Commercial Solutions for Classified (CSfC) Selections for Transport Layer Security (TLS) Software Applications

Overview

Transport Layer Security (TLS) Software Application products (i.e., TLS Client as defined in the Commercial Solutions for Classified (CSfC) Mobile Access (MA) Capability Package (CP)) used in CSfC solutions must be validated by National Information Assurance Partnership (NIAP)/Common Criteria Evaluation and Validation Scheme (CCEVS) or Common Criteria Recognition Arrangement (CCRA) partnering schemes as complying with the current requirements of NIAP's:

- Protection Profile for Application Software Version 2.0; and
- Functional Package for Transport Layer Security (TLS) Version 2.1; and
- Functional Package for X.509 Version 1.0

This validated compliance must include the selectable requirements contained in this document.

TLS can be used/implemented in many different ways, threats and technology continuously progress, and TLS continues to evolve, which may cause the below selections to change or become obsolete. Vendors are encouraged to review the CSfC CPs to ensure the product functions appropriately in CSfC solutions. The objective of the below selections is to provide information to enable the use of the Commercial National Security Algorithm (CNSA) Suite and facilitate the use of TLS Software Applications in CSfC solutions.

Please provide questions, comments on usability, applicability, and/or shortcomings to the CSfC Program (csfc@nsa.gov).

Notes

Note 1: The following selections apply to CSfC TLS Software Application functionality. If needed, functionality and/or configurations outside the scope of a CSfC TLS Software Application that conflict with the CSfC selections could be NIAP validated without using a separate iteration of the Security Functional Requirement (SFR) if the product can be configured to use only the CSfC selections. The Security Target (ST) author should document a specific CSfC Software Application configuration in the product's Administrative Guide with a note that the configuration should be considered the NIAP-certified evaluated configuration for CSfC Software Application Use Cases. The CSfC Software Application configuration should be used to validate compliance with CSfC selections.

Note 2: The below SFRs/Selections contain some mandatory SFRs without Selections or modifications. The exclusion of other mandatory SFRs in the below Selections does not indicate that mandatory PP SFRs are not required (i.e., Compliance with the requirements as prescribed by the PP, Functional Packages, and outlined in the Overview Section above are required). Some mandatory SFRs are included in the below Selections to highlight some SFRs relevant to CSfC Software Application.

Note 3: Some of the below SFRs contain CSfC selections that are only applicable based on specific configurations but may require more than one selection due to dependencies on other SFR that specify the selections required, and/or are only permitted for use in CSfC solutions for specific Use Cases. If needed, the MA CP, CSfC Application Notes, and the NIAP PP Application Notes provide more details on selection dependencies. For example:

- In FCS_CKM.1.1/AK, selecting RSA 3072 would also require selecting ffdhe-3072 or Elliptic Curve P-384 to enable compliance with the selections for FCS_CKM.2.1
- In FCS_CKM.1.1/AK, either one or more selections for Elliptic Curve/Finite Field Diffie-Hellman are required based on the selections in FCS_TLSC_EXT.1
- In FCS_CKM.1.1/AK, for CSfC solutions Leighton-Micali Signature (LMS) and Xtended Merkle Signature Scheme (XMSS) are encouraged but only permitted for digitally signing firmware and software.

Note 4: CSfC TLS software applications must be configured to support (i.e., implement the functionality or invoke platform-provided functionality) and have documented configurations in the ST and Admin Guide for the functionality described in the CSfC Selections for the below SFR:

- FCS_HTTPS_EXT.1
- FCS_HTTPS_EXT.2
- Functional Package for Transport Layer Security (TLS) Selections

Even if the SFR aren't required by the PP due to specific selections made by the ST author, the functionality is required to enable TLS software applications to comply with CSfC CP requirements (e.g., CNSA Suite, TLS capabilities). For example, if the TLS software application invokes platform-provided functionality in FTP_DIT_EXT.1.1, then per PP design, the SFR listed above may not be evaluated, but CSfC requires supported and documented functionality for the SFR (i.e., demonstrate equivalent functionality as noted above) as part of CSfC Components List product eligibility.

Document Conventions

The conventions used in descriptions of the document are as follows:

- Assignment completed within a selection in the PP: the completed assignment text is indicated with *italicized and underlined text*
- Assignment partially completed in the PP: indicated with *italicized text*
- Refinement text is indicated with ~~strikethroughs~~
- CSfC specific selections, refinements (e.g., underline, strikethrough) are highlighted in light blue text (i.e., CSfC mandatory completed assignments/selections unless otherwise indicated by the `light blue Courier New Text` "at least one of the following underlined selections").
- Additional clarifying text or CSfC specific language is indicated with `light blue Courier New Text`
- Links to sources, additional information, and email addresses are indicated with blue underlined text.

