Requirements for Vetting Mobile Apps from the

Protection Profile for Application Software

Version: 1.3

2019-03-01

National Information Assurance Partnership

Revision History

Version	Date	Comment
v 1.0	2014-10-20	Initial release
v 1.1	2014-11-05	Addition to TLS cipher suite selections
v 1.2	2016-04-22	Added server-side TLS requirements (selection-based) Multiple clarification based on NIAP TRRT inquiries Refactored FDP_DEC_EXT.1 into separate components
v 1.3	2019-03-01	Incorporated available Technical Decisions Refactored FPT_TUD Added a selection to FTP_DIT Moved SWID Tags requirement Leveraged TLS Package Added equivalency section

Introduction

Purpose. This document presents functional and assurance requirements found in the *Protection Profile for Application Software* which are appropriate for vetting mobile application software ("apps") **outside** formal Common Criteria (ISO/IEC 15408) evaluations. Common Criteria evaluation, facilitated in the U.S. by the National Information Assurance Partnership (NIAP), is required for IA and IA-enabled products in National Security Systems according to CNSS Policy #11. Such evaluations, including those for mobile apps, must use the complete Protection Profile. However, even apps without IA functionality may impose some security risks, and concern about these risks has motivated the vetting of such apps in government and industry.

Using this document. This representation of the Protection Profile includes:

- *Security Functional Requirements* for use in evaluation. These are featured without the formal Assurance Activities specified in the Protection Profile, to assist the reader who is interested only in the requirements.

  It also includes, in tables shown later, particular types of security functional requirements that are not strictly required in all cases. These are:

  - *Selection-based Security Functional Requirements* which become required when certain selections are made inside the regular Security Functionality Requirements (as indicated by the **[selection:]** construct).
  - *Objective Security Functional Requirements* which are highly desired but not yet widely-available in commercial technology.
  - *Optional Security Functional Requirements* which are available for evaluation and

which some customers may insist upon.

- *Security Assurance Requirements* which relate to developer support for the product under evaluation, development processes, and other non-functionality security relevant requirements.

In addition to providing these security requirements for vetting apps, this document provides a basis for discussion and consideration of the vetting provided by commercially-available app stores. This document does not imply to Authorizing Officials that the vetting provided by commercially-available app stores is either adequate or inadequate for the context in which they must weigh risks. Rather, it is intended to help inform and support decision-making with regard to investment in app vetting processes.

Security Functional Requirements

Random Bit Generation Services

The application shall [**selection**:

- *use no DRBG functionality*,
- *invoke platform-provided DRBG functionality*,
- *implement DRBG functionality*

] for its cryptographic operations.

Application Note: The selection *invoke platform-provided DRBG functionality* should only be chosen for direct invocations of the platform DRBG, calls to platform protocols that may then call the platform's DRBG are not directly using DRBG functionality and should select *use no DRBG functionality*.
If *implement DRBG functionality* is chosen, then additional FCS_RBG_EXT.2 elements shall be included in the ST.
In this requirement, cryptographic operations include all cryptographic key generation/derivation/agreement, IVs (for certain modes), as well as protocol-specific random values. Cryptographic operations in this requirement refer to the other cryptographic requirements in this PP, not additional functionality that is not in scope.

Cryptographic Key Generation Services

The application shall [**selection**:

- *generate no asymmetric cryptographic keys*,
- *invoke platform-provided functionality for asymmetric key generation*,
- *implement asymmetric key generation*

].

Application Note: If *implement asymmetric key generation* or *invoke platform-provided functionality for asymmetric key generation* is chosen, then additional FCS_CKM.1/1 elements shall be included in the ST.

Storage of Credentials

The application shall [**selection**:

- *not store any credentials*,
- *invoke the functionality provided by the platform to securely store [**assignment**: list of credentials]* ,
- *implement functionality to securely store [**assignment**: list of credentials] according to [**selection**: FCS_COP.1(1), FCS_CKM.1(3)]*

] to non-volatile memory.

Application Note: This requirement ensures that persistent credentials (secret keys, PKI private keys, passwords, etc) are stored securely, and never persisted in cleartext form. Application developers are encouraged to use platform mechanisms for the secure storage of credentials. Depending on the platform that may include hardware-backed protection for credential storage. Application developers must choose a selection, or multiple selections, based on all credentials that the application stores. If *not store any credentials* is selected then the application must not store any credentials. If *invoke the functionality provided by the platform to securely store* is selected then the Application developer must closely review the EA for their platform and provide documentation indicating which platform mechanisms are used to store credentials. If *implement functionality to securely store credentials* is selected, then the following components must be included in the ST: FCS_COP.1/1 or FCS_CKM.1/3. If other cryptographic operations are used to implement the secure storage of credentials, the corresponding requirements must be included in the ST. If the OS is Linux and Java KeyStores are used to store credentials, *implement functionality to securely store credentials* must be selected.

