extreme temperature. Physical protection is assumed to be provided by the environment.

3.2 Threats to Security

Threats to the TOE are defined in Table 3.3, below. The asset under attack is the information transiting the TOE. In general, the threat agent includes, but is not limited to: 1) people with TOE access who are expected to possess “average” expertise, few resources, and moderate motivation, or 2) failure of the TOE.

Table 3.3 – Threats

#	Threat	Description
1	T.Attack	An undetected compromise of the cryptography-related IT assets may occur as a result of an attacker (whether an insider or outsider) attempting to perform actions that the individual is not authorised to perform.
2	T.Bypass	An unauthorized individual or user may tamper with security attributes or other data in order to bypass TOE security functions and gain unauthorized access to TOE assets.
3	T.Export	A user or an attacker may export data without security attributes or with unsecure security attributes, causing the data exported to be erroneous and unusable, to allow erroneous data to be added or substituted for the original data, and/or to reveal secrets.
4	T.Hack_Crypto	Cryptographic algorithms may be incorrectly implemented, allowing an unauthorized individual or user to decipher keys generated within the TPM and thereby gain unauthorised access to encrypted data.
5	T.Hack_Physical	An unauthorised individual or user of the TOE may cause unauthorised disclosure or modification of TOE assets by physically interacting with the TOE to exploit vulnerabilities in the physical environment.
6	T.Imperson	An unauthorized individual may impersonate an authorised user of the TOE and thereby gain access to TOE data, keys, and operations.
7	T.Import	A user or attacker may import data or keys without security attributes or with erroneous security attributes, causing key ownership and authorization to be uncertain or erroneous and the system to malfunction or operate in an unsecure manner.

Table 3.3 (concluded)

#	Threat	Description
8	T.Key_Gen_Destroy	Cryptographic keys may be generated or destroyed in an unsecure manner, causing key compromise.
9	T.Malfunction	TOE assets may be modified or disclosed to an unauthorised individual or user of the TOE, through malfunction of the TOE.
10	T.Modify	An attacker may modify TSF or user data, e.g., stored security attributes or keys, in order to gain access to the TOE and its assets.
11	T. Object_Attr_Default	A user may create an object with no security attribute values.
12	T.Object_Attr_Change	A user or attacker may make unauthorized changes to security attribute values for an object.
13	T.Object_SecureValues	A user may set unsecure values for object security attributes.
14	T.Residual_Info	A user may obtain information that the user is not authorized to have when the data is no longer actively managed by the TOE (“data scavenging”).
15	T.Replay	An unauthorized individual may gain access to the system and sensitive data through a “replay” or “man-in-the-middle” attack that allows the individual to capture identification and authentication data.
16	T.Repudiate_Transact	An originator of data may deny originating the data to avoid accountability.
17	T.Test	The TOE may start-up in an unsecure state or enter an unsecure state, allowing an attacker to obtain sensitive data or compromise the system.

4 Security Objectives

4.1 Security Objectives for the TOE

TOE security objectives are defined in Table 4.1, below.

Table 4.1 – Security Objectives for the TOE

#	Objective	Description
1	O.Crypto_Key_Man	The TOE shall generate and destroy cryptographic keys in a secure manner.
2	O.Crypto_Op	The TOE shall perform cryptographic operations, including secure hash, HMAC, RSA digital signature and signature verification, RSA encryption and decryption, and RSA key generation in accordance with specified algorithms and key size; key size must be sufficient size to protect private/public key pairs from deciphering.
3	O.Crypto_Self_Test	The TOE shall provide the ability to verify that the cryptographic functions operate as designed.
4	O.DAC	The TOE shall control and restrict user access to the TOE assets in accordance with a specified access control policy.
5	O.Export	When data are exported outside the TPM, the TOE shall ensure that the data security attributes being exported are unambiguously associated with the data.
6	O.Fail_Secure	The TOE shall preserve the secure state of the system in the event of a cryptographic or other failure.
7	O.General_Integ_Checks	The TOE shall provide periodic checks on system integrity and user data integrity.
8	O.HMAC	The TOE shall provide the ability to detect the modification of security attributes and other data.
9	O.I&A	The TOE shall uniquely identify all users, and shall authenticate the claimed identify before granting a user access to the TOE facilities.
10	O.Import	When data are being imported into the TOE, the TOE shall ensure that the data security attributes are being imported with the data and the data is from authorized source. In addition, the TOE shall verify those security attributes according to the TSF access control rules.
11	O.Invoke	The TSF shall be invoked for all actions.
12	O.Limit_Actions_Auth	The TOE shall restrict the actions a user may perform before the TOE verifies the identity of the user.
13	O.MessageNR	The TOE shall provide user data integrity, source authentication, and the basis for source non-repudiation when exchanging data with a remote system.

Table 4.1 (Concluded)

#	Objective	Description
14	O.No_Residual_Info	The TOE shall ensure there is no “object reuse,” i.e., ensure that there is no residual information in information containers or system resources upon their reallocation to different users.
15	O.Object_Attr_Default	The TOE shall require default security attributes for the object when the object is created.
16	O.Object_Attr_DefaultOver	The TOE shall permit authorised users to override defaulted values for security attributes for an object.
17	O.Obj_Attr_SecureValues	The TOE shall maintain object security attributes by permitting only secure values.
18	O.Security_Attr_Mgt	The TOE shall allow only authorised users to initialise and change object security attributes.
19	O.Security_Roles	The TOE shall maintain security-relevant roles and association of users with those roles.
20	O.Self_Protect	The TSF will maintain a domain for its own execution that protects it and its resources from external interference, tampering, or unauthorized disclosure.
20	O.Single_Auth	The TOE shall provide a single use authentication mechanism and require re-authentication to prevent “replay” and “man-in-the-middle” attacks.
21	O.Tamper_ID	The TOE shall provide features that permit a human to detect physical tampering of a system component.

4.2 Security Objectives for the Environment

Table 4.2 lists security objectives for the environment.

Table 4.2 – Security Objectives for the Environment

	Objective Name	Objective Description
1	OE.Configuration	The TOE shall be installed and configured properly for starting up the TOE in a secure state.
2	OE.PhysSecurity	The environment shall provide an acceptable level of physical security so that the TPM cannot be tampered with or be subject to side channel attacks such as the various forms of power analysis and timing analysis.

5 IT Security Requirements

5.1 Introduction

This section defines the TOE security functional requirements and assurance requirements. All requirements are from the CC Parts 2 and 3. Selections, assignments, and refinements are indicated by *italics*.

5.2 TOE Security Functional Requirements

This section defines the TOE security functional requirements. A list of the requirements is provided in Table 5.1. The full text of the security functional requirements is contained below. Certain security functional requirements have multiple iterations in the text. Iterations are indicated by the user of a ":" in the component identification and by a ";" in the component name.

Table 5.1 – TOE Security Functional Requirements

#	Functional Requirement	Title
1	FCO_NRO.2	Enforced proof of origin
2	FCS_CKM.1	Cryptographic key generation
3	FCS_CKM.4	Cryptographic key destruction
4	FCS_COP.1	Cryptographic operation
5	FDP_ACC.1	Subset access control
6	FDP_ACF.1	Security attribute based access control
7	FDP_ETC.2	Export of user data with security attributes
8	FDP_ITC.2	Import of user data with security attributes
9	FDP_RIP.2	Full residual information protection
10	FIA_ATD.1	User attribute definition
11	FIA_UAU.1	Timing of authentication
12	FIA_UAU.4	Single-use authentication mechanism
13	FIA_UAU.6	Re-authenticating
14	FIA_UID.1	Timing of identification
15	FMT_MOF.1	Management of security functions behaviour
16	FMT_MSA.1	Management of security attributes
17	FMT_MSA.2	Secure security attributes
18	FMT_MSA.3	Static attribute initialisation
19	FMT_MTD.1	Management of TSF data

Table 5.1 – (concluded)

#	Functional Requirement	Title
20	FMT_SMR.2	Restrictions on security roles
21	FPT_AMT.1	Abstract machine testing
22	FPT_FLS.1	Failure with preservation of secure state
23	FPT_PHP.1	Passive detection of physical attack
24	FPT_RCV.4	Function recovery
25	FPT_RPL.1	Replay detection
26	FPT_RVM.1	Non-bypassability of the TSP
27	FPT_SEP.1	TSF domain separation
28	FPT_TDC.1	Inter-TSF basic TSF data consistency
29	FPT_TST.1	TSF testing
30	FTP_TRP.1	Trusted path

5.2.1 Class FCO – Communication

FCO_NRO.2 Enforced proof of origin

Hierarchical to: FCO_NRO.1

FCO_NRO.2.1 The TSF shall enforce the generation of evidence of origin for transmitted *TPM data signed using identity keys* at all times.