Protection Profile for Application Software Version 2.0 Selections

FCS_CKM_EXT.1.1 The application shall [**selection**:

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

].

FCS_RBG_EXT.1.1 The application shall [selection:

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

] for its cryptographic operations.

FCS_STO_EXT.1.1 The application shall [**selection**:

- *not store any credentials*
- *invoke the functionality provided by the platform to securely store [**assignment**: list of credentials]*
- *securely store [**assignment**: list of credentials] with platform provided [**selection**:*
  - *[**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 size of 256-bits.
  - *PBKDF2 function that uses [**selection**:*
    - ~~*HMAC-SHA256*~~
    - *HMAC-SHA384*
    - *HMAC-SHA512*

    *] with [**assignment**: positive integer of 1,000 or greater] iterations and output cryptographic key size of [**assignment**: positive integer of 256 of greater] bits that meet the following [NIST SP 800-132].*

  ]
- *implement functionality to securely store [**assignment**: list of credentials] according to [**selection**: FCS_COP.1/SKC, FCS_PBKDF_EXT.1]*

] 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 "**securely store**" is selected, the application shall leverage platform

cryptographic APIs to implement storage of credentials. If "**implement functionality to securely store credentials**" is selected, then the following components must be included in the ST: (FCS_COP.1/SKC and FCS_SNI_EXT.1) or FCS_PBKDF_EXT.1. If the OS is Linux and Java KeyStores are used to store credentials, "**implement functionality to securely store credentials**" must be selected.

FTP_DIT_EXT.1.1 The application shall [**selection**:

- ~~*not transmit any [selection, choose one of: data, sensitive data]*~~
- *encrypt all transmitted [**selection, choose one of**:* ~~*sensitive data,*~~ *data] with [selection:*
  - *HTTPS as a client in accordance with FCS_HTTPS_EXT.1 and FCS_HTTPS_EXT.2*
  - ~~*HTTPS as a server in accordance with FCS_HTTPS_EXT.1*~~
  - ~~*HTTPS as a server with support for mutual authentication in accordance with FCS_HTTPS_EXT.1 and FCS_HTTPS_EXT.2*~~
  - ~~*TLS as a server as defined in Functional Package for Transport Layer Security (TLS), version 2.1 and also supports functionality for [selection: mutual authentication, none]*~~
  - *TLS as a client as defined in Functional Package for Transport Layer Security (TLS), version 2.1*
  - ~~*DTLS as a server as defined in Functional Package for Transport Layer Security (TLS), version 2.1 and also supports functionality for [selection: mutual authentication, none]*~~
  - ~~*DTLS as a client as defined in Functional Package for Transport Layer Security (TLS), version 2.1*~~
  - ~~*SSH as defined in the Functional Package for Secure Shell (SSH), version 2.0*~~
  - ~~*IPsec as defined in the VPN Client PP-Module, version 3.0*~~

  *] for [**assignment**:* `all network traffic to include to/from a TLS Protected Server` ~~*function(s)*~~*] using certificates as defined in the Functional Package for X.509*

- ~~*invoke platform-provided functionality to encrypt all transmitted sensitive data with [selection: HTTPS, TLS, DTLS, SSH, IPsec] for [assignment: function(s)] using certificates as defined in the Functional Package for X.509*~~
- *invoke platform-provided functionality to encrypt all transmitted data with [selection: HTTPS, TLS,* ~~*DTLS, SSH, IPsec*~~*] for [**assignment**:* `all network traffic to include to/from a TLS Protected Server` ~~*function(s)*~~*] using certificates as defined in the Functional Package for X.509*

] between itself and another trusted IT product.

FCS_CKM.1.1/AK 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**:

