Protection Profile
Encrypted Storage Device

In Cooperation between

Table of content

1 Introduction

1.1 PP reference

Table 1: PP reference	
PP Title	Protection Profile Encrypted Storage Device
PP Version	2.1
TOE	Personal Storage device for temporary storage of data
Evaluation Assurance Level	EAL2 augmented with ATE_COV.2
CC Version	CC v3.1. Release 3
PP Author	Anna-Lena Hallgren

1.2 TOE overview

This Protection Profile (PP) applies to encrypted personal storage devices used for temporary storage of data whilst the data is in transit between two trusted host computers.

All confidential information stored in persistent memory in the Storage device shall be encrypted to prevent the information from being disclosed to unauthorized parties in the event that the Storage device is lost or stolen.

The user shall be required to authenticate to the device by providing a user secret prior to being allowed to perform any other operation. The user secret shall be input directly on the device, or via an application executing on the trusted host computer. The application is considered part of the TOE if user secrets are supposed to be entered into the host computer. The user secret shall not be a biometric identifier.

All keys needed to encrypt the Storage device shall be derived from the user secret using a key derivation scheme or generated by a random number generator. The user secret shall have sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries.

In order to make it more difficult to mount an exhaustive search, the Storage device may implement a delay mechanism with respect to erroneous authentication attempts. This is a recommendation and there are not any requirements for a delay mechanism in this PP.

If an application on the trusted host computer is used to initialize or authenticate to the Storage device, the authenticity of the application shall be verified prior to it being installed or upgraded. A procedure whereby the user can verify the authenticity of the application shall be established.

All cryptographic algorithms and schemes used in the Storage device shall be approved for the intended usage by the relevant national authorities in the countries where the device is to be used. In absence of national regulations HKV 12 830:51795 "Swedish national list of approved cryptographic primitives for use with the PP for Encrypted Storage Devices" shall be used. All security functions, including cryptographic algorithms and schemes, shall be implemented in the Storage device with the possible exception of the KDF, which may be implemented in the Application for authentication executing on the host computer. The Storage device should implement self-tests of random number generation and cryptographic mechanisms. Self-tests are a recommendation and there are not any requirements for self-tests in this PP.

1.2.1 TOE architecture

There are two architectures for the TOE, type A and type B.

In architecture type A, the Application for initialization and the Application for authentication are stored on and executed by the Storage device. The user secret is input directly into the Storage device.

In architecture type B, the Application for initialization and the Application for authentication are executed by the host. The user secret is input to the Storage device via the host. The Application for initialization and the Application for authentication may either be installed on the host or be stored on the Storage device.

In both types, the Application for initialization and the Application for authentication are included in the TOE.