Access to Platform Resources

The application shall restrict its access to [**selection**:

- *no hardware resources*,
- *network connectivity*,
- *camera*,
- *microphone*,
- *location services*,
- *NFC*,
- *USB*,
- *Bluetooth*,
- *[**assignment**: list of additional hardware resources]*

].

Application Note: The intent is for the evaluator to ensure that the selection captures all hardware resources which the application accesses, and that these are restricted to those which are justified. On some platforms, the application must explicitly solicit permission in order to access hardware resources. Seeking such permissions, even if the application does not later make use of the hardware resource, should still be considered access. Selections should be expressed in a manner consistent with how the application expresses its access needs to the underlying platform. For example, the platform may provide *location services* which implies the potential use of a variety of hardware resources (e.g. satellite receivers, WiFi, cellular radio) yet *location services* is the proper selection. This is because use of these resources can be inferred, but also because the actual usage may vary based on the particular platform. Resources that do not need to be explicitly identified are those which are ordinarily used by any application such as central processing units, main memory, displays, input devices (e.g. keyboards, mice), and persistent storage devices provided by the platform.

The application shall restrict its access to [**selection**:

- *no sensitive information repositories*,
- *address book*,
- *calendar*,
- *call lists*,
- *system logs*,
- *[**assignment**: list of additional sensitive information repositories]*

] .

Application Note: *Sensitive information repositories* are defined as those collections of sensitive data that could be expected to be shared among some applications, users, or user roles, but to which not all of these would ordinarily require access.

Network Communications

The application shall restrict network communication to [**selection**:

- *no network communication*,
- *user-initiated communication for [**assignment**: list of functions for which the user can initiate network communication]*,
- *respond to [**assignment**: list of remotely initiated communication]*,
- *[**assignment**: list of application-initiated network communication]*

] .

Application Note: This requirement is intended to restrict both inbound and outbound network communications to only those required, or to network communications that are user initiated. It does not apply to network communications in which the application may generically access the filesystem which may result in the platform accessing remotely mounted drives/shares.

Encryption Of Sensitive Application Data

The application shall [**selection**:

- *leverage platform-provided functionality to encrypt sensitive data*,
- *implement functionality to encrypt sensitive data as defined in the PP-Module for File Encryption*,
- *protect sensitive data in accordance with FCS_STO_EXT.1*,
- *not store any sensitive data*

] in non-volatile memory.

Application Note: If *implement functionality to encrypt sensitive data as defined in the PP-Module for File Encryption* is selected, the TSF must claim conformance to a PP-Configuration that includes the File Encryption PP-Module.
Any file that may potentially contain sensitive data (to include temporary files) shall be protected. The only exception is if the user intentionally exports the sensitive data to non-protected files. ST authors should select *protect sensitive data in accordance with FCS_STO_EXT.1* for the sensitive data that is covered by the FCS_STO_EXT.1 SFR.

Supported Configuration Mechanism

The application shall [**selection**: *invoke the mechanisms recommended by the platform vendor for storing and setting configuration options*, *implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption*]

Application Note: Configuration options that are stored remotely are not subject to this requirement. is generally not considered part of configuration options and should be stored according to fdp_dar_ext.1 or fcs_sto_ext.1.

Secure by Default Configuration

The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

Application Note: Default credentials are credentials (e.g., passwords, keys) that are automatically (without user interaction) loaded onto the platform during application installation. Credentials that are generated during installation using requirements laid out in FCS_RBG_EXT.1 are not by definition default credentials.

The application shall be configured by default with file permissions which protect the application binaries and data files from modification by normal unprivileged users.

Application Note: The precise expectations for file permissions vary per platform but the general intention is that a trust boundary protects the application and its data.

Specification of Management Functions

The TSF shall be capable of performing the following management functions [**selection**:

- *no management functions*,
- *enable/disable the transmission of any information describing the system's hardware, software, or configuration* ,
- *enable/disable the transmission of any PII* ,
- *enable/disable transmission of any application state (e.g. crashdump) information*,
- *enable/disable network backup functionality to [**assignment**: list of enterprise or commercial cloud backup systems]* ,
- *[**assignment**: list of other management functions to be provided by the TSF]*

] .

