Supporting Document
Mandatory Technical Document

PP-Module for File Encryption

Version: 1.0

2019-07-25

National Information Assurance Partnership

Foreword

This is a Supporting Document (SD), intended to complement the Common Criteria version 3 and the associated Common Evaluation Methodology for Information Technology Security Evaluation.

SDs may be “Guidance Documents”, that highlight specific approaches and application of the standard to areas where no mutual recognition of its application is required, and as such, are not of normative nature, or “Mandatory Technical Documents”, whose application is mandatory for evaluations whose scope is covered by that of the SD. The usage of the latter class is not only mandatory, but certificates issued as a result of their application are recognized under the CCRA.

Technical Editor:
National Information Assurance Partnership (NIAP)

Document history:

Version	Date	Comment
v 1.0	2019-07-25	Initial Release

General Purpose:
The purpose of this SD is to define evaluation methods for the functional behavior of file encryption products.

Acknowledgements:
This SD was developed with support from NIAP File Encryption Technical Community members, with representatives from industry, Government agencies, Common Criteria Test Laboratories, and members of academia.

Table of Contents

1 Introduction

1.1 Technology Area and Scope of Supporting Document

The scope of the File Encryption PP-Module is to describe the security functionality of a file encryption in terms of [CC] and to define functional and assurance requirements for them. The PP-Module is intended for use with the Application Software Protection Profile.

This SD is mandatory for evaluations of TOEs that claim conformance to a PP-Configuration that includes the PP-Module for File Encryption, Version 1.0. Although Evaluation Activities are defined mainly for the evaluators to follow, in general they also help Developers to prepare for evaluation by identifying specific requirements for their TOE. The specific requirements in Evaluation Activities may in some cases clarify the meaning of Security Functional Requirements (SFR), and may identify particular requirements for the content of Security Targets (ST) (especially the TOE Summary Specification), user guidance documentation, and possibly supplementary information (e.g. for entropy analysis or cryptographic key management architecture).

1.2 Structure of the Document

Evaluation Activities can be defined for both SFRs and Security Assurance Requirements (SAR), which are themselves defined in separate sections of the SD.

If any Evaluation Activity cannot be successfully completed in an evaluation then the overall verdict for the evaluation is a 'fail'. In rare cases there may be acceptable reasons why an Evaluation Activity may be modified or deemed not applicable for a particular TOE, but this must be approved by the Certification Body for the evaluation.

In general, if all Evaluation Activities (for both SFRs and SARs) are successfully completed in an evaluation then it would be expected that the overall verdict for the evaluation is a ‘pass’. To reach a ‘fail’ verdict when the Evaluation Activities have been successfully completed would require a specific justification from the evaluator as to why the Evaluation Activities were not sufficient for that TOE.

Similarly, at the more granular level of Assurance Components, if the Evaluation Activities for an Assurance Component and all of its related SFR Evaluation Activities are successfully completed in an evaluation then it would be expected that the verdict for the Assurance Component is a ‘pass’. To reach a ‘fail’ verdict for the Assurance Component when these Evaluation Activities have been successfully completed would require a specific justification from the evaluator as to why the Evaluation Activities were not sufficient for that TOE.

1.3 Terms

Common Criteria Terms

The following definitions are for Common Criteria terms used in this document:

Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation.
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Protection Profile Configuration (PP-Configuration)	A comprehensive set of security requirements for a product type that consists of at least one Base-PP and at least one PP-Module.
Protection Profile Module (PP-Module)	An implementation-independent statement of security needs for a TOE type complementary to one or more Base Protection Profiles.
Security Assurance Requirement (SAR)	A requirement to assure the security of the TOE.
Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.

Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
Target of Evaluation (TOE)	The product under evaluation. In this case, file encryption software and its supporting documentation.
TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in a ST.

Technical Terms

The following definitions define Technical terms used in this document:

Administrator	Authorized Users with higher privileges and typically handle configuration and management functions, such as configuring and updating the TOE.
Authorization factor (AF)	A value submitted by the user, present on the host, or present on a separate protected hardware physical device used to establish that the user (and potentially the host) is in the community authorized to use the TOE. The authorization factors are used to generate the KEK. Note that these AFs are not used to establish the particular identity of the user.
Authorized User	A user who has been provided Authorization factors by the administrator to use the TOE.
Data Encryption	The process of encrypting all user data written to volatile memory.
Deterministic Random Bit Generator (DRBG)	A cryptographic algorithm that produces a sequence of bits from a secret initial seed value. Without knowledge of the seed value, the output sequence should be unpredictable up to the security level of the DRBG.
Entropy Source	This cryptographic function provides a seed for a random bit generator by accumulating the outputs from one or more noise sources. The functionality includes a measure of the minimum work required to guess a given output and tests to ensure that the noise sources are operating properly.
File/Set of files	The user data that is selected to be encrypted, which can include individual file encryption (with a FEK per file) or a set of files encrypted with a single FEK.
File Authentication Key (FAK)	The secret value used as input when a keyed hash function is used to perform data authentication.
File Encryption Key (FEK)	The key that is used by the encryption algorithm to encrypt the selected user data on the host machine.
Key Chaining	The method of using multiple layers of encryption keys to protect data. A top layer key encrypts a lower layer key which encrypts the data; this method can have any number of layers.
Key Encryption Key (KEK)	The key that is used to encrypt another key.
Keying Material	The KEK, FEK, authorization factors and random numbers or any other values from which keys are derived.
Key Sanitization	A method of sanitizing encrypted data by securely overwriting the key, as described in the key destruction requirement, that was encrypting the data.
Noise Source	The component of an RBG that contains the non-deterministic, entropy-producing activity.
Operational Environment	Hardware and software that are outside the TOE boundary that support the TOE functionality and security policy, including the platform, its firmware, and the operating system.
Password	A short string of characters used for authorization to the data on the device.
Passphrase	A string of words that may be used for authorization to the data on the device.
Primary Key Chain	The direct key chain from the authorization factor to the FEK.

Random Bit Generator (RBG)	A cryptographic function composed of an entropy source and DRBG that is invoked for random bits needed to produce keying material.
Sensitive Data	Any data of which the compromise with respect to loss, misuse, or unauthorized access to or modification of could adversely affect the interest of the TOE user.
Shutdown	Power down or unintentional loss of power of the TOE or platform.
Supplemental Key Chain	Other key chains that add protection or functionality without compromising the security of the primary key chain.
System files	Files that reside on the host machine that are used in the operation of the file encryption software.
Temporary File	A file created by an application for short term storage of sensitive data.
Trusted Host	Source/destination host configured and maintained to provide the TOE with appropriate IT security commensurate with the value of the user data protected by the TOE.
Unauthorized User	A user who has not been authorized to use the TOE and decrypt encrypted user data.
User Data	All data that originate on the host, or is derived from data that originate on the host, excluding system files and signed firmware updates from the TOE manufacturer.
Volatile memory	Memory that loses its content when power is turned off.
Zeroize	This term is used to make a distinction between dereferencing a memory location and actively overwriting it with a constant. Keying material needs to be overwritten when it is no longer needed.