- *CNSA 2.0 Compliant Algorithms: [**selection**:*
  - ***Leighton-Micali Signature Algorithm** using the parameter sets [**selection**: LMS_SHAKE_M24_H5, LMS_SHAKE_M24_H10, LMS_SHAKE_M24_H15, LMS_SHAKE_M24_H25, LMS_SHAKE_M32_H5, LMS_SHAKE_M32_H10, LMS_SHAKE_M32_H15, LMS_SHAKE_M32_H25, LMS_SHA256_M24_H5, LMS_SHA256_M24_H10, LMS_SHA256_M24_H15, LMS_SHA256_M24_H25, LMS_SHA256_M32_H5, LMS_SHA256_M32_H10, LMS_SHA256_M32_H15, LMS_SHA256_M32_H25] that meet the following [NIST SP 800-208, "Recommendation for Stateful Hash-Based Signature Schemes"]*
  - ***eXtended Merkle Signature Scheme Algorithm** using the parameter sets [**selection**: XMSS-SHA2_10_192, XMSS-SHA2_16_192, XMSS-SHA2_20_192, XMSS-SHA2_10_256, XMSS-SHA2_16_256, XMSS-SHA2_20_256, XMSS-SHAKE_10_192, XMSS-SHAKE_16_192, XMSS-SHAKE_20_192, XMSS-SHAKE_10_256, XMSS-SHAKE_16_256, XMSS-SHAKE_20_256] that meets the following: [NIST SP 800-208, "Recommendation for Stateful Hash-Based Signature Schemes"]*
  - ***Module-Lattice-Based Key-Encapsulation Mechanism Standard** using the parameter set ML-KEM-1024 that meets the following: [FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard]*
  - ***Module-Lattice-Based Digital Signature Standard** using the parameter set ML-DSA-87 that meets the following [FIPS 204, Module-Lattice-Based Digital Signature Standard]*

  ]

- *CNSA 1.0 Compliant Algorithms: [**selection**:*
  - *[**RSA schemes**] using cryptographic key sizes of [3072-bit or greater] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)," Appendix A.1]*
  - *[**ECC schemes**] using [“NIST curves” P-384 and [**selection**: ~~P-521~~, no other curves]] that meet the following: [FIPS PUB 186-5, “Digital Signature Standard (DSS),” Appendix A.2]*
  - *[**FFC Schemes**] using [“safe-prime” groups] [**selection**: MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe-3072, ffdhe-4096, ffdhe-6144, ffdhe-8192] 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:

See Note 3 above for example SFR dependencies for CSfC solutions.

As of publication, for CSfC solutions, if FFDHE is selected for HTTPS/TLS, RFC 7919 ffdhe-3072 must be used.

In 2027, where applicable ML-KEM-1024, ML-DSA-87, LMS, and/or XMSS will be required for new CSfC Components List components. See the CSfC CP for more details.

If RSA is selected, 3072 must be selected. For RSA, the TOE can also support 4096, 6144, and 8192 but must support 3072. See Application Note for FCS_COP.1.1/SigGen for more details.

FCS_CKM.2.1 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**:

- *CNSA 2.0 Compliant Algorithm:*
  Module-Lattice-Based Key-Encapsulation Mechanism Standard using the parameter set ML-KEM-1024 *that meets the following: [FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard]*
- *CNSA 1.0 Compliant Algorithms: [**selection**:*
  - ~~*[RSA-based key establishment schemes] that meet the following: [NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography"]*~~
  - *[**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"**]*
  - *[**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: As of publication, for CSfC solutions, if FFDHE is selected for HTTPS/TLS, RFC 7919 group ffdhe3072 must be used. In 2027, where applicable, ML-KEM-1024 will be required for new CSfC Components List components. See the MA CP for more details.

FCS_COP.1.1/Hash The **application** shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [**selection**:

- *SHA-256*
- *SHA-384*
- *SHA-512*

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

- *256*
- *384*
- *512*

] **bits** that meet the following: [*FIPS Pub 180-4, "Secure Hash Standard"]*.

Application Note: In accordance with Committee on National Security Systems (CNSS) Policy 15 and the CSfC CP:

- SHA-1 hash is no longer permitted to be used as a hash function,
- SHA-256 is permitted only as part of LMS or XMSS. The hash selection should be consistent with the overall strength of the algorithm used for signature generation.

FCS_COP.1.1/KeyedHash The **application** shall perform [*keyed-hash message authentication*] in accordance with a specified cryptographic algorithm [**selection**:

- ~~*HMAC-SHA-256*~~
- *HMAC-SHA-384*
- *HMAC-SHA-512*

] **with** key sizes [**assignment**: *key size (in bits) used in HMAC*] **and message digest sizes** [**selection**: *256, 384, 512*] bits that meet the following: [*FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code," and FIPS Pub 180-4, "Secure Hash Standard"*].

FCS_COP.1.1/SigGen The **application** shall perform [*cryptographic signature services (generation)*] in accordance with a specified cryptographic algorithm [**selection**:

- *CNSA 2.0 Compliant Algorithm:*
  - ***Module-Lattice-Based Digital Signature Standard*** *using the parameter set ML-DSA-87 that meets the following [FIPS 204, Module-Lattice-Based Digital Signature Standard]*
- *CNSA 1.0 Compliant Algorithms: [**selection**:*
  - ***RSA schemes*** *using cryptographic key sizes of [3072-bit or greater] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 5]*
  - ***ECDSA schemes*** *using ["NIST curves" [**selection**: P-384, ~~P-521~~]] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 6]*

  ]

].