Application Note: This requirement stipulates that an application needs to provide the ability to enable/disable only those functions that it actually implements. The application is not responsible for controlling the behavior of the platform or other applications.

User Consent for Transmission of Personally Identifiable Information

The application shall [**selection**:

- *not transmit PII over a network* ,
- *require user approval before executing [**assignment**: list of functions that transmit PII over a network]*

] .

Application Note: This requirement applies only to PII that is specifically requested by the application; it does not apply if the user volunteers PII without prompting from the application into a general (or inappropriate) data field. A dialog box that declares intent to send PII presented to the user at the time the application is started is sufficient to meet this requirement.

Use of Supported Services and APIs

The application shall use only documented platform APIs.

Application Note: The definition of *documented* may vary depending upon whether the application is provided by a third party (who relies upon documented platform APIs) or by a platform vendor who may be able to guarantee support for platform APIs.

Anti-Exploitation Capabilities

The application shall not request to map memory at an explicit address except for [**assignment**: *list of explicit exceptions*].

Application Note: Requesting a memory mapping at an explicit address subverts address space layout randomization (ASLR).

The application shall [**selection**:

- *not allocate any memory region with both write and execute permissions* ,
- *allocate memory regions with write and execute permissions for only [***assignment***: list of functions performing just-in-time compilation]*

] .

Application Note: Requesting a memory mapping with both write and execute permissions subverts the platform protection provided by DEP. If the application performs no just-in-time compiling, then the first selection must be chosen.

The application shall be compatible with security features provided by the platform vendor.

Application Note: This requirement is designed to ensure that platform security features do not need to be disabled in order for the application to run.

The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

Application Note: The purpose of this requirement is to help ensure the integrity of application binaries by supporting file protection mechanisms such as directory-level file permissions and application whitelisting. A user-modifiable file for purposes of this requirement is a file that is writable by an unprivileged user of the application -- either directly through application execution or independently of the application. If the application runs in the context of the application user, then the application should not be able to write to the directory containing the application binaries -- regardless of whether the files are configuration data, audit data, or temporary files. Executables and user-modifiable files may not share the same parent directory, but may share directories above the parent.

The application shall be built with stack-based buffer overflow protection enabled.

Integrity for Installation and Update

The application shall [**selection**: *provide the ability*, *leverage the platform*] to check for updates and patches to the application software.

Application Note: This requirement is about the ability to "check" for updates. The actual installation of any updates should be done by the platform. This requirement is intended to ensure that the application can check for updates provided by the vendor, as updates provided by another source may contain malicious code.

The application shall [**selection**: *provide the ability*, *leverage the platform*] to query the current version of the application software.

The application shall not download, modify, replace or update its own binary code.

Application Note: This requirement applies to the code of the application; it does not apply to mobile code technologies that are designed for download and execution by the application.

The application installation package and its updates shall be digitally signed such that its platform can cryptographically verify them prior to installation.

Application Note: The specifics of the verification of installation packages and updates involves requirements on the platform (and not the application), so these are not fully specified here.

The application is distributed [**selection**: *with the platform OS* , *as an additional software package to the platform OS*].

Application Note: Application software that is distributed as part of the platform operating system is not required to be package for installation or uninstallation. If "as an additional software package to the OS" is selected the requirements from FPT_TUD_EXT.2 must be included in the ST.

Use of Third Party Libraries

The application shall be packaged with only [**assignment**: *list of third-party libraries*].

Application Note: The intention of this requirement is for the evaluator to discover and document whether the application is including unnecessary or unexpected third-party libraries. This includes adware libraries which could present a privacy threat, as well as ensuring documentation of such libraries in case vulnerabilities are later discovered.

Software Identification and Versions

The application shall be versioned with [**selection**: *SWID tags that comply with minimum requirements from ISO/IEC 19770-2:2015* , *[**assignment**: other version information]*] .

Application Note: The use of SWID tag to identify application software is a requirement for DOD IT based on DoD Instruction 8500.01 which requires the use of SCAP which includes SWID tags per the NIST standard. The PP selection of "other version information" will be removed in the next major release of this protection profile. Vendors should begin to version software with valid SWID tags.

Valid SWID tags must contain a SoftwareIdentity element and an Entity element as defined in the ISO/IEC 19770-2:2015 standard. SWID tags must be stored with a .swidtag file extensions as defined in the ISO/IEC 19770-2:2015.