2 Evaluation Activities for SFRs

The EAs presented in this section capture the actions the evaluator performs to address technology specific aspects covering specific SARs (e.g. ASE_TSS.1, ADV_FSP.1, AGD_OPE.1, and ATE_IND.1) – this is in addition to the CEM work units that are performed in 6 Evaluation Activities for SARs.

Regarding design descriptions (designated by the subsections labelled TSS, as well as any required supplementary material that may be treated as proprietary), the evaluator must ensure there is specific information that satisfies the EA. For findings regarding the TSS section, the evaluator's verdicts will be associated with the CEM work unit ASE_TSS.1-1. Evaluator verdicts associated with the supplementary evidence will also be associated with ASE_TSS.1-1, since the requirement to provide such evidence is specified in ASE in the cPP.

For ensuring the guidance documentation provides sufficient information for the administrators/users as it pertains to SFRs, the evaluator's verdicts will be associated with CEM work units ADV_FSP.1-7, AGD_OPE.1-4, and AGD_OPE.1-5.

Finally, the subsection labelled Tests is where the authors have determined that testing of the product in the context of the associated SFR is necessary. While the evaluator is expected to develop tests, there may be instances where it is more practical for the developer to construct tests, or where the developer may have existing tests. Therefore, it is acceptable for the evaluator to witness developer-generated tests in lieu of executing the tests. In this case, the evaluator must ensure the developer's tests are executing both in the manner declared by the developer and as mandated by the EA. The CEM work units that are associated with the EAs specified in this section are: ATE_IND.1-3, ATE_IND.1-4, ATE_IND.1-5, ATE_IND.1-6, and ATE_IND.1-7.

2.1 Application Software Protection Profile

The EAs defined in this section are only applicable in cases where the TOE claims conformance to a PP-Configuration that includes the App PP.

2.1.1 Modified SFRs

The PP-Module does not modify any requirements when the App PP is the base.

2.1.2 TOE SFR Evaluation Activities

2.1.3 Cryptographic Support (FCS)

FCS_CKM_EXT.2 File Encryption Key (FEK) Generation

TSS
FCS_CKM_EXT.2.1: The evaluator shall review the TSS to determine that a description covering how and when the FEKs are generated exists. The description must cover all environments on which the TOE is claiming conformance, and include any preconditions that must exist in order to successfully generate the FEKs. The evaluator shall verify that the description of how the FEKs are generated is consistent with the instructions in the AGD guidance, and any differences that arise from different platforms are taken into account.

Conditional:
If 'using a Random Bit Generator' was selected, the evaluator shall verify that the TSS describes how the functionality described by FCS_RBG_EXT.1 (from the [AppPP]) is invoked to generate FEK. To the extent possible from the description of the RBG functionality in FCS_RBG_EXT.1 (from [AppPP]), the evaluator shall determine that the key size being requested is identical to the key size and mode to be used for the decryption/encryption of the user data (FCS_COP.1(1)) (from [AppPP]).

Conditional:
If 'derived from a password/passphrase' is selected, the examination of the TSS section is performed as part of FCS_CKM_EXT.6 evaluation activities.

FCS_CKM_EXT.2.2: The evaluator shall verify the TSS describes how a FEK is used for a protected resource and associated with that resource. The evaluator confirms that-per this description-the FEK is unique per resource (file or set of files) and that the FEK is established using the mechanisms specified in FCS_CKM_EXT.2.1).

Guidance
The evaluator shall review the instructions in the AGD guidance to determine that any explicit actions that need to be taken by the user to establish a FEK exist-taking into account any differences that arise from different platforms-and are consistent with the description in the TSS.

Tests
None.

FCS_CKM_EXT.4 Cryptographic Key Destruction

TSS
The evaluator shall verify the TSS provides a high level description of what it means for keys and key material to be no longer needed and when they should be expected to be destroyed. The evaluator shall verify the TSS provides a high level description of what it means for keys and key material to be no longer needed and when then should be expected to be destroyed.

KMD

The evaluator examines the KMD to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.

The evaluator shall check to ensure the KMD lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).

The evaluator examines the interface description for each different media type to ensure that the interface supports the selection(s) and description in the KMD.

If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the KMD to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.

The evaluator shall check that the KMD identifies any configurations or circumstances that may not strictly conform to the key destruction requirement.

If the selection "destruction of all KEKs protecting target key, where none of the KEKs protecting the target key are derived" is included the evaluator shall examine the TOE's keychain in the KMD and identify each instance when a key is destroyed by this method. In each instance the evaluator shall verify all keys capable of decrypting the target key are destroyed in accordance with a specified key destruction method in FCS_CKM_EXT.4.1. The evaluator shall verify that all of the keys capable of decrypting the target key are not able to be derived to reestablish the keychain after their destruction.

The evaluator shall verify the KMD includes a description of the areas where keys and key material reside and

when the keys and key material are no longer needed.

The evaluator shall verify the KMD includes a key lifecycle, that includes a description where key material reside, how the key material is used, how it is determined that keys and key material are no longer needed, and how the material is destroyed once it is not needed and that the documentation in the KMD follows FCS_CKM_EXT.4.1 for the destruction.

Guidance
There are a variety of concerns that may prevent or delay key destruction in some cases.

The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS and any other relevant Required Supplementary Information.

The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer and how such situations can be avoided or mitigated if possible.

Some examples of what is expected to be in the documentation are provided here.

When the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-leveling and garbage collection. This may create additional copies of the key that are logically inaccessible but persist physically. In this case, to mitigate this the drive should support the TRIM command and implements garbage collection to destroy these persistent copies when not actively engaged in other tasks.

Drive vendors implement garbage collection in a variety of different ways, as such there is a variable amount of time until data is truly removed from these solutions. There is a risk that data may persist for a longer amount of time if it is contained in a block with other data not ready for erasure. To reduce this risk, the operating system and file system of the OE should support TRIM, instructing the non-volatile memory to erase copies via garbage collection upon their deletion. If a RAID array is being used, only set-ups that support TRIM are utilized. If the drive is connected via PCI-Express, the operating system supports TRIM over that channel.

The drive should be healthy and contains minimal corrupted data and should be end of lifed before a significant amount of damage to drive health occurs, this minimizes the risk that small amounts of potentially recoverable data may remain in damaged areas of the drive.

Tests
These tests are only for key destruction provided by the application, test 2 does not apply to any keys using the selection "new value of a key":

- **Test 1:** Applied to each key held in volatile memory and subject to destruction by overwrite by the TOE (whether or not the value is subsequently encrypted for storage in volatile or non-volatile memory). In the case where the only selection made for the key destruction method was removal of power, then this test is unnecessary.

  The evaluator shall:

  1. Record the value of the key in the TOE subject to clearing.
  2. Cause the cause the TOE or the underlying platform to dump to perform a normal cryptographic processing with the key from Step #1.
  3. Cause the TOE to clear the key.
  4. Cause the TOE to stop the execution but not exit.
  5. Cause the TOE to dump the entire memory of the TOE into a binary file.
  6. Search the content of the binary file created in Step #5 for instances of the known key value from Step #1.
  Steps #1-6 ensure that the complete key does not exist anywhere in volatile memory. If a copy is found, then the test fails.