FCO_NRO.2.2 The TSF shall be able to relate the *identity* of the originator of the information, and the *TPM data* of the information to which the evidence applies.

FCO_NRO.2.3 The TSF shall provide a capability to verify the evidence of origin of information to *recipient* given *evidence only available when requestor properly authenticates*.

Dependencies: FIA_UID.1 Timing of identification

5.2.2 Class FCS – Cryptographic Support

FCS_CKM.1 Cryptographic key generation

Hierarchical to: No other components.

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm *RSA* and specified cryptographic key sizes *RSA 512, 1024, 2048* that meet the following: *PKCS#1 V2*.

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

FCS_CKM.4 Cryptographic key destruction

Hierarchical to: No other components.

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method *erasure of memory areas containing cryptographic keys* that meets the following: *FIPS 140-1, Section 4.8.5, Key Destruction, or equivalent*.

Dependencies: FCS_CKM.1 Cryptographic key generation, FMT_MSA.2 Secure security attributes

FCS_COP.1:1 Cryptographic operation; RSA encrypt and decrypt

Hierarchical to: No other components.

FCS_COP.1.1;1 The TSF shall perform *encryption and decryption* in accordance with a specified cryptographic algorithm *RSA* and cryptographic key sizes *RSA 512, 1024, 2048* that meet the following: *PKCS#1 V2*.

FCS_COP.1:2 Cryptographic operation; RSA signature and signature verification

Hierarchical to: No other components.

FCS_COP.1.1;2 The TSF shall perform *signature generation and signature verification* in accordance with a specified cryptographic algorithm *RSA* and cryptographic key sizes *RSA 512, 1024, 2048* that meet the following: *PKCS#1 V2*.

FCS_COP.1:3 Cryptographic operation; SHA

Hierarchical to: No other components.

FCS_COP.1.1; 3 The TSF shall perform *secure hash* in accordance with a specified cryptographic algorithm *SHA-1* and cryptographic key sizes *not applicable* that meet the following: *FIPS 180-1*.

FCS_COP.1:4 Cryptographic operation; Keyed-Hashing for Message Authentication

Hierarchical to: No other components.

FCS_COP.1.1; 4 The TSF shall perform *keyed-hashing message authentication code (HMAC)* in accordance with a specified cryptographic algorithm *SHA-1* and cryptographic key sizes *160 bits* that meet the following: *RFC 2104*.

Dependencies: FCS_CKM.1 Cryptographic key generation, FCS_CKM.4 Cryptographic key destruction, FMT_MSA.2 Secure security attributes

5.2.3 Class FDP – User Data Protection

FDP_ACC.1 Subset access control

Hierarchical to: No other components.

FDP_ACC.1.1 The TSF shall enforce the *Protected Operations Access Controls* on

a) Subjects: commands executing on behalf of users.

b) Objects: keys and user data.

c) Operations: signature generation, encryption, or decryption;

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACF.1 Security attribute based access control

Hierarchical to: No other components

FDP_ACF.1.1 The TSF shall enforce the *Protected Operations Access Controls* to objects based on *security attributes TCPA_AUTH_DATA_USAGE, TCPA_KEY_FLAGS and TCPA_KEY_USAGE*.

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) *Key and data access is defined as "owner" access or "world" based on the value of TCPA_AUTH_DATA_USAGE*

b) *Cryptographic operations for each key are limited based on the specification of the TCPA_KEY_USAGE value.*

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [assignment: *rules, based on security attributes, that explicitly authorise access of subjects to objects*].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on: [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*].

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

Application Note: *The assignments in FDP_ACF.1.3 and FDP_ACF.1.4 shall be defined by the ST author.*

FDP_ETC.2 Export of user data with security attributes

Hierarchical to: No other components

FDP_ETC.2.1 The TSF shall enforce the *Protected Operations Access Controls* when exporting user data, controlled under the SFP, outside of the TSC.

FDP_ETC.2.2 The TSF shall export the user data with the user data's associated security attributes.

FDP_ETC.2.3 The TSF shall ensure that the security attributes, when exported outside the TSC, are unambiguously associated with the exported user data.

FDP_ETC.2.4 The TSF shall enforce the following rules when user data is exported from the TSC: *A key may be encrypted for migration only if the migratable flag is set in TCPA_KEY_FLAGS,* [assignment: *additional exportation control rules*].

Application note: *Security attributes are encrypted in a blob prior to export. As part of the blob that has been encrypted, the security attributes are unambiguously associated with the data.*

Dependencies: FDP_ACC.1 Subset access control

FDP_ITC.2 Import of user data with security attributes

Hierarchical to: No other components

FDP_ITC.2.1 The TSF shall enforce the *Protected Operations Access Controls* when importing user data, controlled under the SFP, from outside of the TSC.

FDP_ITC.2.2 The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3 The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4 The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: [assignment: *additional importation control rule*s].

Application note: *Security attributes are imported with data as part of the encrypted blob. As part of the blob that has been encrypted, the security attributes are unambiguously associated with the data.*

Application Note: *The assignment in FDP_ITC.2.5 shall be defined by the ST author.*

Dependencies: FDP_ACC.1 Subset access control, FTP_TRP.1 Trusted path, FPT_TDC.1 Inter-TSF basic TSF data consistency

FDP_RIP.2 Full residual information protection

Hierarchical to: FDP_RIP.1

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

Dependencies: None.

5.2.4 Class FIA – Identification and Authentication

Application Note: The TPM identification and authentication capability is used to authenticate an entity owner and to authorize use of an entity. The basic premise is to prove knowledge of a shared secret. This shared secret is the identification and authentication data. Note that the TCPA Main Specification document refers to the identification and authentication process and this data as *authorization*.

FIA_ATD.1 User attribute definition

Hierarchical to: No other components

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: *authentication data, role.*

Dependencies: None

FIA_UAU.1 Timing of authentication

Hierarchical to: No other components

FIA_UAU.1.1 The TSF shall allow *access to data and keys where entity owner has given the "world" access based on the value of TCPA_AUTH_DATA_USAGE; access to the following commands: TPM_PcrRead, TPM_DirRead, and TPM_EvictKey* on behalf of the user to be performed before the user is authenticated.

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

Dependencies: FIA_UID.1 Timing of identification

FIA_UAU.4 Single-use authentication mechanisms

Hierarchical to: No other components

FIA_UAU.4.1 The TSF shall prevent reuse of authentication data related to *the use of the "Object-Independent Authorization Protocol" (OI-AP) and the "Object-Specific Authorization Protocol" (OS-AP) protocols.*

Dependencies: None.

FIA_UAU.6 Re authenticating

Hierarchical to: No other components

FIA_UAU.6.1 The TSF shall re-authenticate the user under the conditions: *for every command that requires user authentication*.

Dependencies: None.

FIA_UID.1 Timing of identification

Hierarchical to: No other components

FIA_UID.1.1 The TSF shall allow *access to data and keys where entity owner has given the "world" access based on the value of TCPA_AUTH_DATA_USAGE; access to the following commands: TPM_PcrRead, TPM_DirRead, and TPM_EvictKey* on behalf of the user to be performed before the user is identified.

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

Dependencies: None.

5.2.5 Class FMT – Security Management

FMT_MOF.1 Management of security functions behavior

Hierarchical to: No other components

FMT_MOF.1.1 The TSF shall restrict the ability to *disable or enable* the functions [assignment: *list of functions*] to *the TPM owner*.

Dependencies: FMT_SMR.1 Security roles

Application Note: *The assignment shall be defined by the ST author.*

FMT_MSA.1 Management of security attributes

Hierarchical to: No other components

FMT_MSA.1.1 The TSF shall enforce the *Protected Operations Access Controls* to restrict the ability to *modify* the security attributes *associated with a particular entity, including TCPA_KEY_USAGE, TCPA_AUTH_DATA_USAGE, migratable flag, and volatility flag* to *the entity owner.*

Dependencies: FMT_SMR.1 Security roles, FDP_ACC.1 Subset access control

FMT_MSA.2 Secure security attributes

Hierarchical to: No other components

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

Dependencies: ADV_SPM.1 Informal TOE security policy model, FMT_SMR.1 Security roles, FDP_ACC.1 Subset access control, FMT_MSA.1 Management of security attributes

FMT_MSA.3 Static attribute initialisation

Hierarchical to: No other components

FMT_MSA.3.1 The TSF shall enforce the *Protected Operations Access Controls* to provide *specific* default values for security attributes that are used to enforce the SFP.

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

Application Note: *The security attribute default values are set by the manufacturer and must be specified in the ST*

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

FMT_MTD.1:1 Management of TSF data – TPM Owner

Hierarchical to: No other components

FMT_MTD.1.1;1 The TSF shall restrict the ability to *modify* the *TSF data: TPM Owner Key, Identity Keys, and SRK; Identification and Authentication data associated with those keys; Migration authorization data* to *the TPM Owner*.