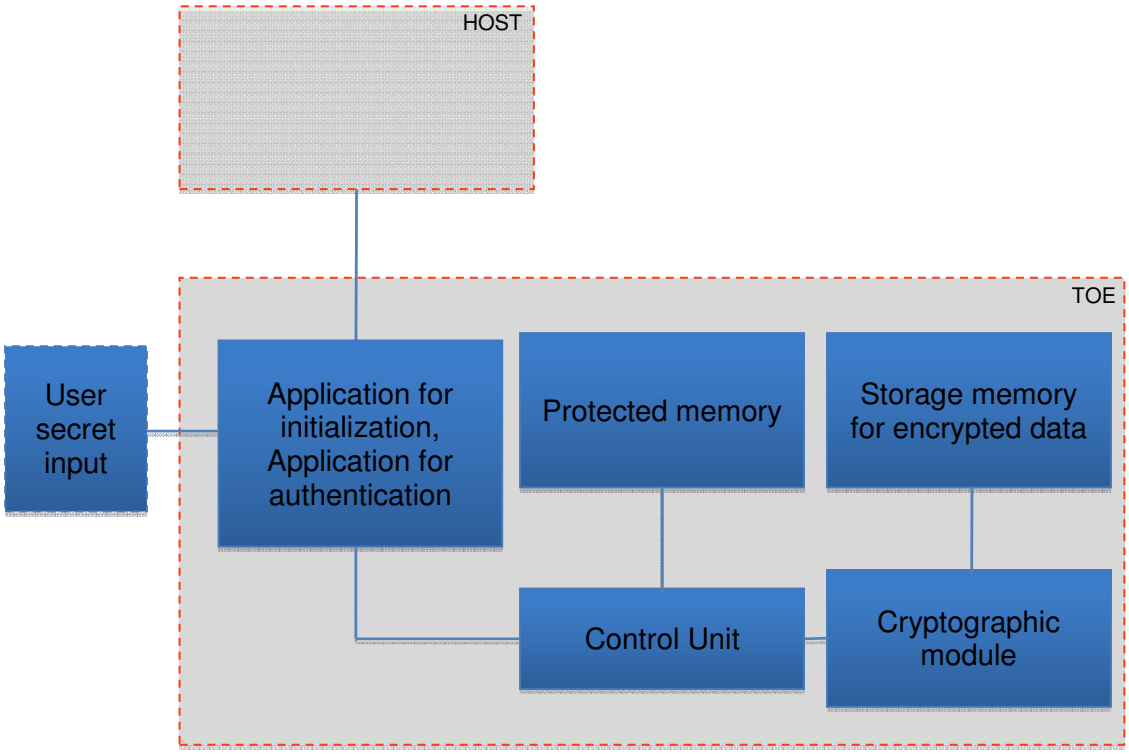

Figure 1 TOE architecture type A

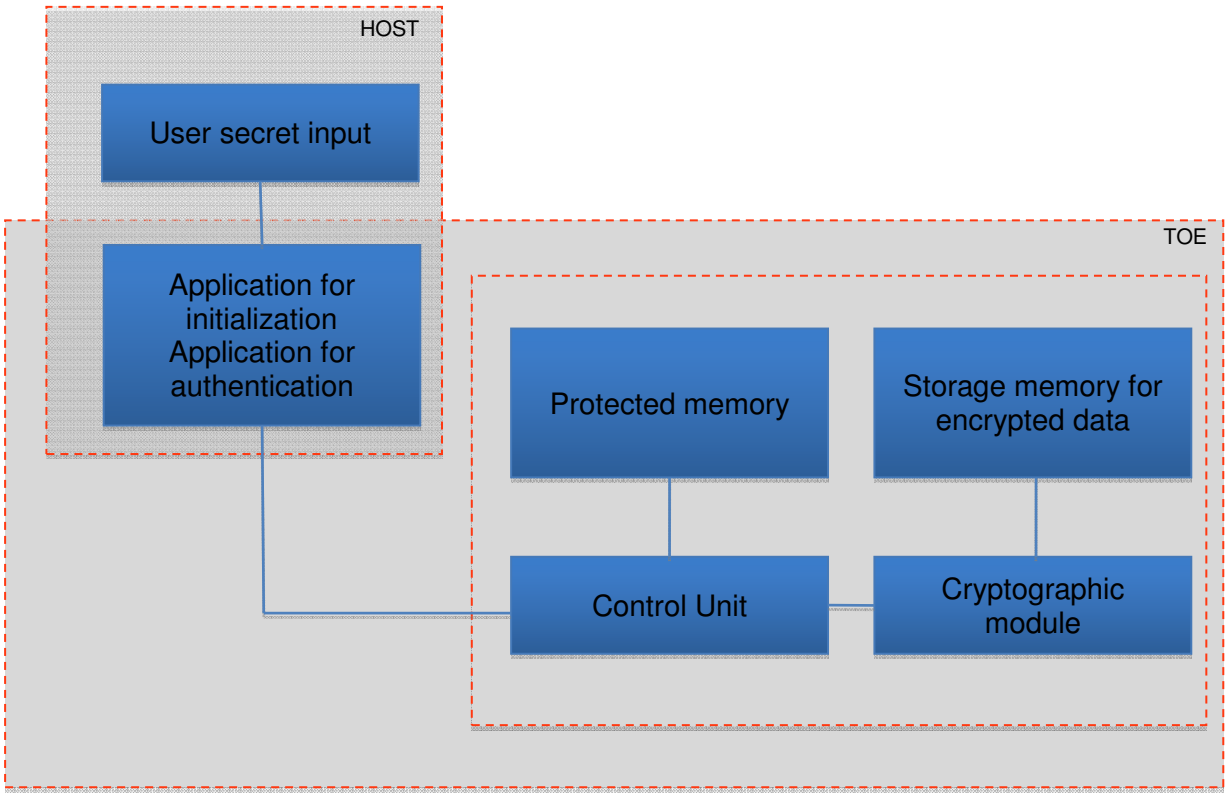

Figure 2 TOE architecture Type B

1.2.2 TOE interfaces

The interface to the Storage device architecture Type A is the host computer.

The interface to the Storage device architecture Type B is the Application for initialization and/or the Application for authentication, if they are placed in the host, and the User secret input in the host.

1.2.3 TOE components

The components of the TOE are as follows.

1.2.3.1 Protected memory

The Protected memory contains system files required for correct operation of the TOE. Files stored in the Protected memory are protected from being manipulated.

The Protected memory may be used to store the Application for initialization and the Application for authentication. The portion of this memory containing the Application for initialization and the Application for authentication may be exposed to the host computer.

The Protected memory shall be read-only with respect to the host. Any upgrade that could modify the Protected memory contents shall be signed by a trusted third party, to prevent the contents of the memory from being manipulated.

All confidential user data is encrypted using a Data Encryption Key (DEK) which is protected by a Key Encryption Key (KEK). The KEK is derived from the user secret using a Key Derivation Function (KDF).

The encrypted DEK and the salt used as input to the KDF may be stored in the Protected memory.

1.2.3.2 Storage memory

The Storage memory is persistent, read/writeable, and is used to store encrypted data blocks.

1.2.3.3 Control unit

The Control unit controls the information flow between the Storage device and the host. It also controls access to the Storage memory and the Protected memory.

1.2.3.4 Cryptographic module

The Cryptographic module manages all cryptographic operations, including random number generation, with the possible exception of the KDF if the user secret is input via an application executing on the host. The encrypted DEK and the salt used as input to the KDF may be stored in the Cryptographic module.

1.2.3.5 Application for authentication

The Application for authentication is executed by the user during authentication. The user proves knowledge of the user secret to the Storage device by using the Application for authentication.

The Application for authentication may be installed on the host computer or be executed directly from the Protected memory on the Storage device. The Application for authentication may be required to handle confidential information.

Such confidential information may be fed into the Application for authentication via peripheral input devices on the host computer, or via an input device directly connected to the Storage device.

1.2.3.6 Application for initialization

The Application for initialization is used to set up the key hierarchy. For additional information regarding initialization, see section 1.2.5.2.

The Application for initialization may be installed on the host computer or be executed directly from the Protected memory on the Storage device.

1.2.4 Intended usage

The TOE is personal and it is used only for temporary storage of data in encrypted personal storage devices whilst the data is in transit between two trusted host computers. The TOE users are trusted and trained in the handling of the TOE. The TOE users are authorized to access the confidential information. The TOE shall be used in a trusted environment by the legitimate user.

The data transported in the TOE can for example be presentations, minutes of meetings, e-mail, business plans and reports, governmental papers. The information is a copy of the information in the host system.

The Storage device shall be stored and handled in such a way that theft and manipulation is prevented. The Storage device shall be discarded in the event that the device is lost or stolen and later recovered, if it may be suspected that the device has been tampered with. The Storage device may have tamper protection measures to render tampering with the device more difficult. This is a recommendation and there are not any requirements for tamper protection in this PP.

The Storage device shall be distributed from the manufacturer to the end user in such a way that the device is not manipulated by a third party while it is in transit.

If an application on the trusted host computer is used to initialize or authenticate to the Storage device, the authenticity of the application shall be verified prior to it being installed or upgraded.

1.2.5 Security features

The TOE features security functions for key management, initialization, authentication, encryption, decryption and memory sanitization. There are security functions in the TOE for firmware upgrade if the firmware is upgradable. Optionally, there may be a manual switch for setting the TOE in read-only mode. This is a recommendation and there are not any requirements for a manual switch for setting the TOE in read-only mode in this PP.

1.2.5.1 Key management

All confidential information stored in persistent memory in the Storage device shall be encrypted under sufficiently strong keys using cryptographic algorithms approved for the intended usage by the relevant national authority.

All cryptographic operations shall be implemented in the Cryptographic module, with the possible exception of the KDF, which may be implemented in the Application for authentication executing on the host computer.

The KDF is used to derive the KEK from the user secret.