- **Test 2:** [Conditional] If new value of a key is selected this test does not apply.
  Applied to each key held in non-volatile memory and subject to destruction by the TOE.
  The evaluator shall use special tools (as needed), provided by the TOE developer if necessary, to ensure the tests function as intended.
  1. Identify the purpose of the key and what access should fail when it is deleted. (e.g. the file encryption key being deleted would cause data decryption to fail.)
  2. Cause the TOE to clear the key.
  3. Have the TOE attempt the functionality that the cleared key would be necessary for.
  4. The test succeeds if Step #3 fails.

  Tests 3 and 4 do not apply for the selection instructing the underlying platform to destroy the representation of the key, as the TOE has no visibility into the inner workings and completely relies on the underlying platform.

- **Test 3:** Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use a tool that provides a logical view of the media (e.g., MBR file system):
  1. Record the value of the key in the TOE subject to clearing.
  2. Cause the TOE to perform a normal cryptographic processing with the key from Step #1.
  3. Cause the TOE to clear the key.
  4. Search the logical view that the key was stored in for instances of the known key value from Step #1.
  If a copy is found, then the test fails.

- **Test 4:** Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use a tool that provides a logical view of the media:
  1. Record the logical storage location of the key in the TOE subject to clearing.
  2. Cause the TOE to perform a normal cryptographic processing with the key from Step #1.
  3. Cause the TOE to clear the key.
  4. Read the logical storage location in Step #1 of non-volatile memory to ensure the appropriate pattern is utilized.
  The test succeeds if correct pattern is used to overwrite the key in the memory location. If the pattern is not found the test fails.

FCS_IV_EXT.1 Initialization Vector Generation

TSS
The evaluator shall ensure the TSS describes how nonces are created uniquely and how IVs and tweaks are handled (based on the AES mode). The evaluator shall confirm that the nonces are unique and the IVs and tweaks meet the stated requirements.

Guidance
None.

Tests
None.

FCS_KYC_EXT.1 Key Chaining and Key Storage

TSS
The evaluator shall verify the TSS contains a high level description of all keychains and authorization methods selected in FIA_AUT_EXT.1 that are used to protect the KEK or FEK.

KMD

The evaluator shall examine the KMD to ensure it describes each key chain in detail, and these descriptions correspond with the selections of the requirement. The description of each key chain shall be reviewed to ensure the options for maintaining the key chain are documented.

The evaluator shall verify the KMD to ensure that it describes how each key chain process functions, such that it does not expose any material that might compromise any key in the chain. A high-level description should include a diagram illustrating the keychain(s) implemented and detail where all keys and keying material is stored or how the keys or key material are derived. The evaluator shall examine the primary key chain to ensure that at no point the chain could be broken without a cryptographic exhaust or knowledge of the KEK or FEK and the effective strength of the FEK is maintained throughout the Key Chain as specified in the requirement.

Guidance
None.

Tests
None.

FCS_VAL_EXT.1 Validation

TSS
Conditional:
If 'validating' is selected in FCS_VAL_EXT.1.1, the evaluator shall examine the TSS to determine which authorization factors support validation.

The evaluator shall examine the TSS to ensure that it contains a high-level description of how how the submasks are validated. If multiple submasks are used within the TOE, the evaluator shall verify that the TSS describes how each is validated (e.g., each submask validated before combining, once combined validation takes place).

Conditional:
If 'receiving assertion' is selected in FCS_VAL_EXT.1.1, the evaluator shall examine the TSS to verify that it

describes the enviroments that can be leveraged with the TOE and how each claims to perform validation. The evaluator shall ensure that none of the stated platform validation mechanisms weaken the key chain of the product.

Guidance
If the validation functionality is configurable, the evaluator shall examine the operational guidance to ensure it describes how to configure the TOE to ensure the limits regarding validation attempts can be established.

Tests
There are no test activities for this requirement.

2.1.4 User Data Protection (FDP)

FDP_PRT_EXT.1 Protection of Selected User Data

TSS
The evaluator shall examine the TSS to determine that it lists each type of resource that can be encrypted (e.g., file, directory) and what "encrypted" means in terms of the resource (e.g., "encrypting a directory" means that all of the files contained in the directory are encrypted, but the data in the directory itself (which are filenames and pointers to the files) are not encrypted).

The evaluator shall also confirm that the TSS describes how each type of resource listed is encrypted and decrypted by the TOE. The evaluator shall ensure that this description includes the case where an existing file or set of files is encrypted for the first time; a new file or set of files is created and encrypted; an existing file or set of files is re-encrypted (that is, it had been initially encrypted; it was decrypted (by the TOE) for use by the user, and is then subsequently re-encrypted); and corresponding decryption scenarios. If other scenarios exist due to product implementation/features, the evaluator shall ensure that those scenarios are covered in the TSS as well.

The evaluator shall examine the TSS to ensure there is a high-level description of how the FEK is protected.

The evaluator shall examine the TSS to ensure there is a description of how the FEK is protected.

The evaluator shall examine the TSS to ensure that it describes all temporary files/resources created or memory used during the decryption/encryption process and when those files/resources or memory is no longer needed.

The TSS shall describe how the TSF or TOE platform deletes the non-volatile memory (for example, files) and volatile memory locations after the TSF is done with its decryption/encryption operation.

Guidance
If the TOE creates temporary objects and these objects can be protected through administrative measures (e.g., the TOE creates temporary files in a designated directory that can be protected through configuration of its access control permissions), then the evaluator shall check the Operational Guidance to ensure that these measures are described.

If there are special measures necessary to configure the method by which the file or set of files are encrypted (e.g., choice of algorithm used, key size, etc.), then those instructions shall be included in the Operational Guidance and verified by the evaluator. In these cases, the evaluator checks to ensure that all non-TOE products used to satisfy the requirements of the ST that are described in the Operational Guidance are consistent with those listed in the ST, and those tested by the evaluation activities of this PP-Module.

There are no additional guidance evaluation activities for FDP_PRT_EXT.1.2.

Tests
The evaluator shall also perform the following tests. All instructions for configuring the TOE and each of the environments must be included in the Operational Guidance and used to establish the test configuration.

For each resource and decryption/encryption scenario listed in the TSS, the evaluator shall ensure that the TSF is able to successfully encrypt and decrypt the resource using the following methodology:
Monitor the temporary resources being created (if any) and deleted by the TSF-the tools used to perform the monitoring (e.g., procmon for a Windows system) shall be identified in the test report. The evaluator shall ensure that these resources are consistent with those identified in the TSS, and that they are protected as specified in the Operational Guidance and are deleted when the decryption/encryption operation is completed.

- **Test 1:** This test only applies for application provided functionality.

  1. Using a file editor, create and save a text file that is encrypted per the evaluation configured encryption policy. The contents of the file will be limited to a known text pattern to ensure that the text pattern will be present in all encryption/decryption operations performed by the TOE.

2. Exit the file editor so that the file (including its known text pattern) has "completed the decryption/encryption operation" and process memory containing the known text pattern is released.