FMT_MTD.1:2 Management of TSF data – Entity Owner

Hierarchical to: No other components

FMT_MTD.1.1;2 The TSF shall restrict the ability to *modify* the *TSF data: Identification and Authentication data associated with entity;* to the *entity Owner*.

FMT_MTD.1:3 Management of TSF data – Manufacturer

Hierarchical to: No other components

FMT_MTD.1.1;3 The TSF shall restrict the ability to *initialize or modify* the *TSF* data: *Endorsement Key Pair, TPMProof* to the *TPM manufacturer or designee*.

Dependencies: FMT_SMR.1 Security roles

FMT_SMR.2 Restrictions on security roles

Hierarchical to: FMT_SMR.1

FMT_SMR.2.1 The TSF shall maintain the roles: *TPM owner, owners of entities, and TPM manufacturer or designee*.

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

FMT_SMR.2.3 The TSF shall ensure that the condition: *successful presentation of correct authentication data* is satisfied.

Dependencies: FIA_UID.1 Timing of identification.

5.2.6 Class FPT – Protection of the TOE Security Functions

FPT_AMT.1 Abstract machine testing

Hierarchical to: No other components

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

Dependencies: None.

FPT_FLS.1 Failure with preservation of secure state

Hierarchical to: No other components

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur: *failure of any crypto operations including RSA encryption, RSA decryption, SHA, RNG, RSA signature generation, HMAC generation; failure of any commands or internal operations*.

Dependencies: ADV_SPM.1 Informal TOE security policy model

FPT_PHP.1 Passive detection of physical attack

Hierarchical to: No other components

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

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

Dependencies: FMT_MOF.1 Management of security functions behavior

FPT_RCV.4 Function recovery

Hierarchical to: No other components

FPT_RCV.4.1	The TSF shall ensure that *all TPM Commands* have the property that the SF either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.
Dependencies:	ADV_SPM.1 Informal TOE security policy model

FPT_RPL.1 Replay detection

Hierarchical to: No other components.

FPT_RPL.1.1	The TSF shall detect replay for the following entities: *command requests that include the nonce parameter.*
FPT_RPL.1.2	The TSF shall perform *destroy session* when replay is detected.
Dependencies:	None.

FPT_RVM.1 Non-bypassability of the TSP

Hierarchical to: No other components

FPT_RVM.1.1	The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.
Dependencies:	None.

FPT_SEP.1 TSF domain separation

Hierarchical to: No other components

FPT_SEP.1.1	The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.
FPT_SEP.1.2	The TSF shall enforce separation between the security domains of subjects in the TSC.
Dependencies:	None.

FPT_TDC.1 Inter-TSF basic TSF data consistency

Hierarchical to: No other components

FPT_TDC.1.1	The TSF shall provide the capability to consistently interpret *TPM commands and responses* when shared between the TSF and another trusted IT product.
FPT_TDC.1.2	The TSF shall use *the TCPA Main Specification* when interpreting the TSF data from another trusted IT product.
Dependencies:	None.

FPT_TST.1 TSF testing

Hierarchical to: No other components

FPT_TST.1.1	The TSF shall run a suite of self tests *during initial start-up and periodically during normal operation, at the request of the authorized user, and at the condition: prior to execution of the first call to a*

capability that uses those functions to demonstrate the correct operation of the TSF.

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

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

Application Note: *The self-test capabilities are designed to enable the creation of a TCPA platform with minimum latency due to TPM self-test. It might be possible to avoid wasting time, waiting for a TPM to do self-test, by designing a platform where TPM self-testing is done in parallel with other system functions, at a time when TPM capabilities are not required.*

Tests will include: at startup, a TPM automatically tests just those internal functions that are used by critical TPM capabilities. This permits the use of those critical TPM capabilities as soon as possible after startup. Remaining TPM capabilities use additional internal functions that must be tested before the remaining TPM capabilities can execute. A test of the additional functions can be explicitly called. Alternatively, those functions will automatically be tested prior to execution of the first call to a capability that uses those functions. At any time, other self-test commands will explicitly cause the TPM to do a full self-test.

TPM_SelfTestFull causes the TPM to do a full self-test.

TPM_CertifySelfTest causes the TPM to do a full self-test and sign the result. It enables the caller to verify that the self-test actually executed and trust the answer. It requires authorization to use a signing key inside the TPM. If the command fails for any reason, the command will not return a signature. The lack of a signature field returning to a Challenger is in itself an indication that some part of the process failed. The failure could be an attack against the signature or a failure in the TPM.

TPM_ContinueSelfTest causes the TPM to test the TPM internal functions that were not tested at startup. TPM_ContinueSelfTest is unusual, in that it returns a result code to the caller before execution of the command and does not return a result code to the caller after execution of the command. If the functions used by a capability have not been tested, TPM_ContinueSelfTest is executed automatically after that capability is called and before it is executed. It is anticipated that the caller or TPM driver software is preprogrammed with knowledge of the time that the TPM will require to complete TPM_ContinueSelfTest. It is anticipated that a call to a TPM that is executing TPM_ContinueSelfTest would result in a "busy" indication.

The tests themselves only return a TCPA_SUCCESS or TCPA_FAIL answer. TPM_GetTestResult must be used to discover why self-test failed. Upon the failure of a self-test the TPM goes into failure mode

and does not allow most other operations to continue. [These self-tests] demonstrate the correct operation of the TSF.

Dependencies: FPT_AMT.1 Abstract machine testing

5.2.7 Class FTP – Trusted Path/Channels

FTP_TRP.1 Trusted path

Hierarchical to: No other components

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

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

FTP_TRP.1.3 The TSF shall require the user of the trusted path for *initial user authentication, for all TPM commands, all user commands, and TSF responses.*

Dependencies: None

5.2.8 Strength of Function Requirement

The threat level for the TOE authentication function is assumed to be SOF-basic. The strength of cryptographic algorithms is outside the scope of the CC. Strength of function only applies to non-cryptographic, probabilistic or permutational mechanisms. The SOF requirement applies to the identification and authentication functionality within the TOE.

5.3 TOE Security Assurance Requirements

The Security Assurance Requirements for the TOE are the assurance components of Evaluation Assurance Level 3 (EAL3) augmented by ADV_SPM.1. They are all drawn from Part 3 of the Common Criteria. The assurance components are listed in Table 5.2. EAL 3 was selected because the TOE requires a moderate level of independently assured security and requires a thorough investigation of the TOE and its development without substantial re-engineering. ADV_SPM.1 was added because it is a dependency of functional security requirements FMT_MSA.2.

Table 5.2 - EAL3 Assurance Requirements, augmented

ACM_CAP.3	Authorisation controls
ACM_SCP.1	TOE CM coverage
ADO_DEL.1	Delivery procedures
ADO_IGS.1	Installation, generation, and start-up procedures
ADV_FSP.1	Informal functional specification
ADV_HLD.2	Security enforcing high-level design
ADV_RCR.1	Informal correspondence demonstration
ADV_SPM.1	Informal TOE security policy model [AUG]
AGD_ADM.1	Administrator guidance
AGD_USR.1	User guidance
ALC_DVS.1	Identification of security measures
ATE_COV.2	Analysis of coverage
ATE_DPT.1	Testing: high-level design
ATE_FUN.1	Functional testing
ATE_IND.2	Independent testing - sample
AVA_MSU.1	Examination of guidance
AVA_SOF.1	Strength of TOE security function evaluation
AVA_VLA.1	Developer vulnerability analysis

5.2.1 Class ACM: Configuration Management

ACM_CAP.3 Authorisation controls

Dependencies: ACM_SCP.1 TOE CM coverage, ALC_DVS.1 Identification of security measures

Developer action elements:

ACM_CAP.3.1D The developer shall provide a reference for the TOE.

ACM_CAP.3.2D The developer shall use a CM system.

ACM_CAP.3.3D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_CAP.3.1C The reference for the TOE shall be unique to each version of the TOE.

ACM_CAP.3.2C The TOE shall be labeled with its reference.

ACM_CAP.3.3C The CM documentation shall include a configuration list and a CM plan.

ACM_CAP.3.4C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.3.5C The CM documentation shall describe the method used to uniquely identify the configuration items.

ACM_CAP.3.6C The CM system shall uniquely identify all configuration items.

ACM_CAP.3.7C The CM plan shall describe how the CM system is used.

ACM_CAP.3.8C The evidence shall demonstrate that the CM system is operating in accordance with the CM plan.

ACM_CAP.3.9C The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.

ACM_CAP.3.10C The CM system shall provide measures such that only authorised changes are made to the configuration items.

Evaluator action elements:

ACM_CAP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_SCP.1 TOE CM Coverage

Dependencies: ACM_CAP.3 Authorisation controls

Developer action elements:

ACM_SCP.1.1D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_SCP.1.1C The CM documentation shall show that the CM system, as a minimum, tracks the following: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, and CM documentation.

