Functional Package for Transport Layer Security (TLS)

Version: 1.1
2019-03-01
National Information Assurance Partnership

Revision History

Version	Date	Comment
1.0	2018-12-17	First publication
1.1	2019-03-01	Clarifications regarding override for invalid certificates, renegotation_info extension, DTLS versions, and named Diffie-Hellman groups in DTLS contexts

Contents

Appendix A - Optional Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE) are contained in the body of this PP. This appendix contains three other types of optional requirements that may be included in the ST, but are not required in order to conform to this PP. However, applied modules, packages and/or use cases may refine specific requirements as mandatory.

The first type (Appendix B - Strictly Optional Requirements) are strictly optional requirements that are independent of the TOE implementing any function. If the TOE fulfills any of these requirements or supports a certain functionality, the vendor is encouraged to include the SFRs in the ST, but are not required in order to conform to this PP.

The second type (Appendix C - Objective Requirements) are objective requirements that describe security functionality not yet widely available in commercial technology. The requirements are not currently mandated in the body of this PP, but will be included in the baseline requirements in future versions of this PP. Adoption by vendors is encouraged and expected as soon as possible.

The third type (Appendix D - Implementation-Based Requirements) are dependent on the TOE implementing a particular function. If the TOE fulfills any of these requirements, the vendor must either add the related SFR or disable the functionality for the evaluated configuration.

Appendix B - Strictly Optional Requirements

This PP does not define any Strictly Optional requirements.

Appendix C - Objective Requirements

C.1 Cryptographic Support (FCS)

TLS Client Support for Signature Algorithms Extension

.1

The product shall present the signature_algorithms extension in the Client Hello with the supported_signature_algorithms value containing the following hash algorithms: [**selection**: *SHA256*, *SHA384*, *SHA512*] and no other hash algorithms.

Application Note: This requirement limits the hashing algorithms supported for the purpose of digital signature verification by the client and limits the server to the supported hashes for the purpose of digital signature generation by the server. The signature_algorithms extension is only supported by TLS 1.2.

Evaluation Activities

The evaluator shall verify that TSS describes the signature_algorithm extension and whether the required behavior is performed by default or may be configured. If the TSS indicates that the signature_algorithm extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the signature_algorithm extension.
The evaluator shall also perform the following tests:

- ***Test 1:*** *The evaluator shall configure the server to send a certificate in the TLS connection that is not supported according to the Client's HashAlgorithm enumeration within the signature_algorithms extension (for example, send a certificate with a SHA-1 signature). The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.*
- ***Test 2:*** *[conditional] If the client supports a DHE or ECDHE cipher suite, the evaluator shall configure the server to send a Key Exchange handshake message including a signature not supported according to the client's HashAlgorithm enumeration (for example, the server signed the Key Exchange parameters using a SHA-1 signature). The evaluator shall verify that the product disconnects after receiving the server's Key Exchange handshake message.*

TSS
Guidance
Tests

TLS Server Support for Signature Algorithms Extension

.1

The product shall present the HashAlgorithm enumeration in

supported_signature_algorithms in the Certificate Request with the following hash algorithms: [**selection**: *SHA256*, *SHA384*, *SHA512*] and no other hash algorithms.

Application Note: This requirement limits the hashing algorithms supported for the purpose of digital signature verification by the server and limits the client to the supported hashes for the purpose of digital signature generation by the client. The supported_signature_algorithms is only supported by TLS 1.2.

Evaluation Activities

The evaluator shall verify that TSS describes the supported_signature_algorithms field of the Certificate Request and whether the required behavior is performed by default or may be configured. If the TSS indicates that the supported_signature_algorithms field must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the supported_signature_algorithms field. The evaluator shall also perform the following test:
The evaluator shall configure the server to send the signature_algorithms extension in the Certificate Request message indicating that the hash algorithm used by the client's certificate is not supported. The evaluator shall attempt a connection using that client certificate and verify that the server denies the client's connection.
TSS
Guidance
Tests

Appendix D - Implementation-Based Requirements

This PP does not define any Implementation-Based requirements.