Application Note: As of publication, for CSfC solutions, X.509 v3 certificates used for IPsec and HTTPS/TLS must be signed with ECDSA or RSA. In 2027, where applicable, ML-DSA-87 will be required for new CSfC Components List components. See the MA CP for more details.

FCS_COP.1.1/SigVer The **application** shall perform [*cryptographic signature services (verification)*] in accordance with a specified cryptographic algorithm [**selection**:

- *CNSA 2.0 Compliant Algorithms: [**selection**:*

    - ***Leighton-Micali Signature Algorithm*** *for verification using cryptographic key sizes of [**selection**: 192, 256] bits that meet the following [NIST SP 800-208, "Recommendation for Stateful Hash-Based Signature Schemes"]*
    - ***eXtended Merkle Signature Scheme Algorithm*** *for verification using cryptographic key sizes of [**selection**: 192, 256] bits that meets the following: [NIST SP 800-208, "Recommendation for Stateful Hash-Based Signature Schemes"]*
    - ***Module-Lattice-Based Digital Signature Standard*** *using the parameter set ML-DSA-87 that meets the following [FIPS 204, Module-Lattice-Based Digital Signature Standard]*

  ]
- *CNSA 1.0 Compliant Algorithms: [**selection**:*
    - ***RSA schemes*** *using cryptographic key sizes of [3072-bit or greater] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 5]*
    - ***ECDSA schemes*** *using ["NIST curves" [**selection**: P-384, ~~P-521~~]] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 6]*

  ]

].

Application Note: As of publication, for CSfC solutions, X.509 v3 certificates used for IPsec must be signed with ECDSA or RSA. In 2027, where applicable, ML-DSA-87, LMS and/or XMSS for digitally signing firmware and software and ML-DSA-87 for certificate digital signatures will be required for new CSfC Components List components. See the MA CP for more details.

If RSASSA-PKCS1- v1_5 and/or RSASSA-PSS is selected, 3072 must be selected. For RSASSA-PKCS1- v1_5 or RSASSA-PSS, the TOE can also support 4096, 6144, and 8192 but must support 3072.

FCS_HTTPS_EXT.2.1 The application shall [**selection**: ~~*establish the connection*~~, *not establish the connection, establish or not establish the connection based on an administrative or user setting*] if the peer certificate is deemed invalid when attempting to establish a HTTPS connection.

Application Note: The inclusion of FCS_HTTPS_EXT.2.1 depends upon selection in FCS_HTTPS_EXT.1.1, FTP_DIT_EXT.1.1.

FCS_RBG.1 Random Bit Generation (RBG)

The inclusion of this selection-based component depends upon selection in FCS_RBG_EXT.1.1.

FCS_RBG.1.1 The TSF shall perform deterministic random bit generation services using [**selection**:

- *Hash_DRBG (*[**selection:** *SHA-384, SHA-512*]*)*
- *HMAC_DRBG (*[**selection:** *SHA-384, SHA-512*]*)*
- *CTR_DRBG (AES-CTR-256)*

] in accordance with [NIST SP 800-90A] after initialization with a seed.

FCS_RBG.1.2 The TSF shall use a [**selection**: ~~*TSF noise source [assignment: name of noise source]*~~, *multiple TSF noise sources [assignment: names of noise sources]*, *TSF interface for seeding*] for initialized seeding.

Application Note: The objective of the RBG selections is to ensure the TOE uses a TSF interface for a hardware-based noise source to seed the DRBG.

FCS_RBG.2 Random Bit Generation (External Seeding)

The inclusion of this selection-based component depends upon selection in FCS_RBG.1.2.

FCS_RBG.2.1 The TSF shall be able to accept a minimum input of [**assignment**: *256-bits*] from a TSF interface for the purpose of seeding.

Application Note: If CTR_DRBG is selected in FCS_RBG.1.1 and there is no derivation function, then the minimum bits of entropy must be at least 384.

FCS_RBG.5 Random Bit Generation (Combining Noise Sources)

The inclusion of this selection-based component depends upon selection in FCS_RBG.1.2.

FCS_RBG.5.1 The TSF shall [**assignment**: *combining operation*] [**selection**: *output from TSF noise source(s)*, *input from TSF interface(s) for seeding*] to create the entropy input into the derivation function as defined in [**assignment**: *list of standards*], resulting in a minimum of [**assignment**: *256*] bits of min-entropy.