ACM_SCP.1.2C The CM documentation shall describe how configuration items are tracked by the CM system.

Evaluator action elements:

ACM_SCP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.2 Class ADO: Delivery and Operation

ADO_DEL.1 Delivery Procedures

Dependencies: No dependencies.

Developer action elements:

ADO_DEL.1.1D The developer shall document procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.1.2D The developer shall use the delivery procedures.

Content and presentation of evidence elements:

ADO_DEL.1.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.

Evaluator action elements:

ADO_DEL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1 Installation, Generation, and Start-up Procedures

Dependencies: AGD_ADM.1 Administrator guidance

Developer action elements:

ADO_IGS.1.1D The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

Content and presentation of evidence elements:

ADO_IGS.1.1C The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

Evaluator action elements:

ADO_IGS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2E The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

5.2.3 Class ADV: Development

ADV_FSP.1 Informal functional specification

Dependencies: ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.1.1D The developer shall provide a functional specification.

Content and presentation of evidence elements:

ADV_FSP.1.1C The functional specification shall describe the TSF and its external interfaces using an informal style.

ADV_FSP.1.2C The functional specification shall be internally consistent.

ADV_FSP.1.3C The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate.

ADV_FSP.1.4C The functional specification shall completely represent the TSF.

Evaluator action elements:

ADV_FSP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

ADV_HLD.2 Security enforcing high-level design

Dependencies: ADV_FSP.1 Informal functional specification, ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_HLD.2.1D The developer shall provide the high-level design of the TSF.

Content and presentation of evidence elements:

ADV_HLD.2.1C The presentation of the high-level design shall be informal.

ADV_HLD.2.2C The high-level design shall be internally consistent.

ADV_HLD.2.3C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.2.4C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.2.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.2.6C The high-level design shall identify all interfaces to the subsystems of the TSF.

ADV_HLD.2.7C The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.

ADV_HLD.2.8C The high-level design shall describe the purpose and method of use of all interfaces to the subsystems of the TSF, providing details of effects, exceptions and error messages, as appropriate.

ADV_HLD.2.9C The high-level design shall describe the separation of the TOE into TSP-enforcing and other subsystems.

Evaluator action elements:

ADV_HLD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.2.2E The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

ADV_RCR.1 Informal correspondence demonstration

Dependencies: No dependencies.

Developer action elements:

ADV_RCR.1.1D The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

Content and presentation of evidence elements:

ADV_RCR.1.1C For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

Evaluator action elements:

ADV_RCR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_SPM.1 Informal TOE security policy model

Dependencies: ADV_FSP.1 Informal functional specification

Developer action elements:

ADV_SPM.1.1D The developer shall provide a TSP model.

ADV_SPM.1.2D The developer shall demonstrate correspondence between the functional specification and the TSP model.

Content and presentation of evidence elements:

ADV_SPM.1.1C The TSP model shall be informal.

ADV_SPM.1.2C The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_SPM.1.3C The TSP model shall include a rationale that demonstrates that it is consistent and complete with respect to all policies of the TSP that can be modeled.

ADV_SPM.1.4C The demonstration of correspondence between the TSP model and the functional specification shall show that all of the security functions in the functional specification are consistent and complete with respect to the TSP model.

Evaluator action elements:

ADV_SPM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.4 Class AGD: Guidance Documents

AGD_ADM.1Administrator guidance

Dependencies: ADV_FSP.1 Informal functional specification

Developer action elements:

AGD_ADM.1.1D The developer shall provide administrator guidance addressed to system administrative personnel.

Content and presentation of evidence elements:

AGD_ADM.1.1C The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.

AGD_ADM.1.2C The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3C The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4C The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE.

AGD_ADM.1.5C The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6C The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7C The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8C The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

Evaluator action elements:

AGD_ADM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_USR.1 User guidance

Dependencies: ADV_FSP.1 Informal functional specification

Developer action elements:

AGD_USR.1.1D The developer shall provide user guidance.

Content and presentation of evidence elements:

AGD_USR.1.1C The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2C The user guidance shall describe the use of user-accessible security functions provided by the TOE.

AGD_USR.1.3C The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

AGD_USR.1.4C The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

AGD_USR.1.5C The user guidance shall be consistent with all other documentation supplied for evaluation.

AGD_USR.1.6C The user guidance shall describe all security requirements for the IT environment that are relevant to the user.

Evaluator action elements:

AGD_USR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.5 Class ALC: Life Cycle Support

ALC_DVS.1 Identification of security measures

Dependencies: No dependencies.

Developer action elements:

ALC_DVS.1.1D The developer shall produce development security documentation.

Content and presentation of evidence elements:

ALC_DVS.1.1C The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.

ALC_DVS.1.2C The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.

Evaluator action elements:

ALC_DVS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_DVS.1.2E The evaluator shall confirm that the security measures are being applied.

5.2.6 Class ATE: Tests

ATE_COV.2 Analysis of coverage

Dependencies: ADV_FSP.1 Informal functional specification, ATE_FUN.1 Functional testing

Developer action elements:

ATE_COV.2.1D The developer shall provide an analysis of the test coverage.

Content and presentation of evidence elements:

ATE_COV.2.1C The analysis of the test coverage shall demonstrate the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.

ATE_COV.2.2C The analysis of the test coverage shall demonstrate that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

Evaluator action elements:

ATE_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_DPT.1 Testing: high-level design

Dependencies: ADV_HLD.1 Descriptive high-level design, ATE_FUN.1 Functional testing

Developer action elements:

ATE_DPT.1.1D The developer shall provide the analysis of the depth of testing.

Content and presentation of evidence elements:

ATE_DPT.1.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design.

Evaluator action elements:

ATE_DPT.1.2E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_FUN.1 Functional testing

Dependencies: No dependencies.

Developer action elements:

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation.

Content and presentation of evidence elements:

ATE_FUN.1.1C	The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.
ATE_FUN.1.2C	The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.
ATE_FUN.1.3C	The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.
ATE_FUN.1.4C	The expected test results shall show the anticipated outputs from a successful execution of the tests.
ATE_FUN.1.5C	The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.

Evaluator action elements:

ATE_FUN.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2 Independent Testing - Sample

Dependencies:	ADV_FSP.1 Informal functional specification, AGD_ADM.1 Administrator guidance, AGD_USR.1 User guidance, ATE_FUN.1 Functional testing

Developer action elements:

ATE_IND.2.1D	The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

ATE_IND.2.1C	The TOE shall be suitable for testing.
ATE_IND.2.2C	The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

Evaluator action elements:

ATE_IND.2.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ATE_IND.2.2E	The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.
ATE_IND.2.3E	The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

5.2.7 Class AVA: Vulnerability Assessment

AVA_MSU.1 Examination of guidance

Dependencies:	ADO_IGS.1 Installation, generation, and start-up procedures, ADV_FSP.1 Informal functional specification, AGD_ADM.1 Administrator guidance, AGD_USR.1 User guidance

Developer action elements:

AVA_MSU.1.1D	The developer shall provide guidance documentation.

Content and presentation of evidence elements:

AVA_MSU.1.1C	The guidance documentation shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.
AVA_MSU.1.2C	The guidance documentation shall be complete, clear, consistent and reasonable.
AVA_MSU.1.3C	The guidance documentation shall list all assumptions about the intended environment.
AVA_MSU.1.4C	The guidance documentation shall list all requirements for external security measures (including external procedural, physical and personnel controls).

Evaluator action elements:

AVA_MSU.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_MSU.1.2E	The evaluator shall repeat all configuration and installation procedures to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.
AVA_MSU.1.3E	The evaluator shall determine that the use of the guidance documentation allows all insecure states to be detected.

AVA_SOF.1 Strength of TOE security function evaluation

Dependencies: ADV_FSP.1 Informal functional specification, ADV_HLD.1 Descriptive high-level design

Developer action elements:

AVA_SOF.1.1D	The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.

Content and presentation of evidence elements:

AVA_SOF.1.1C	For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST.
AVA_SOF.1.2C	For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST.

Evaluator action elements:

AVA_SOF.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_SOF.1.2E	The evaluator shall confirm that the strength claims are correct.

AVA_VLA.1 Developer vulnerability analysis

Dependencies:	ADV_FSP.1 Informal functional specification, ADV_HLD.1 Descriptive high-level design, AGD_ADM.1 Administrator guidance, AGD_USR.1 User guidance

Developer action elements:

AVA_VLA.1.1D	The developer shall perform and document an analysis of the TOE deliverables searching for ways in which a user can violate the TSP.
AVA_VLA.1.2D	The developer shall document the disposition of obvious vulnerabilities.

Content and presentation of evidence elements:

AVA_VLA.1.1C	The documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE.