Appendix E - Selection-Based Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. There are additional requirements based on selections in the body of the PP: if certain selections are made, then additional requirements below must be included.

Appendix F - Cryptographic Support (FCS)

TLS Client Protocol

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

.1

The product shall implement TLS 1.2 (RFC 5246) and [**selection**: *TLS 1.1 (RFC 4346)*, *no earlier TLS versions*] as a client that supports the cipher suites [**selection**:

- *TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*

] and also supports functionality for [**selection**:

- *mutual authentication*,
- *session renegotiation*,
- *none*

].

Application Note: The ST author should select the cipher suites that are supported, and must select at least one cipher suite. The cipher suites to be tested in the evaluated configuration are limited by this requirement. However, this requirement does not restrict the TOE's ability to propose additional cipher suites beyond the ones listed in this requirement in its Client Hello message. That is, the TOE may propose any cipher suite but the evaluation will only test cipher suites from the above list. It is necessary to limit the cipher suites that can be used in an evaluated configuration administratively on the server in the test environment. GCM cipher suites are preferred over CBC cipher suites, ECDHE preferred over RSA and DHE, and SHA256 or SHA384 over SHA.

TLS_RSA_WITH_AES_128_CBC_SHA is not required despite being mandated by RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

If any ECDHE or DHE cipher suites are selected, then is required.

If *mutual authentication* is selected, then the ST must additionally include the requirements from . If the TOE implements mutual authentication, this selection must be made.

If *session renegotiation* is selected, then the ST must additionally include the requirements from . If the TOE implements session renegotiation, this selection must be made.

.2

The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

Application Note: The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the user (e.g. entering a URL into a web browser or clicking a link), by configuration (e.g. configuring the name of a mail server or authentication server), or by an application (e.g. a parameter of an API) depending on the product service. Based on a singular reference identifier's source domain and application service type (e.g. HTTP, SIP, LDAP), the client establishes all reference identifiers which are acceptable, such as a Common Name for the Subject Name field of the certificate and a (case-insensitive) DNS name, URI name, and Service Name for the Subject Alternative Name field. The client then compares this list of all acceptable reference identifiers to the presented identifiers in the TLS server's certificate. The preferred method for verification is the Subject Alternative Name using DNS names, URI names, or Service Names. Verification using the Common Name for the purposes of backwards compatibility is optional. Additionally, support for use of IP addresses in the Subject Name or Subject Alternative name is discouraged, as against best practices, but may be implemented. Finally, the client should avoid constructing reference identifiers using wildcards. However, if the presented identifiers include wildcards, the client must follow the best practices regarding matching; these best practices are captured in the evaluation activity.

.3

The product shall not establish a trusted channel if the server certificate is invalid [**selection**:

- *with no exceptions*,
- *except when override is authorized*

].

Application Note: Validity is determined by the identifier verification, certificate path, the expiration date, and the revocation status in accordance with RFC 5280. Certificate validity shall be tested in accordance with testing performed for as defined in any PP or PP-Module which instantiates this Package.

The selection that permits override for invalid certificates should be interpreted as follows:

- explicit administrator or user action is needed to authorize the override, on a per-certificate basis
- override may be sought or granted at any time, though this typically occurs when an invalid certificate is presented during connection setup
- override decisions may be stored and then consulted later, to permit connections using these otherwise-invalid certificates to establish trusted channels without user or administrator action

As indicated in , note that a PP author may instantiate this SFR using only the first selection, preventing the ability to allow overrides.

Evaluation Activities

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall check the TSS to ensure that the cipher suites specified include those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the product so that TLS conforms to the description in the TSS. The evaluator shall also perform the following tests:

- ***Test 1:*** *The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).*
- ***Test 2:*** *The goal of the following test is to verify that the TOE accepts only certificates with appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation.*

  The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension and verify that a connection is established. The evaluator shall repeat this test

using a different, but otherwise valid and trusted, certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension and ensure that a connection is not established. Ideally, the two certificates should be similar in structure, the types of identifiers used, and the chain of trust.