3. The evaluator will take a dump of volatile memory and search the retrieved dump for the known pattern. The test fails if the known plaintext pattern is found in the memory dump.

4. The evaluator will search the underlying non-volatile storage for the known pattern. The test fails if the known plaintext pattern is found in the search.

FDP_PRT_EXT.2 Destruction of Plaintext Data

TSS
The evaluator shall examine the TSS to ensure that it describes all temporary file (or set of files) that are created in the filesystem of the host during the decryption/encryption process, and that the TSS describes how these files are deleted after the TSF is done with its decryption/encryption operation. Note that if other objects/resources are created on the host that are 1) persistent and 2) visible to other processes (users) on that host that are not filesystem objects, those objects shall be identified and described in the TSS as well.

Guidance
None.

Tests

- **Test 1:** If the TSS creates temporary files/resources during file decryption/encryption, the evaluator shall perform the following tests to verify that the temporary files/resources are destroyed. If the product supported shared files per FIA_FCT_EXT.2, this test must be repeated with a shared file. The evaluator shall use a tool (e.g., procmon for a Windows system) that is capable of monitoring the creation and deletion of files during the decryption/encryption process is performed. A tool that can search the contents of the hard drive (e.g., winhex) will also be needed. The tools used to perform the monitoring shall be identified in the test report.
  (Creating an encrypted document)
  - Open an editing application.
  - Create a special string inside the document. The string could be 5-10 words. It is recommended to remove the spaces. This will create a one page document.
  - Start the file monitoring tool.
  - Save and close the file.
  - Encrypt the file using the TOE (if the TOE does not encrypt automatically for the user).

  Analysis Steps
  - If needed, exit/close the TOE.
  - Stop the file monitoring tool. View the results. Identify any temporary files that were created during the encryption process. Examine to see if the temporary files were destroyed when the TOE closed.
  - If temporary files remain, these temporary files should be examined to ensure that no plaintext data remains. If plaintext data is found in these files, the test fails.
  - Search the contents of the hard drive (using the second tool) for the plaintext string used above. (The search should be performed using both ASCII and Unicode formats.)
  - If the string is found, this means that plaintext from the test fails.

  (Creating, encrypting a blank document and then adding text):
  - Encrypt a blank document using the tool.
  - Create a special string inside the document. The string could be 5-10 words. It is recommended to remove the spaces. This will create a one page document.
  - Start the file monitoring tool.
  - Save and close the file.
  - Perform the "Analysis Steps" listed above.

2.1.5 Identification and Authentication (FIA)

FIA_AUT_EXT.1 Subject Authorization

TSS
The evaluator shall examine the TSS to ensure that it describes how user authentication is performed. The evaluator shall verify that the authorization methods listed in the TSS are specified and included in the requirements in the ST.

Requirement met by the TOE
The evaluator shall first examine the TSS to ensure that the authorization factors specified in the ST are described. For password-based factors the examination of the TSS section is performed as part of FCS_CKM_EXT.6 Evaluation Activities. Additionally in this case, the evaluator shall verify that the operational guidance discusses the characteristics of external authorization factors (e.g., how the authorization factor must be generated; format(s) or standards that the authorization factor must meet) that are able to be used by the TOE. If other authorization factors are specified, then for each factor, the TSS specifies how the factors

are input into the TOE.

Requirement met by the platform
The evaluator shall examine the TSS to ensure a description is included for how the TOE is invoking the platform functionality and how it is getting an authorization value that has appropriate entropy.

Guidance
The evaluator shall verify that the AGD guidance includes instructions for all of the authorization factors. The AGD will discuss the characteristics of external authorization factors (e.g., how the authorization factor is generated; format(s) or standards that the authorization factor must meet, configuration of the TPM device used) that are able to be used by the TOE.

Tests
The evaluator shall ensure that authorization using each selected method is tested during the course of the evaluation, setting up the method as described in the operational guidance and ensuring that authorization is successful and that failure to provide an authorization factor results in denial to access to plaintext data.

[conditional]: If there is more than one authorization factor, ensure that failure to supply a required authorization factor does not result in access to the decrypted plaintext data.

2.1.6 Security Management (FMT)

FMT_SMF.1(2) Specification of File Encryption Management Functions

TSS
Conditional Activities: The evaluator shall examine the TSS to ensure that it describes the sequence of activities that take place from an implementation perspective when this activity is performed (for example, how it determines which resources are associated with the KEK, the decryption and re-encryption process), and ensure that the KEK and FEK are not exposed during this change.

Cryptographic Configuration: None for this requirement.

Guidance
Conditional Activities: The evaluator shall examine the Operational Guidance to ensure that it describes how the password/passphrase-based authorization factor is to be changed.

Cryptographic Configuration: The evaluator shall determine from the TSS for other requirements (FCS_*, FDP_PRT_EXT, FIA_AUT_EXT) what portions of the cryptographic functionality are configurable. The evaluator shall then review the AGD documentation to determine that there are instructions for manipulating all of the claimed mechanisms.

Tests
Cryptographic Erase: If the TOE uses stored FEKS or KEKs, the evaluator shall examine the key chain to determine that the keys destroyed by a cryptographic erase will result in the data becoming unrecoverable. Testing for this activity is performed for other components in this PP-Module.

2.1.7 Protection of the TSF (FPT)

FPT_KYP_EXT.1 Protection of Keys and Key Material

TSS
The evaluator shall verify the TSS for a high level description of the method(s) used to protect keys stored in non-volatile memory.

KMD

The evaluator shall verify the KMD to ensure it describes the storage location of all keys and the protection of all keys stored in non-volatile memory. The description of the key chain shall be reviewed to ensure FCS_COP.1(5) is followed for the storage of wrapped or encrypted keys in non-volatile memory and plaintext keys in non-volatile memory meet one of the criteria for storage.

Guidance
None.

Tests
None.

3 Evaluation Activities for Optional SFRs

3.1 Cryptographic Support (FCS)

FCS_CKM_EXT.5 File Authentication Key (FAK) Support

TSS
FCS_CKM_EXT.5.1: The evaluator shall examine the TSS to determine how the FAK is stored (or not stored) in memory.

FCS_CKM_EXT.5.2: The evaluator shall examine the TSS to determine that it describes how a FAK is created for a protected resource and associated with that resource; protection of the FAK itself is covered by FCS_COP_EXT.1. The evaluator confirms that-per this description-the FAK is unique per resource (file or set of files) and that the FAK is created using a DRBG.

FCS_CKM_EXT.5.3: The TSS must detail that the FAKs are generated on the client machine and are not generated on an external server.

FCS_CKM_EXT.5.4: FCS_CKM_EXT.4 contains the requirements necessary to ensure that plaintext keys and key material do not remain in plaintext form in the TSF's non-volatile memory space. In TOEs where the FAK is protected with a KEK, the FAK will need to be encrypted and stored in non-volatile memory when not being used to decrypt/encrypt a file. (Typically, the encrypted FAK is stored in the meta-data of the encrypted file(s).) The evaluator shall examine the TSS to ensure that it describes how the FAK is encrypted, both after its initial creation and after it has been decrypted for use (note that in the entirely likely possibility that the FAK is not re-encrypted, then this case must be indicated in the TSS and the description for FCS_CKM_EXT.4 will cover disposal of the plaintext FEK and FAK). The evaluator shall further check to ensure that the TSS describes how the FAK and any other associated meta-data necessary to decrypt the file or set of files are associated with the resource. This description can be combined with the description required for FCS_COP_EXT.1.