Evaluator action elements:

AVA_VLA.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_VLA.1.2E	The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

6 Rationale

This section provides further evidence and explanation to support the certification of this PP.

6.1 Security Objectives Rationale

Table 6.1 maps assumptions and threats to objectives, demonstrating that all assumptions and threats are mapped to at least one objective. Table 6.2 maps objectives to threats and assumptions, demonstrating that all objectives are mapped to at least on threat or assumption. A discussion of the rationale for threat mappings is provided below.

Table 6.1 – Mapping the TOE Security Environment to Objectives

	Assumption/Threat	**Objectives**
1E	A.Configuration	OE.Configuration
2E	AE.Physical_Protection	OE.PhysSecurity
1	T.Attack	O.DAC, O.I&A, O.Security_Roles, O.Self_Protect
2	T.Bypass	O.HMAC, O.Security_Attr_Mgt, O.Invoke
3	T.Export	O.Export
4	T.Hack_Crypto	O.Crypto_Op
5	T.Hack_Physical	O.Tamper_ID
6	T.Imperson	O.I&A, O.Security_Roles, O.Import
7	T.Import	O.Import
8	T.Key_Gen_Destroy	O.Crypto_Key_Man
9	T.Malfunction	O.Fail_Secure
10	T.Modify	O.Limit_Actions_Auth, O.Security_Attr_Mgt, O.Security_Roles, O.Crypto_Key_Man
11	T.Object_Attr_Default	O.Object_Attr_Default
12	T.Object_Attr_Change	O.Object_Attr_DefaultOver
13	T.Object_SecureValues	O.Obj_Attr_SecureValues
14	T.Residual_Info	O.No_Residual_Info, O.Cryto_Key_Man
15	T.Replay	O.Single_Auth
16	T.Repudiate_Transact	O.MessageNR
17	T.Test	O.Crypto_Self_Test, O.General_Integ_Checks

Table 6.2 – Tracing of Security Objectives to Threats and Assumptions

	Objectives	**Threat/Assumptions**
1E	OE.Configuration	A.Configuration
2E	OE.PhysSecurity	AE.Physical_Protection
1	O.Crypto_Key_Man	T.Residual_Info, T.Key_Gen_Destroy
2	O.Crypto_Op	T.Hack_Crypto
3	O.Crypto_Self_Test	T.Test
4	O.DAC	T.Attack
5	O.Export	T.Export
6	O.Fail_Secure	T.Malfunction
7	O.General_Integ_Checks	T.Test
8	O.HMAC	T.Bypass
9	O.I&A	T.Attack, T.Imperson
10	O.Import	T.Import, T.Imperson
11	O.Invoke	T.Bypass
12	O.Limit_Actions_Auth	T.Modify
13	O.MessageNR	T.Repudiate_Transact
14	O.No_Residual_Info	T.Residual_Info
15	O.Object_Attr_Default	T.Object_Attr_Default
16	O.Object_Attr_DefaultOver	T.Object_Attr_Change
17	O.Obj_Attr_SecureValues	T.Object_SecureValues
18	O.Security_Attr_Mgt	T.Modify, T.Bypass
19	O.Security_Roles	T.Attack, T.Modify, T.Imperson
20	O.Self_Protect	T.Attack
21	O.Single_Auth	T.Replay
22	O.Tamper_ID	T.Hack_Physical

6.1.1 Threats

This section describes each threat and enumerates and discusses the security objectives that counter the threat.

T.Attack: An undetected compromise of the cryptography-related IT assets may occur as a result of an attacker (whether an insider or outsider) attempting to perform actions that the individual is not authorised to perform.

T.Attack is countered by O.DAC, O.I&A, O.Security_Roles, and O.Self_Protect. These objectives limit the ability of a user to the performance of only those actions that the user is authorized to perform:

- O.DAC: The TOE shall provide its users with the means of controlling and limiting access to the TOE assets in accordance with a specified access control policy. This objective limits an attacker from performing unauthorized actions through a defined access control policy.
- O.I&A: The TOE shall uniquely identify all users, and shall authenticate the claimed identify before granting a user access to the TOE facilities. This objective supports the access control policy by uniquely identifying users (key pairs within the TOE) so that specific access control rules can be applied for each user role.
- O.Security_Roles: The TOE shall maintain security-relevant roles and association of users with those roles. This objective further supports the access control policy by associating each user with a role, which then can be assigned a specific access control policy.
- O.Self_Protect: The TSF will maintain a domain for its own execution that protects it and its resources from external interference, tampering, or unauthorized disclosure.

T.Bypass: An unauthorized individual or user may tamper with security attributes or other data in order to bypass TOE security functions and gain unauthorized access to TOE assets.

T.Bypass is countered by O.HMAC, O.Security_Attr_Mgt, and O.Invoke. These three objectives allow the TOE to detect tampering with data and to counter the ability of unauthorized users from tampering with security attributes or other data:

- O.HMAC: The TOE shall provide the ability to detect the modification of security attributes and other data. This objective provides the capability for the system to detect tampering with data.
- O.Security_Attr_Mgt: The TOE shall allow only authorised users to initialise and change object security attributes. This objective requires that only authorized users be allowed to initialise and change security attributes, which counters the threat of an unauthorized user making such changes.
- O.Invoke: The TSF shall be invoked for all actions. This objective assists in the protection of the system from tampering by unauthorised users, since it requires the TSF to be invoked for all actions and does not allow it to be bypassed by any user.

T.Export: A user or an attacker may export data without security attributes or with unsecure security attributes, causing the data exported to be erroneous and unusable, to allow erroneous data to be added or substituted for the original data, and/or to reveal secrets.

T.Export I countered by O.Export. O.Export states: When data are exported outside the TPM, the TOE shall ensure that the data security attributes being exported are unambiguously associated with the data.

T.Hack_Crypto: Cryptographic algorithms may by incorrectly implemented, allowing an unauthorized individual or user to decipher keys generated within the TPM and thereby gain unauthorised access to encrypted data.

T.Hack_Crypto is countered by O.Crypto_Op, which states: The TOE shall perform cryptographic operations, including secure hash, HMAC, RSA digital signature and signature verification, RSA encryption and decryption, and RSA key generation in accordance with specified algorithms and key size; key size must be sufficient size to protect private/public key pairs from deciphering.

T.Hack_Physical: An unauthorised individual or user of the TOE may cause unauthorised disclosure or modification of TOE assets by physically interacting with the TOE to exploit vulnerabilities in the physical environment.

T.Hack_Physical is countered by O.Tamper_ID, which states: The TOE shall provide features that permit a human to detect physical tampering of a system component. Although this objective does not prevent physical tampering, it allows physical tampering to be detected if the TOE is physically examined.

T.Imperson: An unauthorized individual may impersonate an authorised user of the TOE and thereby gain access to TOE data, keys, and operations.

T.Imperson is countered by O.I&A, O.Security_Roles, and O.Import. These objectives require a user to be identified and authenticated and to function under a predefined role with specified access control policy:

- O.I&A: The TOE shall uniquely identify all users, and shall authenticate the claimed identify before granting a user access to the TOE facilities. This objective requires identification and authentication of users (key pairs within the TOE) so that specific access control rules can be applied for each user role.
- O.Security_Roles: The TOE shall maintain security-relevant roles and association of users with those roles. This objective further requires the association of each user with a role, which then can be assigned a specific access control policy.
- O.Import: When data are being imported into the TOE, the TOE shall ensure that the data security attributes are being imported with the data and the data is from authorized source. In addition, the TOE shall verify those security attributes according to the TSF access control rules.

T.Import: A user or attacker may import data or keys without security attributes or with erroneous security attributes, causing key ownership and authorization to be uncertain or erroneous and the system to malfunction or operate in an unsecure manner.

T.Import is countered by O.Import, which states: When data are being imported into the TOE, the TOE shall ensure that the data security attributes are being imported with the data and the data is from authorized source. In addition, the TOE shall verify those security attributes according to the TSF access control rules.

T.Key_Gen_Destroy: Cryptographic keys may be generated or destroyed in an unsecure manner, causing key compromise.

T.Key_Gen_Destroy is countered by O.Crypto_Key_Man, which states: The TOE shall generate and destroy cryptographic keys in a secure manner.

T.Malfunction: TOE assets may be modified or disclosed to an unauthorised individual or user of the TOE, through malfunction of the TOE.

T.Malfunction is countered by O.Fail_Secure, which states: The TOE shall preserve the secure state of the system in the event of a cryptographic or other failure.

T.Modify: An attacker may modify data, e.g., stored security attributes or keys, in order to impersonate an authorised user or to gain access to the TOE and its assets. The integrity of the information may be compromised due to the unauthorised modification or destruction of the information by an attacker.