- ***Test 3:*** *The evaluator shall send a server certificate in the TLS connection that does not match the server-selected cipher suite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite or send a RSA certificate while using one of the ECDSA cipher suites.) The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.*
- ***Test 4:*** *The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the client denies the connection.*
- ***Test 5:*** *The evaluator shall perform the following modifications to the traffic:*
  - ***Test 5.1:*** *Change the TLS version selected by the server in the Server Hello to an undefined TLS version (for example 1.5 represented by the two bytes 03 06) and verify that the client rejects the connection.*
  - ***Test 5.2:*** *Change the TLS version selected by the server in the Server Hello to the most recent unsupported TLS version (for example 1.1 represented by the two bytes 03 02) and verify that the client rejects the connection.*
  - ***Test 5.3:*** *[conditional] If DHE or ECDHE cipher suites are supported, modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client does not complete the handshake and no application data flows.*
  - ***Test 5.4:*** *Modify the server's selected cipher suite in the Server Hello handshake message to be a cipher suite not presented in the Client Hello handshake message. The evaluator shall verify that the client does not complete the handshake and no application data flows.*
  - ***Test 5.5:*** *[conditional] If DHE or ECDHE cipher suites are supported, modify the signature block in the server's Key Exchange handshake message, and verify that the client does not complete the handshake and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.*
  - ***Test 5.6:*** *Modify a byte in the Server Finished handshake message, and verify that the client does not complete the handshake and no application data flows.*
  - ***Test 5.7:*** *Send a message consisting of random bytes from the server after the server has issued the Change Cipher Spec message and verify that the client does not complete the handshake and no application data flows. The message must still have a valid 5-byte record header in order to ensure the message will be parsed as TLS.*

The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the application-configured reference identifier, including which types of reference identifiers are supported (e.g. Common Name, DNS Name, URI Name, Service Name, or other application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported. The evaluator shall ensure that this description identifies whether and the manner in which certificate pinning is supported or used by the product.
The evaluator shall verify that the AGD guidance includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS.
The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection:

- ***Test 1:*** *The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails.*

  Note that some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.
- ***Test 2:*** *The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type.*
- ***Test 3:*** *[conditional] If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.*
- ***Test 4:*** *The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds.*
- ***Test 5:*** *The evaluator shall perform the following wildcard tests with each supported type of reference identifier. The support for wildcards is intended to be optional. If wildcards are supported, the first, second, and third tests below shall be executed. If wildcards are not supported, then the fourth test below shall be executed.*
  - ***Test 5.1:*** *[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.*

- ***Test 5.2:*** *[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.come) and verify that the connection fails.*
- ***Test 5.3:*** *[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.*
- ***Test 5.4:*** *[conditional]: If wildcards are not supported, the evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection fails.*

- ***Test 6:*** *[conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.*
- ***Test 7:*** *[conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.*

If the selection for authorizing override of invalid certificates is made, then the evaluator shall ensure that the TSS includes a description of how and when user or administrator authorization is obtained. The evaluator shall also ensure that the TSS describes any mechanism for storing such authorizations, such that future presentation of such otherwise-invalid certificates permits establishment of a trusted channel without user or administrator action. The evaluator shall demonstrate that using an invalid certificate (unless excepted) results in the function failing as follows, unless excepted:

- ***Test 1:*** *The evaluator shall demonstrate that a server using a certificate without a valid certification path results in an authentication failure. Using the administrative guidance, the evaluator shall then load the trusted CA certificate(s) needed to validate the server's certificate, and demonstrate that the connection succeeds. The evaluator then shall delete one of the CA certificates, and show that the connection fails.*
- ***Test 2:*** *The evaluator shall demonstrate that a server using a certificate which has been revoked results in an authentication failure.*
- ***Test 3:*** *The evaluator shall demonstrate that a server using a certificate which has passed its expiration date results in an authentication failure.*
- ***Test 4:*** *The evaluator shall demonstrate that a server using a certificate which does not have a valid identifier results in an authentication failure.*

TSS
Guidance
Tests

TLS Client Support for Mutual Authentication

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

.1

The product shall support mutual authentication using X.509v3 certificates.