1.2.5.2 Initialization

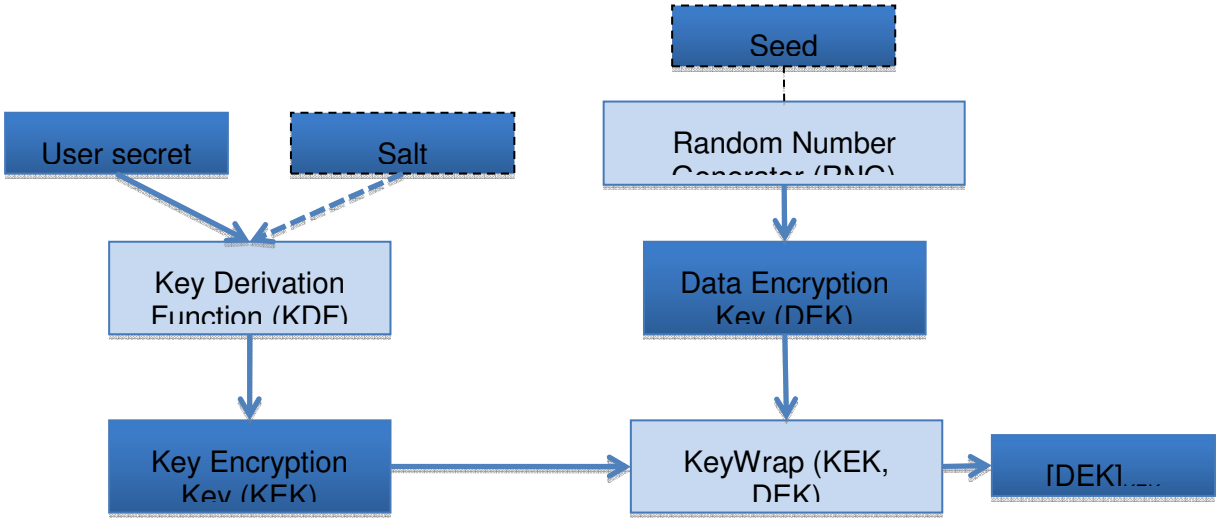

Figure 3 Initialization

Upon initialization of the Storage device, the user shall select the user secret such that it has sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries. The TOE should provide a mechanism to verify that the user secret meets a quality metric defined by an organizational security policy.

The DEK shall be generated by a Random Number Generator (RNG) implemented in the Cryptographic module in the Storage device. A seed can be input to the RNG, deterministic pseudo-random number generators derive pseudo-random bit sequences from a seed. The seed must contain sufficient entropy to render an exhaustive search computationally infeasible for the intended class of adversaries. The seed need not be uniformly distributed. The DEK generated

shall contain sufficient entropy to render an exhaustive search over all possible keys computationally infeasible for the intended class of adversaries.

The salt (if required by the KDF) and the user secret shall be fed as inputs to a KDF to generate the KEK. The DEK shall be encrypted under the KEK and the result, denoted by $[DEK]_{KEK}$, shall be stored in the Storage device.

Not all KDFs require a salt. If the KDF used requires a salt, it shall be unpredictable for the intended class of adversaries. The salt may be generated by an RNG and imported to the TOE, or may be generated and stored during production. The salt shall be stored in the Storage device. Since the salt is not secret, it may be stored in persistent memory.

The plaintext DEK and KEK shall never leave the Cryptographic module.

Additional entropy sources, random number generators, cryptographic mechanisms or keys may be introduced, provided it can be shown that the introduction of such sources, generators, mechanisms or keys do not adversely affect security.

It shall be possible to re-initialize an already initialized Storage device. Upon re-initialization of the Storage device, all encrypted keys are zeroized and the initialization procedure as described above is then performed over again.

1.2.5.3 Authentication

The user shall be required to authenticate prior to being allowed to perform any other action. During authentication, the user shall be required to provide the user secret.

The user secrets provided by the user, and the salt stored in the Storage device, are input to the KDF to generate the KEK. The KEK is then used to decrypt the encrypted DEK. Once the DEK has been decrypted, blocks of user data may be encrypted or decrypted at will.

If the Storage device is disconnected from the host, the session shall be closed and the user shall be required to re-authenticate in order to access data stored on the Storage device.

1.2.5.4 Encryption

When the host computer writes a block of data to the Storage device, the plaintext block shall be encrypted under the DEK by the Cryptographic module in the Storage device. The encrypted block shall then be written to the Storage memory in the Storage device.

1.2.5.5 Decryption

When a block is requested from the Storage device by the host computer, the stored block shall be retrieved from the Storage memory and decrypted under the DEK by the Cryptographic module in the Storage device. The decrypted plaintext block shall then be sent to the host computer.

1.2.5.6 Memory sanitization

The cryptographic keys and the user secret shall be securely zeroized as soon as the said keys and/or secret are no longer required for the correct operation of the TOE.

1.2.6 Optional security features

The Storage device may have an onboard upgradable firmware. If the firmware is upgradable, the firmware upgrade package shall be signed by a trusted party, and the authenticity of the signed package shall be verified by the Storage device prior to it performing any other action with respect to the upgrade package. The public key used for signature verification is generated and stored in the Storage device at production.

1.3 Additional TOE software, hardware, firmware

The TOE security functions are not dependant on components outside the TOE.

1.4 Abbreviations

$[DEK]_{KEK}$	Data Encryption Key (DEK) encrypted under Key Encryption Key (KEK)
CC	Common Criteria
DEK	Data Encryption Key
EAL	Evaluation Assurance Level
KDF	Key Derivation Function
KEK	Key Encryption Key
PP	Protection Profile
RNG	Random Number Generator
TOE	Target Of Evaluation

1.5 Glossary

Entropy	The total amount of information yielded by a set of bits. The entropy is representative of the work effort required for an adversary to be able to reproduce the same set of bits (ISO/IEC 18032). For a key of bit length n, the entropy lies between 0 and n, where the upper limit is attained if and only if every possible value for the key has equal probability.
Fuzz testing	Fuzz testing is a software testing technique, often automated or semi-automated, that involves providing invalid, unexpected, or random data to the inputs of a computer program. The program is then monitored for exceptions such as crashes, or failing built-in code assertions or for finding potential memory leaks. Fuzzing is commonly used to test for

	security problems in software or computer systems.
Key Derivation Function (KDF)	A Key Derivation Function (KDF) derives cryptographic keys from an input bit sequence containing entropy. The input bit sequence need not be uniformly distributed.
Persistent memory	A memory circuit such that the data stored in the circuit is retained when power is lost or cycled.
Random Number Generator (RNG)	A Random Number Generator (RNG) is an algorithm or a physical device that generates bit sequences. The bit sequences generated by the RNG shall be computationally indistinguishable from truly random bit sequences. Random Number Generators (RNGs) used for cryptographic applications produce a sequence of zero and one bits that may be combined into sub-sequences or blocks of random numbers. There are three principal methods used to generate random numbers: physical true RNG, non-physical true RNG, and deterministic RNG. A physical true RNG produces output that dependents on some physical random source inside the TOE boundary only. A non-physical true RNG gets its entropy from sources from outside the TOE boundary (e.g., by system data like RAM data or system time of a PC, output of API functions etc. or human interaction like key strokes, mouse movement etc.). A deterministic RNG consists of an algorithm that produces a sequence of bits from an initial random value (seed). Physical hybrid RNG and Deterministic hybrid RNG: The random number generator uses both methods to calculate the random number. E.g. it uses a true random number as s initial value (often refered to as "seed") for an algorithm, that generates pseudorandom numbers.
Salt	A salt is a value that is input to the KDF alongside the user secret for the purpose of preventing pre-computation attacks. A new salt shall be selected at random for each Storage device produced, in such a way that the salt may not be predicted by the intended class of adversaries.
Seed	Deterministic pseudo-random number generators derive pseudo-random bit sequences from a seed. The seed must contain sufficient entropy to render an exhaustive search computationally infeasible for the intended class of adversaries. The seed need not be uniformly distributed.
Target Of Evaluation (TOE)	The product or system being evaluated.