Functional Package for Transport Layer Security (TLS) Version 2.1 Selections

Table 2: Auditable Events for Selection-based Requirements

Requirement	Auditable Event	Additional Audit Record Contents
FCS_TLSC_EXT.1	[**selection, choose one of**: *Failure to establish a TLS session*, ~~*none*~~]	[**selection, choose one of**: *Reason for failure*, ~~*None*~~]
	[**selection, choose one of**: *Failure to verify presented identifier*, ~~*none*~~]	[**selection, choose one of**: *Presented identifier and reference identifier*, ~~*None*~~]
	[**selection, choose one of**: *Establishment and termination of a TLS session*, ~~*none*~~]	[**selection, choose one of**: *Non-TOE endpoint of connection*, ~~*None*~~]

FCS_TLS_EXT.1.1 The TSF shall implement [**selection**:

- *TLS as a client*
- ~~*TLS as a server*~~
- ~~*DTLS as a client*~~
- ~~*DTLS as a server*~~

].

Application Note: In 2027, when applicable, new CSfC Components List components will be required to support for ML-KEM-1024 and ML-DSA-87 for TLS Servers and Clients. See the MA CP for more details.

FCS_TLSC_EXT.1.1 The TSF shall implement [**selection**: ~~*TLS 1.2 (RFC 5246)*~~, *TLS 1.3 (RFC 8446)*] as a client that supports additional functionality for session renegotiation protection and [**selection**:

- *mutual authentication*
- *supplemental downgrade protection*
- ~~*session resumption*~~
- ~~*no optional functionality*~~

] and shall abort attempts by a server to negotiate any TLS or SSL version prior to supporting only TLS 1.2 (RFC 5246).

Application Note: The objective of this CSfC Selection is to ensure TLS Software Applications in CSfC solutions only use TLS 1.3 and TLS mutual authentication using X.509 certificates is required for TLS connections.

FCS_TLSC_EXT.1.2 The TSF shall be able to support the following [**selection**:

- ~~*TLS 1.2 ciphersuites: [selection:*~~
  - ~~*CNSA 1.0 compliant [selection:*~~
    - ~~*TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 and RFC 8422*~~
    - ~~*TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 and RFC 8422*~~
    - ~~*TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*~~
    - ~~*TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*~~
    - ~~*TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 and RFC 8422*~~
    - ~~*TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 and RFC 8422*~~
    - ~~*ciphersuites using pre-shared secrets: [selection:*~~
      - ~~*TLS_ECDHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 8442*~~
      - ~~*TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487*~~
      - ~~*TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487*~~

~~]~~

~~]~~

- ~~non-CNSA compliant [selection:~~
  - ~~TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246~~
  - ~~TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246~~
  - ~~TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289~~
  - ~~TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289~~
  - ~~TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289~~
  - ~~TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289~~
  - ~~TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246~~
  - ~~TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246~~
  - ~~TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246]~~
  - ~~ciphersuites using pre-shared secrets: [selection:~~
    - ~~TLS_ECDHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 8442~~
    - ~~TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487~~
    - ~~TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487]~~

    ~~]~~

~~]~~

] and no other TLS 1.2 ciphersuites,

- *TLS 1.3 ciphersuites [selection:*
  - *CNSA 2.0 compliant TLS_AES_256_GCM_SHA384 as defined in RFC 8446*
  - ~~non-CNSA compliant [selection:~~
    - ~~TLS_AES_128_GCM_SHA256 as defined in RFC 8446~~
    - ~~[assignment: other TLS 1.3 ciphersuites]~~

  ]

] and no other TLS 1.3 ciphersuites

] offering the supported ciphersuites in a ClientHello message in preference order: [**assignment**: list of supported ciphersuites].

Application Note: Any additional algorithms listed in accordance with TD1025, must conform to Note 1 above to demonstrate compliance with the CSfC selections.

FCS_TLSC_EXT.1.4 The TSF shall be able to support the following TLS ClientHello message extensions:

- signature_algorithms extension (RFC 8446) indicating support for CNSA 1.0 compliant [**selection**:
  - *ecdsa_secp384r1_sha384 (RFC 8446)*
  - *rsa_pkcs1_sha384 (RFC 8446)*

-], and [**selection**:
    - *CNSA 1.0 compliant [selection:*
        - *rsa_pss_pss_sha384 (RFC 8446)*
        - *rsa_pss_rsae_sha384 (RFC 8446)*

      ]
    - ~~*[**assignment**: other non-deprecated, non-CNSA compliant signature algorithms]*~~
    - *no other signature algorithms*

  ] and

[**selection**:

- *signature_algorithms_cert extension (RFC 8446) indicating support for CNSA 1.0 compliant [**selection**:*
    - *ecdsa_secp384r1_sha384 (RFC 8446)*
    - *rsa_pkcs1_sha384 (RFC 8446)*