T.Modify is countered by O.Limit_Actions_Auth, O.Security_Attr_Mgt, O.Security_Roles, and O.Crypto_Key_Man. These objectives support the ability of the TOE to limit unauthorized user access and to maintain data and system integrity through appropriate management of cryptographic data in particular:

- O.Limit_Actions_Auth: The TOE shall restrict the actions a user may perform before the TOE verifies the identity of the user.
- O.Security_Attr_Mgt: The TOE shall allow only authorised users to initialise and change object security attributes.
- O.Security_Roles: The TOE shall maintain security-relevant roles and association of users with those roles.
- O.DAC: The TOE shall control and restrict user access to the TOE assets in accordance with a specified access control policy.

T. Object_Attr_Default: An attacker may create an object with no security attribute values.

T.Object_Attr_Default is countered by O.Object_Attr_Default, which states: The TOE shall require default security attributes for the object when the object is created.

T.Object_Attr_Change: A user or attacker may make unauthorized changes to security attribute values for an object.

T.Object_Attr_Change is countered by O.Object_Attr_DefaultOver, which states: The TOE shall permit authorised users to override defaulted values for security attributes for an object.

T.Object_SecureValues: An attacker or user may set unsecure values for object security attributes.

T.Object_SecureValues is countered by O.Obj_Attr_SecureValues, which states: The TOE shall maintain object security attributes by permitting only secure values.

T.Residual_Info: A user may obtain information that the user is not authorized to have when the data is no longer actively managed by the TOE (“data scavenging”).

T.Residual_Info is countered by O.No_Residual_Info and O.Crypto_Key_Man. O.No_Residual_Info ensure that no residual data is left in buffers or system locations. O.Crypto_Key_Man specifies that cryptographic key destruction must be performed:

- O.No_Residual_Info: The TOE shall ensure there is no “object reuse,” i.e., ensure that there is no residual information in information containers or system resources upon their reallocation to different users.
- O.Crypto_Key_Man: The TOE shall generate and destroy cryptographic keys in a secure manner.

T.Replay: An unauthorized individual may gain access to the system and sensitive data through a “replay” or “man-in-the-middle” attack that allows the individual to capture identification and authentication data.

T.Replay is countered by O.Single_Auth, which states: The TOE shall provide a single use authentication mechanism and require re-authentication to prevent “replay” and “man-in-the-middle” attacks.

T.Repudiate_Transact: An originator of data may deny originating the data to avoid accountability.

T.Repudiate_Transact is countered by O.MessageNR, which states: The TOE shall provide user data integrity, source authentication, and the basis for source non-repudiation when exchanging data with a remote system.

T.Test: The TOE may start-up in an unsecure state or enter an unsecure state, allowing an attacker to obtain sensitive data or compromise the system.

T.Test is countered by O.Crypto_Self_Test and O.General_Integ_Checks. These objectives require the TOE to provide self test and integrity checking functionality in order to detect unsecure states either at startup or during normal operation:

- O.Crypto_Self_Test: The TOE shall provide the ability to verify that the cryptographic functions operate as designed.
- O.General_Integ_Checks: The TOE shall provide periodic integrity checks on both system and user data.

6.2 Security Requirements Rationale

In this section, the objectives are mapped to the functional requirements and rationale is provided for the selected EAL and its components and augmentation.

6.2.1 Functional Security Requirements Rationale

The mapping of security objectives to functional requirements (components) is provided in Table 6.3.

Table 6.3 – Functional Component to Security Objective Mapping

	Objectives	Functional Component
1	O.Crypto_Key_Man	FCS_CKM.1, FCS_CKM.4
2	O.Crypto_Op	FCS_COP.1, all iterations
3	O.Crypto_Self_Test	FPT_AMT.1, FPT_TST.1
4	O.DAC	FDP_ACC.1, FDP_ACF.1, FMT_MOF.1, FMT.MTD.1 (all iterations)
5	O.Export	FDP_ETC.2
6	O.Fail_Secure	FPT_FLS.1, FPT_RCV.4
7	O.General_Integ_Checks	FPT_AMT.1, FPT_TST.1
8	O.HMAC	FCS_COP.1:4
9	O.I&A	FIA_UAU.1, FIA_UID.1, FIA_ATD.1
10	O.Import	FDP_ITC.2, FPT_TDC.1, FTP_TRP.1
11	O.Invoke	FPT_RVM.1
12	O.Limit_Actions_Auth	FIA_UAU.1, FIA_UID.1
13	O.MessageNR	FCO_NRO.2, FDP_ETC.2
14	O.No_Residual_Info	FDP_RIP.2
15	O.Object_Attr_Default	FMT_MSA.3
16	O.Object_Attr_DefaultOver	FMT_MSA.3
17	O.Obj_Attr_SecureValues	FMT_MSA.2, FPT_TDC.1
18	O.Security_Attr_Mgt	FMT_MSA.3, FMT_MSA.1
19	O.Security_Roles	FMT_SMR.2, FIA_ATD.1
20	O.Self_Protect	FPT_SEP.1
21	O.Single_Auth	FIA_UAU.4, FIA_UAU.6, FPT_RPL.1
22	O.Tamper_ID	FPT_PHP.1

A discussion of the rationale for the mapping is provided for each objective below.

O.Crypto_Key_Man: The TOE shall generate and destroy cryptographic keys in a secure manner.

O.Crypto_Key_Man is mapped to:

- FCS_CKM.1, Cryptographic key generation, which requires that cryptographic keys be generated in accordance with the RSA algorithm with specified cryptographic sizes that meet PKCS #1 V.2 standard.
- FCS_CKM.4, Cryptographic key destruction, which requires that cryptographic keys be destroyed in accordance with a specified secure key destruction method.

O.Crypto_Op: The TOE shall perform cryptographic operations, including secure hash, HMAC, RSA digital signature and signature verification, RSA encryption and decryption, and RSA key generation in accordance with specified algorithms and key size; key size must be sufficient size to protect private/public key pairs from deciphering.

O.Crypto_Op is mapped to:

- FCS_COP.1, Cryptographic operations. There are four iterations of this component, including RSA encrypt and decrypt, RSA signature and signature verification, SHA, and Keyed-Hashing for Message Authentication. The iterations cover all cryptographic operations and specify key sizes and standards that must be met.

O.Crypto_Self_Test: The TOE shall provide the ability to verify that the cryptographic functions operate as designed.

O.Crypto_Self_Test is mapped to:

- FPT_AMT.1: Abstract machine testing. This component tests the cryptographic portion of the underlying abstract state machine.
- FPT_TST.1: TSF testing. This component defines self-tests to ensure that the cryptographic functions are operating correctly. Tests conducted during start-up and/or periodically may include known-answer tests of cryptographic operations, as well as statistical tests on random number generators. Additional tests may involve generation of private / public key pairs, pair-wise consistency tests of encryption and decryption, key-entry tests, and key integrity tests.

O.DAC: The TOE shall provide its users with the means of controlling and limiting access to the TOE assets in accordance with a specified access control policy.

O.DAC is mapped to:

- FDP_ACC.1, Subset access control, which requires that Protected Operations Access Controls by enforced on subjects, objects and operations.
- FDP_ACF.1, Security attribute based access control, which defines access controls based on TCPA_AUTH_DATA_USAGE and TCPA_KEY_USAGE values.
- FMT_MOF.1, Management of security functions behaviour, allows the ST author to specify the list of functions that are restricted to the TPM owner.
- FMT_MTD.1, Management of TSF Data, ensures that the TSF data is accessible to authorized users.

O.Export: When data are exported outside the TPM, the TOE shall ensure that the data security attributes being exported are unambiguously associated with the data.

O.Export is mapped to:

- FDP_ETC.2, Export of user data with security attributes, which requires that data exported outside the TSF have security attributes that are unambiguously associated with the data exported.

O.Fail_Secure: The TOE shall preserve the secure state of the system in the event of a cryptographic or other failure.

O.Fail_Secure is mapped to:

- FPT_FLS.1, Failure with preservation of secure state, which requires that the TSF preserve a secure state in the event of a failure.
- FPT_RCV.4, Function recovery, which requires that all TPM Commands either complete successfully or fail and recover to a secure state.

O.General_Integ_Checks: The TOE shall provide periodic integrity checks on both system and user data.

O.General_Integ_Checks is mapped to:

- FPT_AMT.1: Abstract machine testing. This component tests the cryptographic portion of the underlying abstract state machine.
- FPT_TST.1: TSF testing. This component defines self-tests to ensure that the cryptographic functions are operating correctly. Tests conducted during start-up and/or periodically may include known-answer tests of cryptographic operations, as well as statistical tests on random number generators. Additional tests may involve generation of private / public key pairs, pair-wise consistency tests of encryption and decryption, key-entry tests, and key integrity tests.

O.HMAC: The TOE shall provide the ability to detect the modification of security attributes and other data.