Guidance
None.

Tests
An example ciphertext file generated via the TOE shall be provided to the evaluator with the accompanying FAK and prerequisite authorization information used for encryption. The evaluator will use the TOE in conjunction with a debugging or forensics utility to attempt an authentication of the ciphertext file using the provided authorization information. The evaluator will then terminate processing of the TOE and perform a search through non-volatile memory using the provided FAK string. The evaluator must document each command, program or action taken during this process, and must confirm that the FAK was never written to non-volatile memory. This test must be performed three times to ensure repeatability. If during the course of this testing the evaluator finds that the FAK was written to non-volatile memory, they should be able to identify the cause (i.e. the TOE wrote the FAK to disk, the TOE platform dumped volatile memory as a page file, etc.), and document the reason for failure to comply with the requirement.

FCS_COP_EXT.1 FAK Encryption/Decryption Support

TSS
The evaluator shall follow the evaluation activities as laid out in FCS_COP.1(5) to assert proper FAK protection.

Guidance
None.

Tests
None.

3.2 User Data Protection (FDP)

FDP_AUT_EXT.1 Authentication of Selected User Data

TSS
The evaluator shall examine the TSS to determine that it lists each type of resource that can be authenticated (e.g., file, directory) and what "authenticated" means in terms of the resource (e.g., "authenticating a directory" means that all of the files contained in the directory are authenticated, but the data in the directory itself (which are filenames and pointers to the files) are not authenticated).

The evaluator shall also confirm that the TSS describes how each type of resource listed is authenticated by the TOE and how authentication measures are added to each resource (e.g. taking all the encrypted files through a MAC function and appending the MAC to the set of files). The evaluator shall ensure that this description includes the case where an existing file or set of files has authentication measures added for the first time; a new file or set of files is created and adds authentication measure; an existing file or set of files updates or replaces its existing authentication measures (that is, it had a MAC appended to the data; it was authenticated and decrypted (by the TOE) for use by the user, and is then subsequently re-encrypted with an updated MAC); and corresponding decryption scenarios. If other scenarios exist due to product implementation/features, the evaluator shall ensure that those scenarios are covered in the TSS as well.

Guidance
If the TOE creates temporary objects and these objects can be protected through administrative measures (e.g., the TOE creates temporary files in a designated directory that can be protected through configuration of its access control permissions), then the evaluator shall check the Operational Guidance to ensure that these measures are described.

If there are special measures necessary to configure the method by which the file or set of files are authenticated (e.g., choice of function used, additional keys, etc.), then those instructions shall be included in the Operational Guidance and verified by the evaluator. This includes, for instance, lists of allowed platforms, libraries, and devices, and instructions for using them. In these cases, the evaluator checks to ensure that all non-TOE products used to satisfy the requirements of the ST that are described in the Operational Guidance are consistent with those listed in the ST, and those tested by the evaluation activities of this PP-Module.

Tests
The evaluator shall also perform the following tests. These tests must be performed for each data authentication feature and platform claimed in the ST; all instructions for configuring the TOE and each of the environments must be included in the Operational Guidance and used to establish the test configuration.

For each resource and data authentication scenario listed in the TSS, the evaluator shall ensure that the TSF is able to successfully add authentication measures and authenticate the resource using the following methodology.

Monitor the temporary resources being created (if any) and deleted by the TSF-the tools used to perform the monitoring (e.g., procmon for a Windows system) shall be identified in the test report. The evaluator shall ensure that these resources are consistent with those identified in the TSS, and that they are protected as specified in the Operational Guidance and are deleted when the decryption/encryption and authentication operations are completed.

FDP_AUT_EXT.2 Data Authentication Using cryptographic Keyed-Hash Functions

TSS
The evaluator shall check the TSS section to confirm that it describes how a request for each type of supported resource (file (or set of files)) will result in data authentication using a keyed hash function. The evaluator will confirm that the TOE will respond appropriately to a failed authentication, to include notifying the user of an invalid authentication and preventing decryption. The evaluator will confirm that any file encryption utility will be able to identify where the MAC is placed.

The evaluator will confirm that a FAK is used as part of the authentication process and will identify the keyed hash function utilized.

Guidance
It is encouraged for every implementation to use a FAK that is wholly different and independently generated from the FEK.

Tests
The evaluator shall perform the following test:

- **Test 1:** Create an encrypted file and confirm that authentication of this file using the correct FAK will result in a successful decryption.
- **Test 2:** Modify an arbitrary number of bits of ciphertext and attempt to run the authentication and decryption operations on the file. Assert that the TOE successfully identified the forged ciphertext file and notified the user.

FDP_AUT_EXT.3 Data Authentication Using Asymmetric Signing and Verification

TSS
The evaluator shall check the TSS section to confirm that it describes how a request for each type of supported resource (file (or set of files)) will result in data authentication using a secure hash and cryptographic signing process. The evaluator will confirm that the supplied public and private key pair were generated in accordance with FCS_CKM.1(1). The evaluator will confirm that the entire ciphertext file was used to create the hash and that the hash was used as input to the cryptographic signing function. The evaluator will confirm that the TSF notifies the user of an unsuccessful authentication and prevents decryption. The evaluator shall confirm that the signature is appended to the end of the ciphertext file.

Guidance
None.

Tests
The evaluator shall perform the following test:

- **Test 1:** Create an encrypted file and demonstrate that authentication of this file using the correct keying material will be successful.
- **Test 2:** Modify an arbitrary number of bits of ciphertext and attempt to run the authentication and decryption operations on the file. Assert that the TOE successfully identified the forged ciphertext file

and notified the user.

FDP_PM_EXT.1 Protection of Data in Power Managed States

TSS
The evaluator shall examine the TSS to ensure that it describes the state(s) that are supported by this capability. For each state, the evaluator ensures that the TSS contains a description of how the state is entered, and the actions of the TSF on entering the state, specifically addressing how multiple open resources (of each type) are protected, and how keying material associated with these resources is protected (if different from that described elsewhere). The TSF shall also describe how the state is exited, and how the requirements are met during this transition to an operational state.

The evaluator shall verify the TSS provides a description of what keys and key material are destroyed when entering any protected state.

KMD

The evaluator shall verify the KMD includes a description of the areas where keys and key material reside. The evaluator shall verify the KMD includes a key lifecycle that includes a description where key material reside, how the key material is used, and how the material is destroyed once a claimed power state is entered and that the documentation in the KMD follows FCS_CKM_EXT.4.1 for the destruction.

Guidance
The evaluator shall check the Operational Guidance to determine that it describes the states that are supported by the TOE, and provides information related to the correct configuration of these modes and the TOE.

The evaluator shall validate that guidance documentation contains clear warnings and information on conditions in which the TOE may end up in a non-protected state. In that case it must contain mitigation instructions on what to do in such scenarios.