  *], and [**selection**:*
    - *CNSA 1.0-compliant [**selection**:*
        - *rsa_pss_pss_sha384 (RFC 8446)*
        - *rsa_pss_rsae_sha384 (RFC 8446)*

      ]
    - ~~*non-CNSA compliant [**selection**:*~~
        - ~~*rsa_pkcs1_sha256 (RFC 8446)*~~
        - ~~*rsa_pss_rsae_sha256 (RFC 8446)*~~

      ]
    - ~~*[**assignment**: other non-deprecated, non-CNSA compliant signature algorithms]*~~
    - *no other signature algorithms*

  ] and

- *supported_versions extension (RFC 8446) indicating support for TLS 1.3 and [**selection**: TLS 1.2, no other versions]*
- *supported_groups extension indicating support for [**selection**:*
    - *CNSA 1.0 compliant [**selection**:*
        - *secp384r1 (RFC 8446)*
        - *ffdhe3072 (RFC 7919)*
        - *ffdhe4096 (RFC 7919)*

      ]
    - ~~*non-CNSA compliant [selection:*~~
        - ~~*secp256r1 (RFC 8446)*~~
        - ~~*ffdhe2048 (RFC 7919)*~~

      ]
    - *and [**selection**:*
        - ~~*secp521r1 (RFC 8446)*~~
        - *ffdhe6144(RFC 7919)*
        - *ffdhe8192 (RFC 7919)*
        - *no other supported groups*

      ]

]

- *key_share extension (RFC 8446)*
- *post_handshake_auth (RFC 8446), pre_shared_key (RFC 8446), tls_cert_with_extern_psk (RFC 8773), and psk_key_exchange_modes (RFC 8446) indicating psk_dhe_ke (DHE or ECDHE) mode*
- *extended_master_secret extension (RFC 7627) enforcing server support, and [**selection**: allowing legacy servers, no other enforcement mode]*
- *no other extensions*
-] and shall not send the following extensions:
    - early_data
    - psk_key_exchange_modes indicating PSK only mode.

Application Note:
As of publication, TLS Software Applications must claim and support the applicable ffdhe3072 and/or ECDHE secp384r1 extensions. See FCS_TLS_EXT.1.1 and FCS_CKM.2.1 Application Note for more details.

Any additional algorithms listed in accordance with TD1025, must conform to Note 1 above to demonstrate compliance with the CSfC selections

FCS_TLSC_EXT.2.1 The TSF shall support mutual TLS authentication using X.509v3 certificates during the handshake and [**selection**: ~~in support of post-handshake authentication requests~~, at no other time*], in accordance with [**selection**: RFC 5246, Section 7.4.4, RFC 8446, Section 4.3.2*].

FCS_TLSC_EXT.4.1 The TSF shall support secure TLS renegotiation through use of [**selection**:

- ~~the "renegotiation_info" TLS extension~~
- ~~the TLS_EMPTY_RENEGOTIATION_INFO_SCSV signaling ciphersuite signaling value in accordance with RFC 5746~~
- rejection of all renegotiation attempts

] and shall terminate the session if an unexpected ServerHello is received or [**selection**: *hello request message is received, in no other case*].

Functional Package for X.509 Version 1.0 Selections

FIA_XCU_EXT.1.1 The TSF shall [**selection**: *verify, assert*] identities included in X.509 certificates.

Application Note: The ST author claims "verify" when the TOE associates identities included in X.509 certificates with users, external entities or inter-TOE entities authorized to exercise TOE functionality. The ST author claims "assert" when the TOE represents itself or its functions to internal or external entities.

If "verify" is selected, then FIA_X509_EXT.1 and FIA_X509_EXT.2 must be included. If "assert" is selected, FIA_XCU_EXT.2 must be included.

FIA_ESTC_EXT.1 EST Client Certificate Enrollment

The inclusion of this selection-based component depends upon `if the TOE uses Enrollment over Secure Transport (EST)` **and *selection in FIA_X509_EXT.3.1*.**

FIA_ESTC_EXT.1.4 The TSF shall [**selection**: *invoke platform-provided functionality, provide functionality*] to authenticate its certificate enrollment request to receive [**assignment**: *list of certificates*] from an authorized EST server using [**selection**:

- ~~*HTTP basic authentication transported over TLS (HTTPS) in accordance with RFC 7030 section 3.2.3*~~
- ~~*HTTP digest authentication using a cryptographic hash algorithm transported over TLS (HTTPS) in accordance with RFC 7030 section 3.2.3*~~
- *Certificate-based authentication in accordance with RFC 7030 section 3.3.2 using [**assignment**: pre-existing certificate authorized by the EST server]*

].