Application Note: The use of X.509v3 certificates for TLS is addressed in . This requirement adds that a client must be capable of presenting a certificate to a TLS server for TLS mutual authentication. Presenting a certificate is not mandatory in all circumstances: it may depend on the configuration of the client or other factors.

Evaluation Activities

The evaluator shall ensure that the TSS description required per includes the use of client-side certificates for TLS mutual authentication. The evaluator shall also ensure that the TSS describes any factors beyond configuration that are necessary in order for the client to engage in mutual authentication using X.509v3 certificates.
The evaluator shall ensure that the AGD guidance includes any instructions necessary to configure the TOE to perform mutual authentication. The evaluator also shall verify that the AGD

guidance required per includes instructions for configuring the client-side certificates for TLS mutual authentication.
The evaluator shall also perform the following tests:

- ***Test 1:*** *The evaluator shall establish a connection to a server that is not configured for mutual authentication (i.e. does not send Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE did not send Client's Certificate message (type 11) during handshake.*
- ***Test 2:*** *The evaluator shall establish a connection to a server with a shared trusted root that is configured for mutual authentication (i.e. it sends Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE responds with a non-empty Client's Certificate message (type 11) and Certificate Verify (type 15) message.*

TSS
Guidance
Tests

TLS Client Support for Renegotiation

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

.1

The product shall support secure renegotiation through use of the "renegotiation_info" TLS extension in accordance with RFC 5746.

Application Note: RFC 5746 defines an extension to TLS that binds renegotiation handshakes to the cryptography in the original handshake.

Per RFC 5746, the client may present either the "renegotiation_info" extension or the signaling cipher suite value TLS_EMPTY_RENEGOTIATION_INFO_SCSV in the initial ClientHello message to indicate support for renegotiation. (A signaling cipher suite value (SCSV) is presented as a cipher suite, but its only purpose is to provide other information and not to advertise support for a cipher suite.) The TLS_EMPTY_RENEGOTIATION_INFO_SCSV signaling cipher suite value exists as an alternative to presenting the "renegotation_info" extension so that TLS server implementations that immediately terminate the connection when they encounter any extension they do not understand can still proceed with a connection. The client may still choose to reject the connection later, if it insists upon renegotiation support and the server does not support it. In any case, RFC 5746 states that during any renegotiation the "renegotiation_info" extension must be presented by the peer initiating renegotiation, and so the client must support use of this extension.

Evaluation Activities

The evaluator shall perform the following tests:

- ***Test 1:*** *The evaluator shall use a network packet analyzer/sniffer to capture the traffic between the two TLS endpoints. The evaluator shall verify that either the "renegotiation_info" field or the SCSV cipher suite is included in the ClientHello message during the initial handshake.*
- ***Test 2:*** *The evaluator shall verify the Client's handling of ServerHello messages received during the initial handshake that include the "renegotiation_info" extension. The evaluator shall modify the length portion of this field in the ServerHello message to be non-zero and verify that the client sends a failure and terminates the connection. The evaluator shall verify that a properly formatted field results in a successful TLS connection.*
- ***Test 3:*** *The evaluator shall verify that ServerHello messages received during secure renegotiation contain the "renegotiation_info" extension. The evaluator shall modify either the "client_verify_data" or "server_verify_data" value and verify that the client terminates the connection.*

Tests

TLS Client Support for Supported Groups Extension

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

.1

The product shall present the Supported Groups Extension in the Client Hello with the supported groups [**selection**:

- *secp256r1,*

- *secp384r1*,
- *secp521r1*,
- *ffdhe2048(256)*,
- *ffdhe3072(257)*,
- *ffdhe4096(258)*,
- *ffdhe6144(259)*,
- *ffdhe8192(260)*

].

Application Note: If an elliptic curve or Diffie-Hellman ciphersuite is selected in or , then shall be included in the ST. This requirement does not limit the elliptic curves the client may propose for authentication and key agreement. The Supported Groups Extension was previously referred to as the Supported Elliptic Curves Extension and is described in RFC 7919.