Protection of Data in Transit

The application shall [**selection**:

- *not transmit any [**selection**: data, sensitive data]* ,
- *encrypt all transmitted [**selection**: sensitive data, data] with [**selection**: HTTPS in accordance with FCS_HTTPS_EXT.1, TLS as defined in the TLS Package, DTLS as defined in the TLS Package, SSH as conforming to the Extended Package for Secure Shell, IPsec as defined in the PP-Module for VPN Client]* ,
- *invoke platform-provided functionality to encrypt all transmitted sensitive data with [**selection**: HTTPS, TLS, DTLS, SSH]* ,
- *invoke platform-provided functionality to encrypt all transmitted data with [**selection**: HTTPS, TLS, DTLS, SSH]*

] between itself and another trusted IT product.

Application Note: Encryption is not required for applications transmitting data that is not sensitive.

If *encrypt all transmitted* is selected and *TLS* is selected, then evaluation of elements from either FCS_TLSC_EXT.1 or FCS_TLSS_EXT.1 is required.

If *encrypt all transmitted* is selected and *HTTPS* is selected, FCS_HTTPS_EXT.1 is required.

If *encrypt all transmitted* is selected and *DTLS* is selected, FCS_DTLS_EXT.1 is required.

If *encrypt all transmitted* is selected and *SSH* is selected, the TSF shall be validated against the *Extended Package for Secure Shell*.

If *encrypt all trasnmitted* is selected and *IPsec* is selected, the TSF must claim conformance to a *PP-Configuration that includes the VPN Client PP-Module*

If *encrypt all transmitted* is selected the corresponding FCS_COP.1 requirements will be included.

Security Assurance Requirements

Selection-Based Security Functional Requirements

Random Bit Generation from Application

The application shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [**selection**: *Hash_DRBG (any)*, *HMAC_DRBG (any)*, *CTR_DRBG (AES)*]

Application Note: This requirement shall be included in STs in which *implement DRBG functionality* is chosen in FCS_RBG_EXT.1.1. The ST author should select the standard to which the RBG services comply (either SP 800-90A or FIPS 140-2 Annex C).
SP 800-90A contains three different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used (if SP 800-90A is selected), and include the specific underlying cryptographic primitives used in the requirement or in the TSS. While any of the identified hash functions (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed.

The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [**selection**:

- *a software-based noise source*,
- *a hardware-based noise source*,
- *no other noise source*

] with a minimum of [**selection**:

- *128 bits*,
- *256 bits*

] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

Application Note: This requirement shall be included in STs in which *implement DRBG functionality* is chosen in FCS_RBG_EXT.1.1. For the first selection in this requirement, the ST author selects 'software-based noise source' if any additional noise sources are used as input to the application's DRBG. Note that the application must use the platform's DRBG to seed its DRBG.
In the second selection in this requirement, the ST author selects the appropriate number of bits of entropy that corresponds to the greatest security strength of the algorithms included in the ST. Security strength is defined in Tables 2 and 3 of NIST SP 800-57A. For example, if the implementation includes 2048-bit RSA (security strength of 112 bits) and AES 256 (security strength 256 bits), then the ST author would select 256 bits.

Cryptographic Asymmetric Key Generation

The **application** shall [**selection**:

- *invoke platform-provided functionality*,
- *implement functionality*

] **to generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm** [**selection**:

- ***[RSA schemes]*** *using cryptographic key sizes of* ***[2048-bit or greater]*** *that meet the following* ***FIPS PUB 186-4, "Digital Signature Standard (DSS), Appendix B.3"*** ,
- ***[ECC schemes]*** *using* ***[“NIST curves” P-256, P-384 and [selection: P-521 , no other curves]]****that meet the following:* ***[FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4]*** ,
- ***[FFC schemes]*** *using cryptographic key sizes of* ***[2048-bit or greater]*** *that meet the following:* ***[FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.1]*** ,
- *[FFC Schemes]* ***using Diffie-Hellman group 14*** *that meet the following:* ***RFC 3526, Section 3*** ,
- *[FFC Schemes]* ***using “safe-prime” groups*** *that meet the following:* ***NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”*** *and [****selection****: RFC 3526, RFC 7919]*

].

Application Note: The ST author shall select all key generation schemes used for key establishment and entity authentication. When key generation is used for key establishment, the schemes in FCS_CKM.2.1 and selected cryptographic protocols must match the selection. When key generation is used for entity authentication, the public key is expected to be associated with an X.509v3 certificate.
If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.