User secret	An authentication attribute (SA.User_Secret), such as a password, used to authorize the user to access the confidential data.
Volatile memory	A memory circuit such that the data stored in the circuit is not retained when power is lost or cycled.
Zeroization (of encryption key or secret)	A memory region is zeroized by actively overwriting the region. After a memory region has been zeroized it shall be infeasible to reconstruct the data or portions of the data originally stored in the memory region. When a secret is zeroized, all memory regions containing the secret or portions or derivatives thereof shall be zeroized, so that it is infeasible to reconstruct the secret.

2 Conformance claims

2.1 CC conformance claim

This protection profile claims conformance to:

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; CCMB-2009-07-001, Version 3.1, Revision 3, July 2009
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; CCMB-2009-07-002, Version 3.1, Revision 3, July 2009
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; CCMB-2009-07-003, Version 3.1, Revision 3, July 2009

as follows:

- This Protection Profile is Common Criteria Part 2 extended and Common Criteria Part 3 conformant.
- This Protection Profile claims conformance to assurance requirement package EAL2 augmented with ATE_COV.2.

2.2 PP claim

This Protection Profile does not claim conformance to any other Protection Profile.

2.3 Conformance statement

This PP requires demonstrable conformance by any ST or PP claiming conformance to this PP.

3 Security problem definition

3.1 Assets

The assets to protect are:

Table 2: Assets	
ID	Description
SA.User_Data	Confidential plaintext user data stored in or processed by the TOE.
SA.User_Secret	The secret used by the user when authenticating to the TOE.
SA.Keys	Plaintext encryption keys, e.g. the DEK and the KEK.

3.2 Threat agents

An attacker (threat agent) is a person who finds a lost Storage device or steals it (in powered off state) and performs attacks to the TOE. The main goal of the attacker is to access the confidential data (SA.User_Data). The attacker has basic attack potential.

3.3 Threats

Table 3: Threats	
ID	Description
T.Extract_User_Data	An attacker gains access to a Storage device when in a powered off state and is able to extract SA.User_Data, or parts thereof, in plaintext from a memory circuit.
T.Extract_User_Secret	An attacker gains access to a Storage device when in a powered off state, and is able to extract SA.User_Secret, or parts thereof, in plaintext from a memory circuit. An attacker gains access to the Application for authentication or the Application for initialization when the Storage device is disconnected and is able to extract SA.User_Secret in plaintext.
T.Extract_Keys	An attacker gains access to a Storage device when in a powered off state, and is able to extract SA.Keys, or correlated information, in plaintext from a memory circuit. If the KDF is executed in the Host computer: An attacker gains access to the Application for authentication or the

	Application for initialization when the Storage device is disconnected and is able to extract the KEK in plaintext.
T.Manipulation	An attacker gains temporary access to a Storage device which is in a powered off state, and manipulates the security functions in the Storage device so as to allow the attacker to extract or compute SA.User_Data in the future. Application note: The attack is possible only if the user authenticates to the Storage device after it has been manipulated, and if the attacker thereafter gains physical access to the Storage device for a second time.
T.Exhaustive_Search	An attacker gains access to a Storage device and attempts to compute SA.User_Secret or SA.Keys by mounting an exhaustive search.
Optional, this is applicable when firmware upgrades are possible: T.Malicious_Upgrade	An attacker upgrades or fools the TOE user into upgrading the TOE with a malicious firmware or software upgrade for the purpose of manipulating the security functions in the TOE so as to allow the attacker to extract or compute SA.User_Data in the future. Application note: The attack is possible only if the user authenticates to the Storage device after it has been manipulated, and if the attacker thereafter gains physical access to the TOE.

3.4 Assumptions

Table 4: Assumptions	
ID	Description
A.Trusted_Users	The TOE users are trusted and trained in the handling of the TOE. The TOE users are authorized to access the confidential information.
A.Trusted_Host	The host computers are trusted. Only authorized users have access to the host computers. The host computer environment is trusted.
A.Legitimate_Usage	The TOE shall be under the control of the trusted user in a trusted environment.
A.Lost_Storage_Device	The Storage device shall be discarded in the event that the device is lost or stolen and later recovered, if it may be suspected that an unauthorized party has tampered with the device.
Optional, this is required	The firmware upgrade signing key is handled by a trusted party and

when firmware upgrades are possible: A.Signing_Key	is used exclusively to sign approved firmware upgrade packages. The trusted party handles the key so that it is protected against disclosure and manipulation. The key is generated using a method which provides sufficiently strong keys to prevent cryptanalysis of the private signing keys by the intended class of adversaries.
Optional, this is required when the Application for initialization and the Application for authentication are executed on the host: A.Application_for_initialization_and_for_authentication	The authenticity of the Application for initialization and Application for authentication shall be verified prior to them being installed or upgraded. A procedure whereby the user can verify the authenticity of the applications shall be established.

3.5 Organizational security policies

Table 5: Organizational security policies

ID	Description
P.Entropy	The user secret (SA.User_Secret) shall have sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries.
P.Crypto	The cryptographic keys shall have sufficient strength to render an exhaustive search with respect to the cryptographic keys computationally infeasible for the intended class of adversaries.

4 Security objectives

4.1 Security objectives for the TOE

Table 6: Objectives for the TOE	
ID	Description
O.Encrypted_Information	The TOE will ensure that SA.User_Data and SA.Keys in the Storage memory are encrypted in accordance with this PP. The TOE will ensure that SA.User_Data, SA.Keys and SA.User_Secret are not stored in plain text in persistent memory. Neither shall it be possible to reconstruct SA.Keys or SA.User_Secret from information stored in persistent memory.
O.Authentication	The TOE will ensure that SA.User_Data is not available in plain text until user authentication is successfully performed.
O.Key_Derivation	The TOE will ensure that the KDF is sufficiently complex to render an exhaustive search over the space of all possible inputs computationally infeasible. If the KDF used requires a salt, the salt shall be generated in such a way that it is unpredictable for the intended class of adversaries.
O.Key_Generation	The TOE will ensure that generation of the DEK has the property that it does not result in more efficient ways of breaking the encryption than if the DEK was completely random.
Optional, this is required when firmware upgrades are possible: O.Firmware_Upgrade	The firmware upgrade package shall be signed by a trusted party, and the authenticity of the signed package shall be verified by the Storage device prior to it performing any other action with respect to the upgrade package.
O.Key_Zeroization	The TOE shall ensure that SA.Keys are securely zeroized when they are no longer needed for the correct operation of the TOE. Application note: The zeroization shall ensure that the adversary is not able to recreate data that has been zeroized.
O.Secret_Zeroization	The TOE shall ensure that SA.User_Secret is securely zeroized when it is no longer needed for the correct operation of the TOE. Application note: The zeroization shall ensure that the adversary is not able to recreate data that has been zeroized.