Evaluation Activities

The evaluator shall verify that TSS describes the Supported Groups Extension. The evaluator shall also perform the following test:

- ***Test 1:*** *The evaluator shall configure a server to perform key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.*

TSS
Tests

TLS Server Protocol

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

.1

The product shall implement TLS 1.2 (RFC 5246) and [**selection**: *TLS 1.1 (RFC 4346), no earlier TLS versions*] as a server that supports the cipher suites [**selection**:

- *TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*

] and no other cipher suites, and also supports functionality for [**selection**:

- *mutual authentication*,
- *session renegotiation*,
- *none*

].

Application Note: The ST author should select the cipher suites that are supported, and must select at least one cipher suite. It is necessary to limit the cipher suites that can be used in an evaluated configuration administratively on the server in the test environment. If administrative steps need to be taken so that the cipher suites negotiated by the implementation are limited to those in this requirement, then the appropriate instructions need to be contained in the guidance. GCM cipher suites are preferred over CBC cipher suites, ECDHE preferred over RSA and DHE, and SHA256 or SHA384 over SHA.

TLS_RSA_WITH_AES_128_CBC_SHA is not required despite being mandated by RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

If *mutual authentication* is selected, then the ST must additionally include the requirements from . If the TOE implements mutual authentication, this selection must be made.

If *session renegotiation* is selected, then the ST must additionally include the requirements from . If the TOE implements session renegotiation, this selection must be made.

.2

The product shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [**selection**: *TLS 1.1*, *none*].

Application Note: All SSL versions are denied. Any TLS version not selected in should be selected here.

.3

The product shall perform key establishment for TLS using [**selection**:

- *RSA with size [**selection**: 2048 bits, 3072 bits, 4096 bits, no other sizes]* ,
- *Diffie-Hellman parameters with size [**selection**: 2048 bits, 3072 bits, 4096 bits, 6144 bits, 8192 bits, no other sizes]* ,
- *Diffie-Hellman groups [**selection**: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, no other groups]* ,
- *ECDHE parameters using elliptic curves [**selection**: secp256r1, secp384r1, secp521r1] and no other curves* ,
- *no other key establishment methods*

].

Application Note: If the ST lists an RSA cipher suite in , the ST must include the RSA selection in the requirement.

If the ST lists a DHE cipher suite in , the ST must include either the Diffie-Hellman selection for parameters of a certain size, or for particular Diffie-Hellman groups. The selection for "Diffie-Hellman parameters" refers to the method defined by RFC 5246 (and RFC 4346) Section 7.4.3 where the server provides Diffie-Hellman parameters to the client. The Supported Groups extension defined in RFC 7919 identifies particular Diffie-Hellman groups, which are listed in the following selection. Regarding this distinction, it is acceptable to use Diffie-Hellman group 14 with TLS (there is currently no ability to negotiate group 14 using the Supported Groups extension, but it could be used with the "Diffie-Hellman parameters" selection). As in RFC 7919, the terms "DHE" and "FFDHE" are both used to refer to the finite-field-based Diffie-Hellman ephemeral key exchange mechanism, distinct from elliptic-curve-based Diffie Hellman ephemeral key exchange (ECDHE).

If the ST lists an ECDHE cipher suite in , the ST must include the selection for ECDHE using elliptic curves in the requirement.

Evaluation Activities

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall check the TSS to ensure that the cipher suites specified include those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS. The evaluator shall also perform the following tests:

- ***Test 1:** The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).*
- ***Test 2:** The evaluator shall send a Client Hello to the server with a list of cipher suites that does not contain any of the cipher suites in the server's ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the server denies the connection.*
- ***Test 3:** If RSA key exchange is used in one of the selected ciphersuites, the evaluator shall use a client to send a properly constructed Key Exchange message with a modified*

EncryptedPreMasterSecret field during the TLS handshake. The evaluator shall verify that the handshake is not completed successfully and no application data flows.