Password Conditioning

Refinement: A password/passphrase shall perform [Password-based Key Derivation Functions] in accordance with a specified cryptographic algorithm as specified in FCS_COP.1/4, with [assignment: positive integer of 1,000 or more] iterations, and output cryptographic key sizes [**selection**: *128*, *256*] that meet the following **[NIST SP 800-132]**.

The TSF shall generate salts using a RBG that meets FCS_RGB_EXT.1 and with entropy corresponding to the security strength selected for PBKDF in FCS_CKM.1.1/3

Application Note: This should be included if selected in FCS_STO_EXT.1
Conditioning can be performed using one of the identified hash functions or the process described in NIST SP 800-132; the method used is selected by the ST Author. SP 800-132 requires the use of a pseudo-random function (PRF) consisting of HMAC with an approved hash function. The ST author selects the hash function used, also includes the appropriate requirements for HMAC and the hash function.
Appendix A of SP 800-132 recommends setting the iteration count in order to increase the computation needed to derive a key from a password and, therefore, increase the workload of performing a password recovery attack. A significantly higher value is recommended to ensure optimal security. This value is expected to increase to a minimum of 10,000 in a future iteration based on SP800-63.

Cryptographic Key Establishment

The application shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method:

[**selection**:

- ***[RSA-based key establishment schemes]*** *that meets the following:* ***[NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography"]*** ,
- ***[RSA-based key establishment schemes]*** *that meet the following:* ***RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1"*** ,
- ***[Elliptic curve-based key establishment schemes]*** *that meets the following:* ***[NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"]*** ,
- ***[Finite field-based key establishment schemes]*** *that meets the following:* ***[NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"]*** ,
- ***[Key establishment scheme using Diffie-Hellman group 14]*** *that meets the following:* ***RFC 3526, Section 3*** ,
- ***[FFC Schemes using "safe-prime" groups]*** *that meet the following:* ***'NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"*** *and [**selection**: RFC 3526, RFC 7919].*

].

Application Note: The ST author shall select all key establishment schemes used for the selected cryptographic protocols. TLS requires cipher suites that use RSA-based key establishment schemes.
The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B; however, Section 9 relies on implementation of other sections in SP 800-56B. If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.
The elliptic curves used for the key establishment scheme shall correlate with the curves specified in FCS_CKM.1.1/1.
The domain parameters used for the finite field-based key establishment scheme are specified by the key generation according to FCS_CKM.1.1/1.

Cryptographic Operation - Encryption/Decryption

The **application** shall perform *encryption/decryption* in accordance with a specified cryptographic algorithm [**selection**:

- *AES-CBC (as defined in NIST SP 800-38A) mode* ,
- *AES-GCM (as defined in NIST SP 800-38D) mode*,
- *AES-XTS (as defined in NIST SP 800-38E) mode*

] and cryptographic key sizes [**selection**: *128-bit*, *256-bit*] .

Application Note: This is dependent on implementing cryptographic functionality, as in FTP_DIT_EXT.1.
For the first selection, the ST author should choose the mode or modes in which AES operates. For the second selection, the ST author should choose the key sizes that are supported by this functionality. 128-bit key size is required in order to comply with certain TLS implementations.

Cryptographic Operation - Hashing

The **application** shall perform *cryptographic hashing* services in accordance with a specified cryptographic algorithm [**selection**:

- *SHA-1*,
- *SHA-256*,
- *SHA-384*,
- *SHA-512*,
- *no other*

] and message digest sizes [**selection**:

- *160*,
- *256*,
- *384*,
- *512*,
- *no other*

] bits that meet the following: FIPS Pub 180-4.

Application Note: This is dependent on implementing cryptographic functionality, as in FTP_DIT_EXT.1.
Per NIST SP 800-131A, SHA-1 for generating digital signatures is no longer allowed, and SHA-1 for verification of digital signatures is strongly discouraged as there may be risk in accepting these signatures.
SHA-1 is currently included in order to comply with the TLS. If the TLS package is included in the ST, the hashing algorithms selection for FCS_COP.1(2) must match the hashing algorithms used in the mandatory and selected cipher suites of the TLS package. Vendors are strongly encouraged to implement updated protocols that support the SHA-2 family; until updated protocols are supported, this PP allows support for SHA-1 implementations in compliance with SP 800-131A.
The intent of this requirement is to specify the hashing function. The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used (for example, SHA 256 for 128-bit keys).

Cryptographic Operation - Signing

The **application** shall perform *cryptographic signature services (generation and verification)* in accordance with a specified cryptographic algorithm [**selection**:

- ***RSA schemes** using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 4* ,
- ***ECDSA schemes** using “NIST curves” P-256, P-384 and [**selection**: P-521, no other curves] that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5*

] .