4.2 Security objectives for the environment

Table 7: Objectives for the environment

ID	Description
OE.Trusted_Users	The TOE users are trusted and trained in the handling of the TOE. The TOE users are authorized to access the confidential information.
OE.Trusted_Host	The host computers are trusted. Only authorized users have access to the host computers.
OE.Legitimate_Usage	The TOE shall be under the control of the trusted user in a trusted environment.
OE.Lost_Storage_Device	The Storage device shall be discarded in the event that the device is lost or stolen and later recovered, if it may be suspected that an unauthorized party has tampered with the device.
OE.Entropy	The user secret (SA.User_Secret) shall have sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries.
OE.Crypto	The cryptographic keys shall have sufficient strength to render an exhaustive search with respect to the cryptographic keys computationally infeasible for the intended class of adversaries.
Optional, this is required when firmware upgrades are possible: OE.Signing_Key	The firmware upgrade signing key is handled by a trusted party and is used exclusively to sign approved firmware upgrade packages. The trusted party handles the key so that it is protected against disclosure and manipulation.
Optional, this is required when the Application for initialization and/or the Application for authentication are executed on the host: OE.Application_for_initialization_and_for_authentication	The authenticity of the Application for initialization and/or Application for authentication shall be verified prior to them being installed or upgraded. A procedure whereby the user can verify the authenticity of the applications shall be established.

4.3 Rationales

4.3.1 Threats - Security objectives

Table 8: Threats – Security objectives	
Threat	Security objective
T.Extract_User_Data	O.Encrypted_Information O.Authentication
T.Extract_User_Secret	O.Encrypted_Information O.Secret_Zeroization
T.Extract_Keys	O.Encrypted_Information O.Key_Zeroization
T.Manipulation	OE.Lost_Storage_Device
T.Exhaustive_Search	O.Encrypted_Information O.Key_Derivation O.Key_Generation OE.Entropy OE.Crypto
Optional, this is applicable when firmware upgrades are possible: T.Malicious_Upgrade	Optional, this is required when firmware upgrades are possible: O.Firmware_Upgrade

T.Extract_User_Data is met by O.Encrypted_Information, which ensures that SA.User_Data is not stored in plaintext in persistent memory, supported by O.Authentication, which ensures that SA.User_Data is not available in plaintext until user authentication is successfully performed.

T.Extract_User_Secret is met by O.Encrypted_Information, which ensures that SA.User_Secret is not stored in plaintext in persistent memory, supported by O.Secret_Zeroization, which ensures that SA.User_Secret is securely zeroized when it is no longer needed for the correct operation of the TOE.

T.Extract_Keys is met by O.Encrypted_Information, which ensures that SA.Keys is not stored in plaintext in persistent memory, supported by O.Key_Zeroization which ensures that SA.Keys are securely zeroized when they are no longer needed for the correct operation of the TOE.

T.Manipulation is met by OE.Lost_Storage_Device, which ensures that if the Storage device is lost or stolen and later recovered, it shall be discarded if it may be suspected that an unauthorized party has tampered with the device.

T.Exhaustive_Search is met by O.Encrypted_Information which ensures that all SA.User_Data and SA.Keys in the Storage memory are encrypted, supported by OE.Entropy which ensures that SA.User_Secret has sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries, OE.Crypto which ensures that cryptographic keys has sufficient strength to render an exhaustive search with respect to the cryptographic keys computationally infeasible for the intended class of adversaries, O.Key_Derivation which ensures that the KDF is sufficiently complex to render an exhaustive search over the space of all possible inputs computationally infeasible and O.Key_Generation which ensures that generation of the DEK has the property that it does not result in more efficient ways of breaking the encryption than if the DEK was completely random.

Optional, this is applicable when firmware upgrades are possible:

T.Malicious_Upgrade is met by O.Firmware_Upgrade, which ensures that the authenticity of the signed package is verified by the Storage device prior to it performing any other action with respect to the upgrade package.

4.3.2 Assumptions – Security objectives

Table 9: Assumption – Security objective	
Assumption	Security objective
A.Trusted_Users	OE.Trusted_Users
A.Trusted_Host	OE.Trusted_Host
A.Legitimate_Usage	OE.Legitimate_Usage
A.Lost_Storage_Device	OE.Lost_Storage_Device
Optional, this is required when firmware upgrades are possible: A.Signing_Key	Optional, this is required when firmware upgrades are possible: OE.Signing_Key
Optional, this is applicable when the Application for initialization and/or the Application for authentication are executed on the host: A.Application_for_initialization_and_for_authentication	Optional, this is required when the Application for initialization and/or the Application for authentication are executed on the host: OE.Application_for_initialization_and_for_authentication

A.Trusted_Users is met by OE.Trusted_Users, which ensures that the TOE users are trusted and trained in the handling of the TOE, and that they are authorized to access the confidential information.

A.Trusted_Host is met by OE.Trusted_Host, which ensures that the host computers are trusted; that only authorized users have access to the host computers and that the host computer environment is trusted.

A.Legitimate_Usage is met by OE.Legitimate_Usage, which ensures that the TOE is under the control of the trusted user in a trusted environment.

A.Lost_Storage_Device is met by OE.Lost_Storage_Device, which ensures that if the Storage device is lost or stolen and later recovered, it shall be discarded if it may be suspected that an unauthorized party has tampered with the device.

Optional, this is applicable when firmware upgrades are possible:

A.Signing_Key is met by OE.Signing_Key which ensures that the firmware upgrade signing key is handled by a trusted party and is used exclusively to sign approved firmware upgrade packages. The trusted party handles the key so that it is protected against disclosure and manipulation.

Optional, this is applicable when the Application for initialization and/or the Application for authentication are executed on the host:

A.Application_for_initialization_and_for_authentication is met by OE.Application_for_initialization_and_for_authentication, which ensures that the authenticity of the Application for initialization and the Application for authentication are verified prior to them being installed or upgraded, and that a procedure whereby the user can verify the authenticity of the applications is established.