Application Note: This SFR with the indicated selection is to be included if EST is implemented by the TOE. EST may be required in CSfC Solutions in the future.

FIA_X509_EXT.1.1 The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to validate certificates in accordance with the following rules:

- Certification path validation meets requirements of RFC 5280 for certificate paths of [**selection**: *unlimited path length, maximum path length of [**assignment**: number greater than or equal to 0] certificates*] and certificate paths exceeding the maximum path length are invalid.
- The current time is within the notBefore and notAfter values of all certificates in the certification path.
- The certification path shall terminate at a trust anchor element appropriate for the supported function.
- Certificates containing subjectUniqueID or issuerUniqueID fields are considered invalid.
- Certificates are signed using cryptographic signatures and hashes in accordance with RFC 8603, and [**selection**:
    - ~~*[assignment: list of supported cryptographic algorithms]*~~
    - *no other algorithms*

  ] and certificates signed using other cryptographic algorithms are considered invalid.
- [**selection**:
    - *CRLs are signed using cryptographic signatures and hashes in accordance with RFC 8603 and [selection:*
        - ~~*[assignment: list of supported cryptographic algorithms]*~~
        - *no other algorithms*

      ] and CRLs signed using other cryptographic algorithms are considered invalid;
    - *OCSP responses are signed using [**selection**:*
        - *sha384WithRSAEncryption with key size of 3072 bits or greater,*
        - *ecdsa-with-SHA384 using [**selection**: secp384r1, ~~secp521r1~~],*
        - *ecdsa-with-SHA512 using [**selection**: secp384r1, ~~secp521r1~~],*

      *] and [**selection**:*

- ~~*[assignment: list of other supported algorithms]*~~
- *no other algorithms*

] requested using the preferredSignatureAlgorithm extension and OCSP responses are considered invalid if using other algorithms;

- ~~*no other algorithm constraints*~~

]

FIA_X509_EXT.1.2 The TSF shall [**selection**: *invoke platform-provided functionality, implement*] processing of the extensions indicated in RFC 5280, section 4.2,

- Authority Key Identifier,
- Subject Key Identifier
- keyUsage

and [**selection**:

- *basicConstraints*
- *authorityInformationAccess* and/or (see Application Note Below)
- *cRLDistributionPoints* (see Application Note Below)
- *certificatePolicies*
- *policyMapping*
- *Subject alternate name containing any of the following name types [**selection**:*
  - *rfc822Name*
  - *dNSName*
  - *directoryName*
  - *uniformResourceIdentifier*
  - *iPAddress*
  - *[**assignment**: other name types]*

  ]
- *extendedKeyUsage*
- *nameConstraints*
- *[**assignment**: other extensions]*
- *no other extensions*

].

Application Note: If the TOE supports an applicable certificate revocation status checking mechanism, the TOE must claim and support the applicable extension authorityInformationAccess and/or cRLDistributionPoints based on the X.509 certificate revocation status checking mechanism used (e.g., CRL Distribution Point (CDP), Online Certificate Status Protocol (OCSP)) to comply with the CSfC selections and CP requirements.

FIA_X509_EXT.1.3 The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to validate revocation status of the certificate using [**selection**:

- *The Online Certificate Status Protocol (*OCSP*) as specified in* RFC *6960*
- *Certificate Revocation Lists (*CRL*) as specified in* RFC *5280 and refined by* RFC *8603*
- ~~*Certificate Revocation Lists as specified in RFC 5280*~~
- ~~*Based on validity period: Certificates expiring within [assignment: time less than 24 hours] of the current time are considered valid when no other valid revocation status information is available*~~
- *Administrative notification of revocation: [**assignment**: administrative action upon notification] using [**assignment**: method to invalidate use of certificates in supported functions] when the certificate is revoked.*
- *Direct association with Certification Authority: [**assignment**: direct revocation status information implementations]*

].

Application Note: CRLs, CDPs with CRLs, or OCSP are all acceptable in CSfC solutions.