Application Note: This is dependent on implementing cryptographic functionality, as in FTP_DIT_EXT.1.
The ST Author should choose the algorithm implemented to perform digital signatures; if more than one algorithm is available, this requirement should be iterated to specify the functionality. For the algorithm chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

Cryptographic Operation - Keyed-Hash Message Authentication

The **application** shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm

- HMAC-SHA-256

and [**selection**:

- *SHA-1*,
- *SHA-384*,
- *SHA-512*,
- *no other algorithms*

] with key sizes [**assignment**: *key size (in bits) used in HMAC*] and message digest sizes 256 and [**selection**: *160*, *384*, *512*, *no other size*] bits that meet the following: FIPS Pub 198-1 *The Keyed-Hash Message Authentication Code* and FIPS Pub 180-4 *Secure Hash Standard*.

Application Note: This is dependent on implementing cryptographic functionality, as in FTP_DIT_EXT.1.
The intent of this requirement is to specify the keyed-hash message authentication function used for key establishment purposes for the various cryptographic protocols used by the application (e.g., trusted channel). The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1/1.

HTTPS Protocol

The application shall implement the HTTPS protocol that complies with RFC 2818.

The application shall implement HTTPS using TLS as defined in the TLS package.

The application shall [**selection**: *not establish the application-initiated connection* , *notify the user and not establish the user-initiated connection* , *notify the user and request authorization to establish the user-initiated connection*] if the peer certificate is deemed invalid.

Application Note: Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

X.509 Certificate Validation

The application shall [**selection**: *invoked platform-provided functionality* , *implement functionality*] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The application shall validate the revocation status of the certificate using [**selection**: *the Online Certificate Status Protocol (OCSP) as specified in RFC 2560* , *a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3* , *a Certificate Revocation List (CRL) as specified in RFC 5759* , *an OCSP TLS Status Request Extension (i.e., OCSP stapling) as specified in RFC 6066*] .
- The application shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.
  - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the extendedKeyUsage field.

Application Note: FIA_X509_EXT.1.1 lists the rules for validating certificates. The ST author shall select whether revocation status is verified using OCSP or CRLs. FIA_X509_EXT.2 requires that certificates are used for HTTPS, TLS and DTLS; this use requires that the extendedKeyUsage rules are verified.
Regardless of the selection of *implement functionality* or *invoke platform-provided functionality*, the validation is expected to end in a trusted root CA certificate in a root store managed by the platform.

The application shall treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: This requirement applies to certificates that are used and processed by the TSF and restricts the certificates that may be added as trusted CA certificates.

X.509 Certificate Authentication

The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [**selection**: *HTTPS*, *TLS*, *DTLS*, *SSH*, *IPsec*].

Application Note: The ST author's selection shall match the selection in FTP_DIT_EXT.1.1.

When the application cannot establish a connection to determine the validity of a certificate, the application shall [**selection**: *allow the administrator to choose whether to accept the certificate in these cases* , *accept the certificate* , *not accept the certificate*].

Application Note: Often a connection must be established to perform a verification of the revocation status of a certificate - either to download a CRL or to perform OCSP. The selection is used to describe the behavior in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate valid according to all other rules in FIA_X509_EXT.1, the behavior indicated in the selection shall determine the validity. The TOE must not accept the certificate if it fails any of the other validation rules in FIA_X509_EXT.1.

Integrity for Installation and Update

The application shall be distributed using the format of the platform-supported package manager.

The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.

Application Note: Applications software bundled with the system/firmware image are not subject to this requirement if the user is unable to remove the application through means provided by the OS.

Objective Security Functional Requirements

Use of Supported Services and APIs

The application [**selection**: *shall use platform-provided libraries*, *does not implement functionality*] for parsing [**assignment**: *list of formats parsed that are included in the IANA MIME media types*].

Application Note: The IANA MIME types are listed at http://www.iana.org/assignments/media-types and include many image, audio, video, and content file formats. This requirement does not apply if providing parsing services is the purpose of the application.

Optional Security Functional Requirements

Cryptographic Symmetric Key Generation

The **application** shall generate **symmetric** cryptographic keys **using a Random Bit Generator as specified in FCS_RBG_EXT.1** and specified cryptographic key sizes [**selection**:

- *128 bit*,
- *256 bit*

].

Application Note: Symmetric keys may be used to generate keys along the key chain.