Tests
The following tests must be performed by the evaluator for each supported State, type of resource, platform, and authorization factor:

- **Test 1:** Following the Operational guidance, configure the Operational Environment and the TOE so that the lower power state of the platform is enabled and protected by the TOE. Open several resources (documented in the test report) that are protected. Invoke the lower power state. On resumption of normal power attempt to access a previously-opened protected resource, observe that an incorrect entry of the authorization factor(s) does not result in access to the system, and that correct entry of the authorization factor(s) does result in access to the resources.

FDP_PRT_EXT.3 Protection of Third-Party Data

TSS
The evaluator shall examine the TSS to ensure that it describes how the TOE detects and encrypts temporary files (or set of files) that are created in the filesystem of the host by third party products.

Guidance
[conditional] If any configuration is required for this process the evaluator shall verify it is described in the guidance documentation.

Tests
The evaluator shall utilize any third party application that would be protected under this protection to generate files, then verify those files are being encrypted.

3.3 Identification and Authentication (FIA)

FIA_FCT_EXT.1 Multi-User Authorization

TSS
The evaluator shall examine the TSS to determine that it identifies each of the resource protected in encrypted form and the key chain that protects that resource.

The evaluator shall examine the TSS to verify key chains are separate ensuring resources are protected from other users, with the exception of files permitted to be shared under the mechanism described in FIA_FCT_EXT.2.

Guidance
- The evaluator shall examine the operation guidance to determine that it contains instructions on how to establish multiple accounts and protect resources from other users. If different for different underlying

platforms, the evaluator determines that all platforms listed in the ST are addressed.

Tests
The evaluator shall ensure that different users using different authorization factors are unable to decrypt each others protected resources for each type of protected resource identified in the TSS. The test succeeds if the users are unable to decrypt resources not chained to them, with the exception of any resources linked in FIA_FCT_EXT.2.

FIA_FCT_EXT.2 Authorized Key Sharing

TSS
The evaluator shall examine the TSS to determine that it identifies each of the resources that is sharable in encrypted form (for instance, encrypted files may be sharable among users, but encrypted directories may not), and the method by which the resource can be shared among users with different authorization factors.

The evaluator shall examine the operation guidance to determine that it contains instructions on how to set up and share resources with other users, if additional actions are necessary due to use of the encryption product. If different for different underlying platforms, the evaluator determines that all platforms listed in the ST are addressed.

Guidance
- The evaluator shall examine the operation guidance to determine that it contains instructions on how to set up and share resources with other users, if additional actions are necessary due to use of the encryption product. If different for different underlying platforms, the evaluator determines that all platforms listed in the ST are addressed.

Tests

- **Test 1:** For each type of resource that is identified in the TSS as sharable in its encrypted form, the evaluator shall ensure that different users using different authorization factors are able to successfully access the resource using different authorization factors. This should include making changes to the resource to ensure that the same resource is being shared, and that a per-user copy of the resource is not being made.
- **Test 2:** For each type of resource that is identified in the TSS as sharable in its encrypted form, the evaluator shall ensure that a different unauthorized user is unable to access the encrypted file shared. This test succeeds if the unauthorized user is unable to access a file shared between other authorized users.

4 Evaluation Activities for Selection-Based SFRs

4.1 Cryptographic Support (FCS)

FCS_CKM_EXT.3 Key Encrypting Key (KEK) Support

TSS
The evaluator shall review the TSS to determine that a description covering how and when KEK(s) are generated exists. The description must cover all environments on which the TOE is claiming conformance, and include any preconditions that must exist in order to successfully generate the KEKs. The evaluator shall verify that the description of how the KEK(s) are generated is consistent with the instructions in the AGD guidance, and any differences that arise from different platforms are taken into account.

Conditional:
If using a RBG was selected the evaluator shall examine the TSS and verify that it describes how the functionality described by FCS_RBG_EXT.1 (from the [AppPP]) is invoked to generate KEK(s). To the extent possible from the description of the RBG functionality in FCS_RBG_EXT.1 (from [AppPP]), the evaluator shall determine that the key size being requested is identical to the key size selected.

Conditional:
If derived from a password/passphrase is selected the examination of the TSS section is performed as part of FCS_CKM_EXT.6 evaluation activities.

Guidance
The evaluator shall review the instructions in the AGD guidance to determine that any explicit actions that need to be taken by the user to establish a KEK exist-taking into account any differences that arise from different platforms-and are consistent with the description in the TSS.

Tests
None.

FCS_CKM_EXT.6 Cryptographic Password/Passphrase Conditioning

TSS

FCS_CKM_EXT.6.1: There are two aspects of this component that require evaluation: passwords/passphrases of the length specified in the requirement (at least 64 characters or a length defined by the platform) are supported, and that the characters that are input are subject to the selected conditioning function. These activities are separately addressed in the text below.

Support for minimum length: The evaluators shall check to ensure that the TSS describes the allowable ranges for password/passphrase lengths, and that at least 64 characters or a length defined by the platform may be specified by the user.

Support for character set: The evaluator shall check to ensure that the TSS describes the allowable character set and that it contains the characters listed in the SFR.

Support for PBKDF: The evaluator shall examine the TSS to ensure that the formation of all KEKs or FEKs (as decided in the FCS_CKM_EXT.3 selection) is described and that the key sizes match that described by the ST author.

The evaluator shall check that the TSS describes the method by which the password/passphrase is first encoded and then fed to the SHA algorithm. The settings for the algorithm (padding, blocking, etc.) shall be described, and the evaluator shall verify that these are supported by the selections in this component as well as the selections concerning the hash function itself. The evaluator shall verify that the TSS contains a description of how the output of the hash function is used to form the submask that will be input into the function and is the same length as the KEK as specified in FCS_KYC_EXT.1.

For the NIST SP 800-132-based conditioning of the password/passphrase, the required evaluation activities will be performed when doing the evaluation activities for the appropriate requirements (FCS_COP.1.1(4)). If any manipulation of the key is performed in forming the submask that will be used to form the FEK or KEK, that process shall be described in the TSS.

No explicit testing of the formation of the submask from the input password is required.

FCS_CKM_EXT.6.2: The ST author shall provide a description in the TSS regarding the salt generation. The evaluator shall confirm that the salt is generated using an RBG described in FCS_RBG_EXT.1 (from the [AppPP]).

Guidance
Support for minimum length: The evaluators shall check the Operational Guidance to determine that there are instructions on how to generate large passwords/passphrases, and instructions on how to configure the password/passphrase length to provide entropy commensurate with the keys that the authorization factor is protecting.

Tests
Support for Password/Passphrase characteristics: In addition to the analysis above, the evaluator shall also perform the following tests on a TOE configured according to the Operational Guidance:

- **Test 1:** Ensure that the TOE supports password/passphrase lengths as defined in the SFR assignments.
- **Test 2:** Ensure that the TOE does not accept more than the maximum number of characters specified in FCS_CKM_EXT.6.1.
- **Test 3:** Ensure that the TOE does not accept less than the minimum number of characters specified in FCS_CKM_EXT.6.4. If the minimum length is settable by the administrator, the evaluator determines the minimum length or lengths to test.
- **Test 4:** Ensure that the TOE supports passwords consisting of all characters listed in FCS_CKM_EXT.6.2.