FIA_X509_EXT.1.4 The TSF shall [**selection**:

- ~~*not obtain revocation status information by the TSF due to [selection: determining that the certificate expires within [assignment: time less than 24 hours], determining that the [assignment: supported function] validates revocation status using [assignment: methods supported by the function]]*~~
- *[**selection**: invoke platform-provided functionality, implement functionality] to obtain supported revocation status information via [selection:*
  - *Network connection to [**selection**: ~~CA~~, CRL distribution point, OCSP responder, [~~**assignment**: alternate sources~~]]*
  - *Local revocation status information from [**selection**: cached CRL~~, embedded CA repository~~, ~~local OCSP responder~~, ~~administrator configuration~~]*
  - ~~*An OCSP TLS Status Request Extension (OCSP stapling) as specified in RFC 6066,*~~
  - ~~*An OCSP TLS Multiple Certificate Status Request Extension (OCSP multi-stapling) as specified in RFC 6961*~~

  ]

]

Application Note: Any of the following X.509 certificate revocation status checking mechanisms, CRL Distribution Point (CDP) and/or Online Certificate Status Protocol (OCSP) are permitted in CSfC solutions. If the TOE uses a cached CRL, the TOE must attempt to download the latest CRL from a CDP at least once every 24 hours to comply with the CSfC selections and CP requirements. See the MA CP, Campus WLAN, and Key Management Requirements Annex for more details.

FIA_X509_EXT.2.1 The TSF shall [**selection**: *invoke platform-provided functionality to validate, validate*] X.509v3 certificates in accordance with FIA_X509_EXT.1 to support [**assignment**: *supported functions*] using [**selection**:

- *[**selection**: TLS, ~~DTLS~~, IPsec or IKE, SMIME, SSH, [**assignment**: other authenticated communications protocol]]*
- *[**selection**: code signing for system software updates, code signing for software integrity testing, integrity verification for TSF protected data, administrator authentication, user authentication, [**assignment**: other uses]]*

]

FIA_X509_EXT.2.2 For each function indicated in FIA_X509_EXT.2.1, the TSF shall [**selection**: *invoke the TOE platform to determine, determine*] whether the [**selection**: *administrator is allowed to configure certificate acceptance, supported function determines acceptance via [**assignment**: method of determining acceptance], ~~certificate is accepted~~, certificate is not accepted*] when valid certificate revocation status information cannot be obtained from a source indicated in FIA_X509_EXT.1.3.

Application Note: For CSfC solutions, ingress and egress of Certificate Revocation traffic (e.g., OCSP queries, HTTP GET to CDP) on the Black Interface is prohibited. See the MA CP and Campus WLAN CP for more details.

FIA_X509_EXT.3.1 The TSF shall [**selection**: *invoke the TOE platform to generate, generate*] Certificate Requests as specified by [**selection**:

- *RFC 2986 (PKCS-10)*
- *RFC 7030 as updated by RFC 8996 (EST)*
- *RFC 5272 as updated by RFC 6402 (CMC)*
- *RFC 5272 as updated by RFC 8756 (CNSA CMC)*
- *RFC 4210 as updated by RFC 6712 and RFC 9481 (v2 CMP)*
- *RFC 4210 as updated by RFC 6712 and RFC 9480 (v3 CMP)*

] and be able to provide the following information in the request: public key, [**selection**:

- *Subject DN consisting of values for [**selection**:*
  - *U*
  - *O*
  - *OU*
  - *CN*
  - *[**assignment**: other subject attributes]*

]

- *one or more of the following SAN types [**selection**:*
  - *rfc822Name*
  - *dNSName*
  - *directoryName*
  - *uniformResourceIdentifier*
  - *iPAddress*
  - *[**assignment**: other SAN types]*

]

] and [**selection**:

- *[**assignment**: list of other certificate field and extension values]*
- *[**assignment**: list of identifying information]*
- *no other information.*

]

Application Note: The supported certificate request mechanisms are claimed in the first selection. PKCS-10 is claimed whether a manual request process is used or if the PKCS-10 request is posted to the certification authority (for embedded CAs or as in ACME implementations). Other options embed the certificate request in a formalized certificate management protocol. If EST is claimed, FIA_ESTC_EXT.1 is also claimed; if CMC or CNSA CMC are claimed, FIA_CMCC_EXT.1 is also claimed; if v2 or v3 CMP is claimed, FIA_CMPC_EXT.1 is also claimed.

Application Note: PKCS-10 must be selected and supported. RFC 7030 as updated by RFC 8996 (EST) and RFC 5272 as updated by RFC 8756 (CNSA CMC) may also be selected and supported.

FIA_XCU_EXT.2.1 The TSF shall [**selection**:

- *request certificates from an [**selection**: external, embedded]* CA,
- *obtain certificates from an embedded CA*

] to represent [**assignment**: *TOE functions*] for [**selection**:

- *[**selection**: TLS, ~~DTLS~~, IPsec or IKE, SMIME, SSH, [**assignment**: other authenticated communications protocol]]*
- *[**selection**: code signing for system software updates, code signing for software integrity testing, integrity verification for TSF protected data, administrator authentication, user authentication, [**assignment**: other uses]]*

].