4.3.3 Organizational Security Policy – Security objectives

Table 10: OSP – Security Objective	
OSP	Security objective
P.Entropy	OE.Entropy
P.Crypto	OE.Crypto

P.Entropy is met by OE.Entropy, which ensures that SA.User_Secret has sufficient entropy to render an exhaustive search with respect to the user secret computationally infeasible for the intended class of adversaries.

P.Crypto is met by OE.Entropy which ensures that the cryptographic keys has sufficient strength to render an exhaustive search with respect to the cryptographic keys computationally infeasible for the intended class of adversaries

4.3.4 Security objectives – Threat – Assumption – OSP

Table 11: Security objectives – Threat – Assumption – OSP			
Security objective	Threat	Assumption	OSP
O.Encrypted_Information	T.Extract_User_Data T.Extract_User_Secret T.Extract_Keys T.Exhaustive_Search	-	-
O.Authentication	T.Extract_User_Data	-	-
O.Key_Derivation	T.Exhaustive_Search	-	-
O.Key_Generation	T.Exhaustive_Search	-	-
O.Key_Zeroization	T.Extract_Keys	-	-
O.Secret_Zeroization	T.Extract_User_Secret	-	-

Optional, this is required when firmware upgrades are possible: O.Firmware_Upgrade	Optional, this is applicable when firmware upgrades are possible: T.Malicious_Upgrade	-	-
OE.Trusted_Users	-	A.Trusted_Users	-
OE.Trusted_Host	-	A.Trusted_Host	-
OE.Legitimate_Usage	-	A.Legitimate_Usage	-
OE.Lost_Storage_Device	T.Manipulation	A.Lost_Storage_Device	-
OE.Entropy	T.Exhaustive_Search	-	P.Entropy
OE.Crypto	T.Exhaustive_Search	-	P.Crypto
Optional, this is required when firmware upgrades are possible: OE.Signing_Key	-	Optional, this is required when firmware upgrades are possible: A.Signing_Key	-
Optional, this is required when the Application for initialization and/or the Application for authentication are executed on the host: OE.Application_for_initialization_and_for_authentication	-	Optional, this is required when the Application for initialization and the Application for authentication are executed on the host: A.Application_for_initialization_and_for_authentication	-

5 Extended components definition

5.1 FCS_RNG Generation of random numbers

FCS_RNG.1 Generation of random numbers requires that the random number generator implements defined security capabilities and that the random numbers meet a defined quality metric.

5.1.1 Family Behaviour

This family defines quality requirements for the generation of random numbers that are intended to be used for cryptographic purposes.

5.1.2 Component leveling

FCS_RNG.1 is not hierarchical to any other component within the FCS_RNG family.

5.1.3 Management

There are no management activities foreseen.

5.1.4 Audit

There are no actions defined to be auditable.

5.1.5 FCS_RNG.1 Random number generation

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_RNG.1.1 The TSF shall provide a [selection: physical, non-physical true, deterministic, physical hybrid, deterministic hybrid] random number generator that implements: [assignment: list of security capabilities].

FCS_RNG.1.2 The TSF shall provide random numbers that meet [assignment: a defined quality metric].

5.1.6 Rationale

The quality of the random number generator is defined using this SFR.

6 IT Security Requirements

6.1 Information flow control policies

6.1.1 User data access control SFP

Subject: Host

Objects: SA.User_Data

Attributes: Data identified as being encrypted or decrypted

Rules to apply: SA.User_Data on the Storage device shall only be accessible after it has been decrypted.

6.1.2 User secret access control SFP

6.1.2.1 TOE architecture type A

Subject: Host

Objects: SA.User_Secret

Attributes: Data identified as being SA.User_Secret

Rules to apply: SA.User_Secret on the Storage shall not be accessible from the host.

6.1.2.2 TOE architecture type B

Subject: Host or the Application for initialization and the Application for authentication on the host

Objects: SA.User_Secret

Attributes: Data identified as being SA.User_Secret

Rules to apply: SA.User_Secret on the storage device shall not be accessible to the host or to the Application for initialization and the Application for authentication.

6.2 Security Functional Requirements

6.2.1 Cryptographic support

6.2.1.1 FCS_CKM.1_DEK Cryptographic key generation

FCS_CKM.1.1	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [assignment: cryptographic key generation algorithm] and specified cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*].

Application note: The TSF shall generate the DEK using an RNG implemented in the Cryptographic module in the TOE. The DEK generated shall contain sufficient entropy to render an exhaustive search over all possible keys computationally infeasible for the intended class of

adversaries.

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.1.2 FCS_CKM.1_KEK Cryptographic key generation

FCS_CKM.1.1	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [assignment: *cryptographic key generation algorithm*] and specified cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*].

Application note: The TSF shall derive KEK with a KDF implemented in the TOE.

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.1.3 FCS_RNG.1 Random number generation

FCS_RNG.1.1	The TSF shall provide a [selection: physical, non-physical true, deterministic, physical hybrid, deterministic hybrid] random number generator that implements: [assignment: *list of security capabilities*].
FCS_RNG.1.2	The TSF shall provide random numbers that meet [assignment: *a defined quality metric*].

6.2.1.4 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1	The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method: **zeroization** that meets the following: [assignment: *list of standards*]**.**

Application note: SA.Keys shall be securely zeroized as soon as the keys are no longer required for the correct operation of the TOE.

Application note: Upon re-initialization of the Storage device, all encrypted keys shall be zeroized and the initialization procedure shall then be performed over again.

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.1.5 FCS_COP.1_Data Cryptographic operation

FCS_COP.1.1	The TSF shall perform **encryption and decryption of confidential data** in accordance with a specified cryptographic algorithm [assignment: *cryptographic algorithm*] and cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following:

[assignment: *list of standards*].

Application note: The TSF shall perform encryption of SA.User_Data to be stored in persistent memory in the Storage device, and decryption of SA.User_Data when the user is authenticated.

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.1.6 FCS_COP.1_Key Cryptographic operation

FCS_COP.1.1 The TSF shall perform **key encryption and decryption** in accordance with a specified cryptographic algorithm [assignment: *cryptographic algorithm*] and cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*].

Application note: The KEK is used to encrypt and decrypt the DEK.

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.1.7 FCS_COP.1_Signature_Verification Cryptographic operation

Optional, this is required when firmware upgrades are possible:

FCS_COP.1.1 The TSF shall perform **verification of the firmware upgrade package signature** in accordance with a specified cryptographic algorithm [assignment: *cryptographic algorithm*] and cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards].*

Application note: All cryptographic algorithms and schemes used in the Storage device shall be approved by the relevant national authorities in the countries where the device is to be used.

6.2.2 User data protection

6.2.2.1 FDP_ETC.1 Export of user data without security attributes

FDP_ETC.1.1 The TSF shall enforce the **User data access control SFP** when exporting user data, controlled under the SFP(s), outside of the TOE.