Conditioning: No explicit testing of the formation of the authorization factor from the input password/passphrase is required.

FCS_COP.1(5) Cryptographic operation (Key Wrapping)

TSS
Conditional: If use platform provided functionality was selected, then the evaluator shall examine the TSS to verify that it describes how the FEK encryption/decryption is invoked.

Conditional: If implement functionality was selected, The evaluator shall check that the TSS includes a description of encryption function(s) used for key wrapping. The evaluator should check that this description of the selected encryption function includes the key sizes and modes of operations as specified in the selection above. The evaluator shall check that the TSS describes the means by which the TOE satisfies constraints on algorithm parameters included in the selections made for 'cryptographic algorithm' and 'list of standards'.

The evaluator shall verify the TSS includes a description of the key wrap function(s) and shall verify the key wrap uses an approved key wrap algorithm according to the appropriate specification.

KMD

The evaluator shall review the KMD to ensure that all keys are wrapped using the approved method and a description of when the key wrapping occurs.

Guidance
If multiple encryption modes are supported, the evaluator examines the guidance documentation to determine that the method of choosing a specific mode/key size by the end user is described.

Tests
The evaluation activity tests specified for AES in GCM mode in the underlying [AppPP] shall be performed in the case that "GCM" is selected in the requirement.

AES Key Wrap (AES-KW) and Key Wrap with Padding (AES-KWP) Test

The evaluator will test the authenticated encryption functionality of AES-KW for EACH combination of the following input parameter lengths:

- 128 and 256 bit key encryption keys (KEKs)
- Three plaintext lengths. One of the plaintext lengths shall be two semi-blocks (128 bits). One of the plaintext lengths shall be three semi-blocks (192 bits). The third data unit length shall be the longest supported plaintext length less than or equal to 64 semi-blocks (4096 bits).

using a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KW authenticated encryption. To determine correctness, the evaluator will use the AES-KW authenticated-encryption function of a known good implementation.

The evaluator will test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption.

The evaluator will test the authenticated-encryption functionality of AES-KWP using the same test as for AES-KW authenticated-encryption with the following change in the three plaintext lengths:

- One plaintext length shall be one octet. One plaintext length shall be 20 octets (160 bits).
- One plaintext length shall be the longest supported plaintext length less than or equal to 512 octets (4096 bits).

The evaluator will test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption.

AES-CCM Tests

It is not recommended that evaluators use values obtained from static sources such as http://csrc.nist.gov/groups/STM/cavp/documents/mac/ccmtestvectors.zip or use values not generated expressly to exercise the AES-CCM implementation.

The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths: Keys: All supported and selected key sizes (e.g., 128, 256 bits). Associated Data: Two or three values for associated data length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported associated data lengths, and 2^16 (65536) bytes, if supported. Payload: Two values for payload length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported payload lengths. Nonces: All supported nonce lengths (7, 8, 9, 10, 11, 12, 13) in bytes. Tag: All supported tag lengths (4, 6, 8, 10, 12, 14, 16) in bytes.

The testing for CCM consists of five tests. To determine correctness in each of the below tests, the evaluator shall compare the ciphertext with the result of encryption of the same inputs with a known good implementation.

Variable Associated Data Test
For each supported key size and associated data length, and any supported payload length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Payload Text
For each supported key size and payload length, and any supported associated data length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Nonce Test
For each supported key size and nonce length, and any supported associated data length, payload length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Tag Test
For each supported key size and tag length, and any supported associated data length, payload length, and nonce length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Decryption-Verification Process Test
To test the decryption-verification functionality of AES-CCM, for each combination of supported associated data length, payload length, nonce length, and tag length, the evaluator shall supply a key value and 15 sets of input plus ciphertext, and obtain the decrypted payload. Ten of the 15 input sets supplied should fail verification and five should pass.

FCS_COP.1(6) Cryptographic operation (Key Transport)

TSS
The evaluator shall verify the TSS provides a high level description of the RSA scheme and the cryptographic key size that is being used, and that the asymmetric algorithm being used for key transport is RSA. If more than one scheme/key size are allowed, then the evaluator shall make sure and test all combinations of scheme and key size. There may be more than one key size to specify - an RSA modulus size (and/or encryption exponent size), an AES key size, hash sizes, MAC key/MAC tag size.

If the KTS-OAEP scheme was selected, the evaluator shall verify that the TSS identifies the hash function, the mask generating function, the random bit generator, the encryption primitive and decryption primitive. If the KTS-KEM-KWS scheme was selected, the evaluator shall verify that the TSS identifies the key derivation method, the AES-based key wrapping method, the secret value encapsulation technique, and the random number generator.

Guidance
None.

Tests
For each supported key transport schema, the evaluator shall initiate at least 25 sessions that require key transport with an independently developed remote instance of a key transport entity, using known RSA key-pairs. The evaluator shall observe traffic passed from the sender-side and to the receiver-side of the TOE, and shall perform the following tests, specific to which key transport scheme was employed. If the KTS-OAEP scheme was selected, the evaluator shall perform the following tests:

- **Test 1:** The evaluator shall inspect each cipher text, C, produced by the RSA-OAEP encryption operation of the TOE and make sure it is the correct length, either 256 or 384 bytes depending on RSA key size. The evaluator shall also feed into the TOE's RSA-OEAP decryption operation some cipher texts that are the wrong length and verify that the erroneous input is detected and that the decryption operation exits with an error code.
- **Test 2:** The evaluator shall convert each cipher text, C, produced by the RSA-OAEP encryption operation of the TOE to the correct cipher text integer, c, and use the decryption primitive to compute em = RSADP((n,d),c) and convert em to the encoded message EM. The evaluator shall then check that the first byte of EM is 0x00. The evaluator shall also feed into the TOE's RSA-OEAP decryption operation some cipher texts where the first byte of EM was set to a value other than 0x00, and verify that the erroneous input is detected and that the decryption operation exits with an error code.
- **Test 3:** The evaluator shall decrypt each cipher text, C, produced by the RSA-OAEP encryption operation of the TOE using RSADP, and perform the OAEP decoding operation (described in NIST SP 800-56B section 7.2.2.4) to recover HA' || X. For each HA', the evaluator shall take the corresponding A and the specified hash algorithm and verify that HA' = Hash(A). The evaluator shall also force the TOE to perform some RSA-OAEP decryption where the A value is passed incorrectly, and the evaluator shall verify that an error is detected.
- **Test 4:** The evaluator shall check the format of the 'X' string recovered in OAEP.Test.3 to ensure that the format is of the form PS || 01 || K, where PS consists of zero or more consecutive 0x00 bytes and K is the transported keying material. The evaluator shall also feed into the TOE's RSA-OEAP decryption operation some cipher texts for which the resulting 'X' strings do not have the correct format (i.e., the leftmost non-zero byte is not 0x01). These incorrectly formatted 'X' variables shall be detected by the RSA-OEAP decrypt function.
- **Test 5:** The evaluator shall trigger all detectable decryption errors and validate that the returned error codes are the same and that no information is given back to the sender on which type of error occurred. The evaluator shall also validate that no intermediate results from the TOE's receiver-side operations are revealed to the sender.