O.HMAC is mapped to:

- FCS_COP.1.1;4, which requires that the TOE provide HMAC capability in conformance with the referenced standard to provide the ability to detect the modification of security attributes and other data.

O.I&A: The TOE shall uniquely identify all users, and shall authenticate the claimed identify before granting a user access to the TOE facilities. The TPM identification and authentication capability is used to authenticate an entity owner and to authorize use of an entity. The basic premise is to prove knowledge of a shared secret. This shared secret is the identification and authentication data. Note that the TCPA Main Specification document refers to the identification and authentication process and this data as ***authorization***.

O.I&A is mapped to:

- FIA_UAU.1, Timing of authentication, which states that a user shall be successfully authenticated before performing all actions except those explicitly defined.
- FIA_UID.1, Timing of identification, which states that a user shall be successfully identified before performing all actions except those explicitly defined.
- FIA_ATD.1, User attribute definition, which supports FIA_UAU.1 and FIA_UID.1 by providing a requirement for user attributes. Authentication data is defined as

a user attribute. Authentication data in this case is associated with a specific key, which is analogous to a user.

O.Import: When data are being imported into the TOE, the TOE shall ensure that the data security attributes are being imported with the data and the data is from authorized source. In addition, the TOE shall verify those security attributes according to the TSF access control rules.

O.Import is mapped to:

- FDP_ITC.2, Import of user data with security attributes, which states that data imported into the TOE must have security attributes. These include authentication data on user keys.
- FPT_TDC.1, Inter-TSF basic TSF data consistency, defines security attributes and requires that they be consistently interpreted when importing data.
- FTP_TRP.1, Trusted path ensures that the data is being received from an authorized source. Trusted path is also a dependency of FDP_ITC.2, requiring a trusted path for data import.

O.Invoke: The TSF shall be invoked for all actions.

O.Invoke is mapped to:

- FPT_RVM.1, Non-bypassability of the TSP, which ensures that TSP functions are invoked and succeed before each function within the TSC is allowed to proceed.

O.Limit_Actions_Auth: The TOE shall restrict the actions a user may perform before the TOE verifies the identity of the user.

O.Limit_Actions_Auth is mapped to:

- FIA_UAU.1, Timing of authentication, which states that a user shall be successfully authenticated before performing all actions except those explicitly defined.
- FIA_UID.1, Timing of identification, which states that a user shall be successfully identified before performing all actions except those explicitly defined.

O.MessageNR: The TOE shall provide user data integrity, source authentication, and the basis for source non-repudiation when exchanging data with a remote system.

O.MessageNR is mapped to:

- FCP_NRO.2, Enforced proof of origin, which requires that the TSF enforce generation of data that provides evidence of origin for data transmitted.
- FDP_ETC.2, Export of user data with security attributes, ensures that access control SFPs and security attributes are associated with exported data, thereby providing user data integrity.

O.No_Residual_Info: The TOE shall ensure there is no “object reuse,” i.e., ensure that there is no residual information in information containers or system resources upon their reallocation to different users.

O.No_Residual_Info is mapped to:

- FDP_RIP.2, Full residual information protection, which requires that any previous information content of a resource be made unavailable.

O.Object_Attr_Default: The TOE shall require default security attributes for the object when the object is created.

O.Object_Attr_Default is mapped to:

- FMT_MSA.3, Static attribute initialisation, which requires that security attributes be specified and that certain defaults be in place.

O.Object_Attr_DefaultOver: The TOE shall permit authorised users to override defaulted values for security attributes for an object.

O.Object_Attr_DefaultOver is mapped to:

- FMT_MSA.3, Static attribute initialisation, which requires that security attributes be specified, that certain defaults be defined, and that authorised users have the capability to override the defaults.

O.Obj_Attr_SecureValues: The TOE shall maintain object security attributes by permitting only secure values.

O.Obj_Attr_SecureValues is mapped to:

- FMT_MSA.2, Secure security attributes, which requires that only secure values be accepted for security attributes.
- FPT_TDC.1, Inter-TSF basic TSF data consistency, defines security attributes.

O.Security_Attr_Mgt: The TOE shall allow only authorised users to initialise and change object security attributes.

O.Security_Attr_Mgt is mapped to:

- FMT_MSA.3, Static attribute initialisation, which requires that security attributes be specified.
- FMT_MSA.1, Management of security attributes, which specifies that access controls, requiring security attributes for objects be enforced.

O.Security_Roles: The TOE shall maintain security-relevant roles and association of users with those roles.

O.Security_Roles is mapped to:

- FMT_SMR.2, Restrictions on security roles, which requires that the TSF maintain roles and that the roles be associated with users.
- FIA_ATD.1, User attribute definition, which provides a requirement for user attributes. Authentication data is defined as a user attribute. Authentication data in this case is associated with a specific key, which is analogous to a user. Note that the TPM identification and authentication capability is used to authenticate an entity owner and to authorize use of an entity. The basic premise is to prove knowledge of a shared secret. This shared secret is the identification and authentication data. Note that the TCPA Main Specification document refers to the identification and authentication process and this data as ***authorization***.

O.Self_Protect: The TSF will maintain a domain for its own execution that protects it and its resources from external interference, tampering, or unauthorized disclosure.

O.Self_Protect is mapped to FPT_SEP.1, TSP domain separation which requires the TSF to protect itself.

O.Single_Auth: The TOE shall provide a single use authentication mechanism and require re-authentication to prevent “replay” and “man-in-the-middle” attacks.

O.Single_Auth is mapped to:

- FIA_UAU.4, Single-use authentication mechanisms, which prevents the reuse of authentication data.
- FIA_UAU.6, Re authenticating, which requires that a user be re authenticated for every command that requires user authentication.
- FPT_RPL.1, Replay detection, prevents replay attacks.

O.Tamper_ID: The TOE shall provide features that permit a human to detect physical tampering of a system component.

O.Tamper_ID is mapped to:

- FPT_PHP.1, Passive detection of physical attack, which requires that physical tampering with the TOE be detectable.

6.2.2 Assurance Requirement Rationale

EAL 3 was selected because the TOE requires a moderate level of independently assured security and requires a thorough investigation of the TOE and its development without substantial re-engineering. EAL 3 provides assurance by an analysis of the security functions, using a functional and interface specification, guidance documentation, and the high-level design of the TOE to understand the security behaviour. The analysis is supported by independent testing of the TOE security functions, evidence of developer testing based on the functional specification and high-level design, selective independent confirmation of the developer test results, strength of function analysis, and evidence of a developer search for obvious vulnerabilities.

EAL 3 is augmented with ADV_SPM.1 because ADV_SPM.1 is a dependency of functional security requirements FMT_MSA.2.

6.2.3 Strength of Function Rationale

The threat level for the TOE authentication function is SOF-basic. The strength of cryptographic algorithms is outside the scope of the CC. Strength of function only applies to non-cryptographic, probabilistic or permutational mechanisms. The SOF requirement applies to the identification and authentication functionality within the TOE. Attackers are expected to have moderate skills and limited time, which supports a threat level of SOF-basic.

A SOF rating reflects the attacker, described in terms of attack potential, against which the probabilistic or permutational security function is designed to protect. To determine a SOF rating for the I&A functionality provided in the TPM, the developer of the ST must define the attack potential. Analysis would include comparison to the table below, taken from Annex B of the CEM. Note that the I&A mechanism used in the TPM is

manufacturer specific and SOF analysis must be performed as part of ST development. Note that the SOF rating is required by this PP to be Basic or higher.

Sample Calculated Attack Potential from CEM Annex B

Result	Attack Potential to effect a successful attack	Resistant to attacker with attack potential	SOF Rating	VLA
0 –1	Minimum			
2	Low	Minimum		VLA.1
3 – 8	Moderate	Minimum and Low	Basic	VLA.2
9	High	Minimum to Moderate	Medium	VLA.3
10	Beyond Practicality	Moderate to High	High	VLA.4

6.3 Dependency Rationale

Table 6.4 – Functional Requirements Dependencies

#	Requirement	Dependencies
1	FCO_NRO.2	FIA_UID.1
2	FCS_CKM.1	FCS_COP.1, FCS_CKM.4, FMT_MSA.2
3	FCS_CKM.4	FCS_CKM.1, FMT_MSA.2
4	FCS_COP.1	FCS_CKM.1, FCS_CKM.4, FMT_MSA.2
5	FDP_ACC.1	FDP_ACF.1
6	FDP_ACF.1	FDP_ACC.1, FMT_MSA.3
7	FDP_ETC.2	FDP_ACC.1
8	FDP_ITC.2	FDP_ACC.1, FTP_TRP.1, FPT_TDC.1
9	FDP_RIP.2	None
10	FIA_ATD.1	None
11	FIA_UAU.1	FIA_UID.1
12	FIA_UAU.4	None
13	FIA_UAU.6	None
14	FIA_UID.1	None
15	FMT_MOF.1	FMT_SMR.1
16	FMT_MSA.1	FDP_ACC.1, FMT_SMR.1
17	FMT_MSA.2	ADV_SPM.1, FDP_ACC.1, FMT_MSA.1, FMT_SMR.1
18	FMT_MSA.3	FMT_MSA.1, FMT_SMR.1
19	FMT_MTD.1	FMT_SMR.1
20	FMT_SMR.2	None
21	FPT_AMT.1	None
22	FPT_FLS.1	ADV_SPM.1
23	FPT_PHP.1	FMT_MOF.1