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

6.2.2.2 FDP_ACC.1_User_Data Subset access control

FDP_ACC.1.1 The TSF shall enforce the **User data access control SFP** on **Host access to SA.User_Data on the Storage device to ensure that SA.User_Data is accessible by the Host only after it has been**

decrypted.

6.2.2.3 FDP_ACC.1_User_Secret Subset access control

FDP_ACC.1.1	The TSF shall enforce the **User secret access control SFP** on **Host access to SA.User_Secret on the Storage device to ensure that SA.User_Secret is not accessible by the host or the Application for initialization and the Application for authentication on the host.**

Application note: The TSF shall ensure SA.User_Secret on the TOE is not accessible from the host (TOE architecture type A) or to the Application for initialization and the Application for authentication (TOE architecture type B).

6.2.2.4 FDP_ACF.1_User_Data Secure attribute based access control

FDP_ACF.1.1	The TSF shall enforce the **User data access control SFP** to objects based on the following: **Subject: Host, Object: SA.User_Data, Security attribute: data identified as being encrypted or decrypted**.
FDP_ACF.1.2	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **host access to SA.User_Data will only be allowed for data being identified as being decrypted**.
FDP_ACF.1.3	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **no additional rules**.
FDP_ACF.1.4	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **no additional rules**.

Application note: Decrypted information means information that has been decrypted by the TOE. Such information may have been encrypted by other means outside of the TOE, but from the point of the TOE it is considered as being decrypted once it has been decrypted by the TOE.

6.2.2.5 FDP_ACF.1_User_Secret Security attribute based access control

FDP_ACF.1.1	The TSF shall enforce the **User secret access control SFP** to objects based on the following: **Subject: Host, Object: SA.User_Secret, Security attribute: data identified as being SA.User_Secret**.
FDP_ACF.1.2	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **host access to SA.User_Secret will never be allowed for data being identified as being SA.User_Secret**.
FDP_ACF.1.3	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **no additional rules**.
FDP_ACF.1.4	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **no additional rules**.

Application note: Note that host access in FDP_ACF.1.2 includes access from the host (TOE architecture type A) as well as access from the Application for initialization and the Application for authentication (TOE architecture type B). They are both considered host access since in TOE architecture type B the Application for initialization and the Application for authentication are both located on the host. Note also that the SA.User_Secret exists only temporary in the TOE. FDP_RIP.1 ensures that SA.User_Secret is zeorized when no longer needed within the TOE.

6.2.2.6 FDP_RIP.1 Subset residual information protection

FDP_RIP.1.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the **deallocation of the resource from** the following objects: **SA.User_Secret.**

Application note: The TSF shall ensure that SA.User_Secret is securely zeroized when it is no longer needed for correct operation of the TSF. The zeroization shall ensure that the adversary is not able to recreate data that has been zeroized.

6.2.3 Identification and authentication

6.2.3.1 FIA_UAU.2 User authentication before any action

FIA_UAU.2.1	The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application note: During authentication, the user shall be required to provide SA.User_Secret.

6.2.4 Security management

6.2.4.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: **user selects SA.User_Secret, user performs re-initialization**.

Application note: There is only one TOE user.

6.3 Dependencies between functional components

Table 12: Dependencies between functional components

Security functional component	Dependencies	Note
FCS_CKM.1_DEK	FCS_CKM.2 or FCS_COP.1	Yes FCS_COP.1_Key FCS_COP.1_Data
	FCS_CKM.4	Yes
FCS_CKM.1_KEK	FCS_CKM.2 or FCS_COP.1	Yes FCS_COP.1_Key
	FCS_CKM.4	Yes
FCS_RNG.1	No	-
FCS_CKM.4	FDP_ITC.1 or FDP_ITC.2, or FCS_CKM.1	Yes FCS_CKM.1_DEK FCS_CKM.1_KEK
FCS_COP.1_Data	FDP_ITC.1 or FDP_ITC.2, or FCS_CKM.1	Yes FCS_CKM.1_DEK
	FCS_CKM.4	Yes
FCS_COP.1_Key	FDP_ITC.1 or FDP_ITC.2, or FCS_CKM.1	Yes FCS_CKM.1_DEK FCS_CKM.1_KEK
	FCS_CKM.4	Yes

Optional, this is required when firmware upgrades are possible: FCS_COP.1_Signature_Verification	FDP_ITC.1 or FDP_ITC.2, or FCS_CKM.1 and FCS_CKM.4	No FDP_ITC.1 or FDP_ITC.2 are not applicable as the key used for signature verification is generated and stored in the Storage device at production. FCS_CKM.1 is not applicable because the TOE does not generate any keys for signature verification. FCS_CKM.4 is not applicable since the key used for signature verification is public.
FDP_ETC.1	FDP_ACC.1 or FDP_IFC.1	Yes FDP_ACC.1_User_Data
FDP_ACC.1_User_Data	FDP_ACF.1	Yes FDP_ACF.1_User_Data
FDP_ACC.1_User_Secret	FDP_ACF.1	Yes FDP_ACF.1_User_Secret.
FDP_ACF.1_User_Data	FDP_ACC.1 and FMT_MSA.3	Yes by FDP_ACC.1_User_Data but not to FMT_MSA.3. FMT_MSA.3 is not applicable since the attributes cannot be initialized or managed.
FDP_ACF.1_User_Secret	FDP_ACC.1 and FMT_MSA.3	Yes by FDP_ACC.1_User_Secret but not to FMT_MSA.3. FMT_MSA.3 is not applicable since the attributes cannot be initialized or managed.
FDP_RIP.1	No	-
FIA_UAU.2	FIA_UID.1	No FIA_UID.1 is not applicable. It is only supposed to be one user of the TOE and therefore there is no need for identification, only authentication.

FMT_SMF.1	No	-

6.4 Security Assurance Requirements

The assurance requirements for the evaluation of the TOE, its development and operating environment is the predefined assurance package EAL2 augmented with ATE_COV.2.

Two refinements have been made to the assurance requirements of the EAL2 package. They apply to the ADV_TDS.1 and AVA_VAN.2.

The refinements have been done by reproducing the original SAR of the EAL2 package and mark the refinements in bold.