If the KTS-KEM-KWS scheme was selected, the evaluator shall perform the following tests:

- **Test 1:** The evaluator shall inspect each cipher text, C, produced by KTS-KEM-KWS encryption operation of the TOE and make sure the length (in bytes) of the cipher text, cLen, is greater than nLen (the length, in bytes, of the modulus of the RSA public key) and that cLen - nLen is consistent with the byte lengths supported by the key wrapping algorithm. The evaluator shall feed into the KTS-KEM-KWS decryption operation a cipher text of unsupported length and verify that an error is detected and that the decryption process stops.
- **Test 2:** The evaluator shall separate the cipher text, C, produced by the sender-side of the TOE into its

C0 and C1 components and use the RSA decryption primitive to recover the secret value, Z, from C0. The evaluator shall check that the unsigned integer represented by Z is greater than 1 and less than n-1, where n is the modulus of the RSA public key. The evaluator shall construct examples where the cipher text is created with a secret value Z = 1 and make sure the KTS-KEM-KWS decryption process detects the error. Similarly, the evaluator shall construct examples where the cipher text is created with a secret value Z = n - 1 and make sure the KTS-KEM-KWS decryption process detects the error.

- **Test 3:** The evaluator shall attempt to successfully recover the secret value Z, derive the key wrapping key, KWK, and unwrap the KWA-cipher text following the KTS-KEM-KWS decryption process given in NISP SP 800-56B section 7.2.3.4. If the key-wrapping algorithm is AES-CCM, the evaluator shall verify that the value of any (unwrapped) associated data, A, that was passed with the wrapped keying material is correct The evaluator shall feed into the TOE's KTS-KEM-KWS decryption operation examples of incorrect cipher text and verify that a decryption error is detected. If the key-wrapping algorithm is AES-CCM, the evaluator shall attempt at least one decryption where the wrong value of A is given to the KTS-KEM-KWS decryption operation and verify that a decryption error is detected. Similarly, if the key-wrapping algorithm is AES-CCM, the evaluator shall attempt at least one decryption where the wrong nonce is given to the KTS-KEM-KWS decryption operation and verify that a decryption error is detected.
- **Test 4:** The evaluator shall trigger all detectable decryption errors and validate that the resulting error codes are the same and that no information is given back to the sender on which type of error occurred. The evaluator shall also validate that no intermediate results from the TOE's receiver-side operations (in particular, no Z values) are revealed to the sender.

FCS_COP.1(7) Cryptographic operation (Key Encryption)

TSS

Requirement met by the platform

If the platform provides the FEK encryption/decryption, then the evaluator shall examine the TSS to verify that it describes how the FEK encryption/decryption is invoked.

Requirement met by the TOE

The evaluator shall verify the TSS includes a description of the key size used for encryption and the mode used for the key encryption

Guidance
None.

Tests
The evaluation activity tests specified for AES in CBC mode in FCS_COP.1.1(1) in the underlying [AppPP] shall be performed.

FCS_KDF_EXT.1 Cryptographic Key Derivation Function

TSS
The evaluator shall verify the TSS includes a description of the key derivation function and shall verify the key derivation uses an approved derivation mode and key expansion algorithm according to SP 800-108 and SP 800-132.

Guidance
None.

Tests
None.

FCS_SMC_EXT.1 Submask Combining

TSS
If keys are XORed together to form an intermediate key, the TSS section shall identify how this is performed (e.g., if there are ordering requirements, checks performed, etc.). The evaluator shall also confirm that the TSS describes how the length of the output produced is at least the same as that of the FEK.

Guidance
None.

Tests
None.

FCS_VAL_EXT.2 Validation Remediation

TSS

The evaluator shall examine the TSS to determine which remediation options are supported for which authentication options.

Guidance
If the remediation functionality is configurable, the evaluator shall examine the operational guidance to ensure it describes how to configure the TOE to ensure the limits regarding validation attempts can be established.

Tests
The evaluator shall perform the following tests:

- **Test 1:** The evaluator shall determine the limit on the average rate of the number of consecutive failed authorization attempts. For each authentication factor supported, the evaluator will test the TOE by entering that number of incorrect authorization factors in consecutive attempts to access the protected data. If the limit mechanism includes any "lockout" period, the time period tested should include at least one such period. Then the evaluator will verify that the TOE behaves as described in the TSS.

5 Evaluation Activities for Objective SFRs

The PP-Module does not define any objective requirements.

6 Evaluation Activities for SARs

The PP-Module does not define any SARs beyond those defined within the App PP base to which it must claim conformance. It is important to note that a TOE that is evaluated against the PP-Module is inherently evaluated against this Base-PP as well. The App PP includes a number of Evaluation Activities associated with both SFRs and SARs. Additionally, the PP-Module includes a number of SFR-based Evaluation Activities that similarly refine the SARs of the Base-PPs. The evaluation laboratory will evaluate the TOE against the Base-PP and supplement that evaluation with the necessary SFRs that are taken from the PP-Module.

7 Required Supplementary Information

This Supporting Document has no required supplementary information beyond the ST, operational guidance, and testing.

Appendix A - References

Identifier	Title
[CC]	Common Criteria for Information Technology Security Evaluation - • Part 1: Introduction and General Model, CCMB-2017-04-001, Version 3.1 Revision 5, April 2017. • Part 2: Security Functional Components, CCMB-2017-04-002, Version 3.1 Revision 5, April 2017. • Part 3: Security Assurance Components, CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.
[AppPP]	Protection Profile for Application Software, Version 1.3
[FIPS140-2]	Federal Information Processing Standard Publication (FIPS-PUB) 140-2, Security Requirements for Cryptographic Modules, National Institute of Standards and Technology, March 19, 2007
[FIPS180-4]	Federal Information Processing Standards Publication (FIPS-PUB) 180-4, Secure Hash Standard, March, 2012
[FIPS186-4]	Federal Information Processing Standard Publication (FIPS-PUB) 186-4, Digital Signature Standard (DSS), National Institute of Standards and Technology, July 2013
[FIPS197]	Federal Information Processing Standards Publication (FIPS-PUB) 197, Specification for the Advanced Encryption Standard (AES), November 26, 2001
[FIPS198-1]	Federal Information Processing Standards Publication (FIPS-PUB) 198-1, The Keyed-Hash Message Authentication Code (HMAC), July 2008
[NIST800-38A]	NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation: Methods and Techniques, 2001 Edition
[NIST800-	NIST Special Publication 800-56A, Recommendation for Pair-Wise Key Establishment Schemes

56A]	Using Discrete Logarithm Cryptography (Revised), March 2007
[NIST800-56B]	NIST Special Publication 800-56B, Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography, August 2009
[NIST800-90]	NIST Special Publication 800-90, Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised), March 2007
[NIST800-132]	NIST Special Publication 800-132, Recommendation for Password-Based Key Derivation, December 2010
[NIST800-38F]	NIST Special Publication 800-38F,Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, December 2012