Table 6.4 – (concluded)

#	Requirement	Dependencies
24	FPT_RCV.4	ADV_SPM.1
25	FPT_RPL.1	None
26	FPT_RVM.1	None
27	FPT_SEP.1	None
28	FPT_TDC.1	None
29	FPT_TST.1	FPT_AMT.1
30	FPT_TRP.1	None

6.4 Security Functional Requirements Grounding in Objectives

Table 6.5 – Requirements to Objectives Mapping

#	Requirements	Objectives
1	FCO_NRO.2	O.MessageNR
2	FCS_CKM.1	O.Crypto_Key_Man
3	FCS_CKM.4	O.Crypto_Key_Man
4-1	FCS_COP.1; 1	O.Crypto_Op
4-2	FCS_COP.1; 2	O.Crypto_Op
4-3	FCS_COP.1; 3	O.Crypto_Op
4-4	FCS_COP.1; 4	O.Crypto_Op, O.HMAC
5	FDP_ACC.1	O.DAC
6	FDP_ACF.1	O.DAC
7	FDP_ETC.2	O.Export, O.MessageNR
8	FDP_ITC.2	O.Import
9	FDP_RIP.2	O.No_Residual_Info
10	FIA_ATD.1	O.I&A, O.Security_Roles
11	FIA_UAU.1	O.I&A, O.Limit_Actions_Auth
12	FIA_UAU.4	O.Single_Auth
13	FIA_UAU.6	O.Single_Auth
14	FIA_UID.1	O.I&A, O.Limit_Actions_Auth
15	FMT_MOF.1	O.DAC
16	FMT_MSA.1	O.Security_Attr_Mgt
17	FMT_MSA.2	O.Obj_Attr_SecureValues
18	FMT_MSA.3	O.Security_Attr_Mgt, O.Object_Attr_Default, O.Object_Attr_DefaultOver
19	FMT_MTD.1	O.DAC (all iterations of FMT_MTD.1)
20	FMT_SMR.2	O.Security_Roles

Table 6.5 - (concluded)

#	Requirements	Objectives
21	FPT_AMT.1	O.Crypto_Self_Test, O.General_Integ_Checks
22	FPT_FLS.1	O.Fail_Secure
23	FPT_PHP.1	O.Tamper_ID
24	FPT_RCV.4	O.Fail_Secure
25	FPT_RPL.1	O.Single_Auth
26	FPT_RVM.1	O.Invoke
27	FPT_SEP.1	O.Self_Protect
28	FPT_TDC.1	O.Obj_Attr_SecureValues, O.Import
29	FPT_TST.1	O.Crypto_Self_Test, O.General_Integ_Checks
30	FTP_TRP.1	O.Import

Appendix – Acronyms and Glossary

Acronyms

CC - Common Criteria
EAL - Evaluation Assurance Level
IT - Information Technology
PP - Protection Profile
SF - Security Function
SFP - Security Function Policy
SOF - Strength of Function
ST - Security Target
TOE - Target of Evaluation
TSC - TSF Scope of Control
TSF - TOE Security Functions
TSFI - TSF Interface
TSP - TOE Security Policy

Glossary

3DES:	DES using a key of a size that is 3X the size that of a DES key. See DES.
Blob:	Opaque data of fixed or variable size. The meaning and interpretation of the data is outside the scope and context of the Subsystem.
Challenger:	An entity that requests and has the ability to interpret integrity metrics from a Subsystem.
Conformance Credential:	A credential that states the conformance to the TCPA specification of: the TPM; the method of incorporation of the TPM into the platform; the RTM; and the method of incorporation of the RTM into the platform.
Denial-of-service attack:	An attack on a system (or subsystem) which has no affect on information except to prevent its use.
DES:	Symmetric key encryption using a key size of 56 bits defined by NIST as FIPS 46-3.
Endorsement Credential:	A credential containing a public key (the endorsement public key) that was generated by a genuine TPM.
Endorsement Key:	A term used ambiguously, depending on context, to mean a pair of keys, or the public key of that pair, or the private key of that pair; an asymmetric key pair generated by or inserted in a TPM that is used as proof that a TPM is a genuine TPM; the public endorsement key (PUBEK); the private endorsement key (PRIVEK).
Identity Credential:	A credential issued by a Privacy CA that provides an identity for the TPM.
Integrity metric(s):	Values that are the results of measurements on the integrity of the platform.
Man-in-the-middle attack:	An attack by an entity intercepting communications between two others without their knowledge and by intercepting that

	communication is able to obtain or modify the information between them.
Migratable:	A key which may be transported outside the specific TPM.
Nonce:	A nonce is a random value that provides protection from replay and other attacks. Many of the commands and protocols in the specification require a nonce.
Non-Migratable:	A key which cannot be transported outside a specific TPM; a key that is (statistically) unique to a particular TPM.
Owner:	The entity that owns the platform in which a TPM is installed. Since there is, by definition, a one-to-one relationship between the TPM and the platform, the Owner is also the Owner of the TPM. The Owner of the platform is not necessarily the "user" of the platform (e.g., in a corporation, the Owner of the platform might be the IT department while the user is an employee.) The Owner has administration rights over the TPM.
PKI Identity Protocol:	The protocol used to insert anonymous identities into the TPM.
Platform Credential:	A credential that states that a specific platform contains a genuine TCPA Subsystem.

Privacy CA: An entity that issues an Identity Credential for a TPM based on trust in the entities that vouch for the TPM via the Endorsement Credential, the Conformance Credential, and the Platform Credential.

Private Endorsement Key (PRIVEK): The private key of the key pair that proves that a TPM is a genuine TPM. The PRIVEK is (statistically) unique to only one TPM.

Public Endorsement Key (PUBEK): A public key that proves that a TPM is a genuine TPM. The PUBEK is (statistically) unique to only one TPM.

Random number generator (RNG): A pseudo-random number generator that must be initialised with unpredictable data and provides, "random" numbers on demand.

Root of Trust for Measurement (RTM): The point from which all trust in the measurement process is predicated.

Root of Trust for Reporting (RTR): The point from which all trust in reporting of measured information is predicated.

Root of Trust for Storing (RTS): The point from which all trust in Protected Storage is predicated.

RSA:	An (asymmetric) encryption method using two keys: a private key and a public key. Reference: http://www.rsa.com.
SHA-1:	A NIST defined hashing algorithm producing a 160-bit result from an arbitrary sized source as specified in FIPS 180-1.
Storage Root Key (SRK):	The root key of a hierarchy of keys associated with a TPM; generated within a TPM; a non-migratable key.
Subsystem:	The combination of the TSS and the TPM.
Support Services (TSS):	Services to support the TPM but which do not need the protection of the TPM. The same as Trusted Platform Support Services.
TCPA-protected capability:	A function which is protected within the TPM, and has access to TPM secrets.

TPM Identity: One of the anonymous PKI identities belonging to a TPM; a TPM may have multiple identities.

Trusted Platform Agent (TPA): Trusted Platform Agent; the component within the platform that reports integrity metrics, logs, Validation Data, etc. to a Challenger; outside the scope of this specification.

Trusted Platform Measurement Store (TPMS): Storage locations within the Subsystem, which contain unprotected logs of measurement process.

Trusted Platform Module (TPM): The set of functions and data that are common to all types of platform, which must be trustworthy if the Subsystem is to be trustworthy; a logical definition in terms of protected capabilities and shielded locations.

Trusted Platform Support Services (TSS): The set of functions and data that are common to all types of platform, which are not required to be trustworthy (and therefore do not need to be part of the TPM).

User: An entity that uses the platform in which a TPM is installed. The only rights that a User has over a TPM are the rights given to the User by the Owner. These rights are expressed in the form of authentication data, given by the Owner to the User, that permits access to entities protected by the TPM. The User of the platform is not necessarily the “owner” of the platform (e.g., in a corporation, the owner of the platform might be the IT department while the User is an employee). There can be multiple Users.

Validation Credential: A credential that states values of measurements that should be obtained when measuring a particular part of the platform when the part is functioning as expected.

Validation Data: Data inside a Validation Credential; the values that the integrity measurements should produce when the part of a platform described by the Validation Credential is working correctly.

Validation Entity: An entity that issues a Validation Certificate for a component; the manufacturer of that component; an agent of the manufacturer of that component.