6.4.1 Refinement of ADV_TDS.1

Developer action elements:

ADV_TDS.1.1D	The developer shall provide the design of the TOE.
ADV_TDS.1.2D	The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.
ADV_TDS.1.3D	**The developer shall make available the parts of the implementation representation of the TSF implementing the FCS SFRs.**
ADV_TDS.1.4D	**The developer shall provide a mapping between the TOE design description and the parts of the implementation representation of the TSF implementing the FCS SFRs.**

Content and presentation elements:

ADV_TDS.1.1C	The design shall describe the structure of the TOE in terms of subsystems.
ADV_TDS.1.2C	The design shall identify all subsystems of the TSF.
ADV_TDS.1.3C	The design shall describe the behaviour of each SFR-supporting or SFR-non-interfering TSF subsystem in sufficient detail to determine that it is not SFR-enforcing.
ADV_TDS.1.4C	The design shall summarise the SFR-enforcing behaviour of the SFR-enforcing subsystems.
ADV_TDS.1.5C	The design shall provide a description of the interactions among SFR-enforcing subsystems of the TSF, and between the SFR-enforcing subsystems of the TSF and other subsystems of the TSF.

ADV_TDS.1.6C	The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.
ADV_TDS.1.7C	**The design shall provide a mapping from the subsystems of the TSF that are enforcing the SFRs of the class FCS to the modules of the TSF.**
ADV_TDS.1.8C	**The design shall describe each SFR-enforcing module of the class FCS in terms of its purpose and relationship with other modules.**
ADV_TDS.1.9C	**The design shall describe each SFR-enforcing module of the class FCS in terms of its SFR-related interfaces, return values from those interfaces, interaction with other modules and called SFR-related interfaces to other SFR- enforcing modules.**
ADV_TDS.1.10C	**The implementation representation shall define the TSF to a level of detail such that the TSF can be generated without further design decisions.**
ADV_TDS.1.11C	**The implementation representation shall be in the form used by the development personnel.**
ADV_TDS.1.12C	**The mapping between the TOE design description and the sample of the implementation representation shall demonstrate their correspondence for the parts of the implementation representation of the TSF implementing the FCS SFRs.**

Evaluator action elements:

ADV_TDS.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_TDS.1.2E	The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.
ADV_TDS.1.3E	**The evaluator shall confirm that, for the selected part of the implementation representation, the information provided meets all requirements for content and presentation of evidence.**

6.4.2 Refinement of AVA_VAN.2

Developer action elements:

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

Content and presentation elements:

AVA_VAN.2.1C	The TOE shall be suitable for testing.

Evaluator action elements:

AVA_VAN.2.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_VAN.2.2E	The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.
AVA_VAN.2.3E	The evaluator shall perform an independent vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, **implementation representation** and security architecture description to identify potential vulnerabilities in the TOE.
AVA_VAN.2.4E	The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.
AVA_VAN.2.5E	**The evaluator shall perform fuzzy testing of the TOE interface to the host, and that these fuzzy tests cover all cryptographic primitives visible on that interface.**

6.5 Dependencies between the assurance components

Since all dependencies within the assurance package EAL2 have been resolved, this table only identifies any dependencies from the augmentations made.

Table 13: Dependencies between assurance components		
Security assurance component	Dependencies	Note
ATE_COV.2	ADV_FSP.2	Yes (as part of EAL2)
	ATE_FUN.1	Yes (as part of EAL2)

6.6 Security requirements rationale

6.6.1 Security functional requirements rationale

Table 14: SFR rationale	
Security objective	SFR
O.Encrypted_Information	FCS_COP.1_Data

	FCS_COP.1_Key FCS_CKM.1_KEK FIA_UAU.2 FDP_ACC.1_User_Secret FDP_ACC.1_User_Data FDP_ACF.1_User_Secret FDP_ACF.1_User_Data FDP_ETC.1
O.Authentication	FIA_UAU.2 FCS_CKM.1_KEK FCS_COP.1_Key FCS_COP.1_Data FMT_SMF.1
O.Key_Derivation	FCS_CKM.1_KEK
O.Key_Generation	FCS_CKM.1_DEK FCS_RNG.1
Optional, this is required when firmware upgrades are possible: O.Firmware_Upgrade	FCS_COP.1_Signature_Verification (Optional)
O.Key_Zeroization	FCS_CKM.4
O.Secret_Zeroization	FDP_RIP.1

O.Encrypted_Information is ensured by:
FCS_COP.1_Data ensures encryption of SA.User_Data, FCS_COP.1_Key ensures that the DEK is encrypted with the KEK and stored in the TOE. The KEK is not stored in the TOE. The KEK is generated with FCS_CKM.1_KEK every time during the TOE user authentication FIA_UAU.2.

FDP_ACC.1_User_Secret defines rules for SA.User_Secret transfer. Together with FDP_ACF.1_User_Secret it ensures that SA.User_Secret is not accessible from the host (TOE architecture type A) or the host, including the the Application for Initialization and authentication (TOE architecture type B).

FDP_ACC.1_User_Data defines rules for SA.User_Data is not accessible out of the TOE. Together with FDP_ACF.1_User_Data which ensures that only decrypted SA.User_Data is accessible for the host. FDP_ETC.1 ensures that SA.User_Data can be exported from the TOE.

O.Authentication is ensured by:
FIA_UAU.2 ensures that the TOE user is authenticated before any TSF-mediated action. The user is required to provide SA.User_Secret. SA.User_Secret is then used as input to the KDF to generate the KEK as defined in FCS_CKM.1_KEK. The generated KEK is then used to decrypt the DEK as specified in FCS_COP.1_Key. After the DEK is successfully decrypted, SA.User_Data stored in the TOE may be decrypted as specified in FCS_COP.1_Data. The TOE user selects a SA.User_Secret upon initialization as defined in FMT_SMF.1.

O.Key_Derivation is ensured by:
FCS_CKM.1_KEK ensures that the KDF is sufficiently complex to render an exhaustive search over the space of all possible inputs computationally infeasible.

O.Key_Generation is ensured by:
The DEK is generated as defined in FCS_CKM.1_DEK by a Random Number Generator (RNG) that is defined in FCS_RNG.1.

O.Firmware_Upgrade (optional) is ensured by:
FCS_COP.1_Signature_Verification ensures that signature verification is performed with respect to the upgrade package. This is required when firmware upgrades are possible.

O.Key_Zeroization is ensured by:
FCS_CKM.4 ensures that SA.Keys are securely zeroized when they are no longer needed for correct operation of the TOE.

O.Secret_Zeroization is ensured by:
FDP_RIP.1 ensures that SA.User_Secret is securely zeroized when it is no longer needed for correct operation of the TOE.

6.6.2 Security assurance requirements rationale

The assurance package EAL2 has been augmented with ATE_COV.2. EAL2 is applicable in those circumstances where developers or users require a low to moderate level of independently assured security in the absence of ready availability of the complete development record. The ATE_COV.2 augmentation has been made to ensure full test coverage of all TSFIs.

To address the need for vulnerability analysis of the cryptographic functions, refinements have been made to the ADV_TDS.1 to provide additional information of the design information and implementation representation of the cryptographic functions. This information will be used when performing AVA_VAN.2. Also AVA_VAN.2 has been refined to specify specific testing as required by MSB.