- ***Test 4:*** *The evaluator shall perform the following modifications to the traffic:*
  - ***Test 4.1:*** *Change the TLS version proposed by the client in the Client Hello to a non-supported TLS version (for example 1.3 represented by the two bytes 03 04) and verify that the server rejects the connection.*
  - ***Test 4.2:*** *Modify a byte in the data of the client's Finished handshake message, and verify that the server rejects the connection and does not send any application data.*
  - ***Test 4.3:*** *Demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption): Generate a Fatal Alert by sending a Finished message from the client before the client sends a ChangeCipherSpec message, and then send a Client Hello with the session identifier from the previous incomplete session, and verify that the server does not resume the session.*
  - ***Test 4.4:*** *Send a message consisting of random bytes from the client after the client has issued the ChangeCipherSpec message and verify that the server denies the connection.*

The evaluator shall verify that the TSS contains a description of the denial of old SSL and TLS versions consistent relative to selections in . The evaluator shall verify that the AGD guidance includes any configuration necessary to meet this requirement.

- ***Test 1:*** *The evaluator shall send a Client Hello requesting a connection with version SSL 2.0 and verify that the server denies the connection. The evaluator shall repeat this test with SSL 3.0 and TLS 1.0, and TLS 1.1 if it is selected.*

The evaluator shall verify that the TSS describes the key agreement parameters of the server's Key Exchange message. The evaluator shall verify that any configuration guidance necessary to meet the requirement must be contained in the AGD guidance.

The evaluator shall conduct the following tests. The testing can be carried out manually with a packet analyzer or with an automated framework that similarly captures such empirical evidence. Note that this testing can be accomplished in conjunction with other testing activities. For each of the following tests, determining that the size matches the expected size is sufficient.

- ***Test 1:*** *[conditional] If RSA-based key establishment is selected, the evaluator shall configure the TOE with a certificate containing a supported RSA size and attempt a connection. The evaluator shall verify that the size used matches that which is configured and that the connection is successfully established. The evaluator shall repeat this test for each supported size of RSA-based key establishment.*
- ***Test 2:*** *[conditional] If finite-field (i.e. non-EC) Diffie-Hellman ciphers are selected, the evaluator shall attempt a connection using a Diffie-Hellman key exchange with a supported parameter size or supported group. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported parameter size or group.*
- ***Test 3:*** *[conditional] If ECDHE ciphers are selected, the evaluator shall attempt a connection using an ECDHE ciphersuite with a supported curve. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported elliptic curve.*

TSS
Guidance
Tests

TLS Server Support for Mutual Authentication

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

.1

The product shall support authentication of TLS clients using X.509v3 certificates.

.2

The product shall not establish a trusted channel if the client certificate is invalid.

Application Note: The use of X.509v3 certificates for TLS is addressed in This requirement adds that this use must include support for client-side certificates for TLS mutual authentication. Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280. Certificate validity shall be tested in accordance with testing performed for .

.3

The product shall not establish a trusted channel if the Distinguished Name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match one of the expected identifiers for the client.

Application Note: The client identifier may be in the Subject field or the Subject Alternative Name extension of the certificate. The expected identifier may either be configured, may be compared to the domain name, IP address, username, or email address used by the client, or may be passed to a directory server for comparison. In the latter case, the matching itself may be performed outside the TOE.

Evaluation Activities

The evaluator shall ensure that the TSS description required per includes the use of client-side certificates for TLS mutual authentication. The evaluator shall verify that the AGD guidance required per includes instructions for configuring the client-side certificates for TLS mutual authentication. The evaluator shall ensure that the AGD guidance includes instructions for configuring the server to require mutual authentication of clients using these certificates. The evaluator shall use TLS as a function to verify that the validation rules in are adhered to and shall perform the following tests. The evaluator shall apply the AGD guidance to configure the server to require TLS mutual authentication of clients for the following tests, unless overridden by instructions in the test activity:

- ***Test 1:*** *The evaluator shall configure the server to send a certificate request to the client. The client shall send a certificate_list structure which has a length of zero. The evaluator shall verify that the handshake is not finished successfully and no application data flows.*
- ***Test 2:*** *The evaluator shall configure the server to send a certificate request to the client. The client shall send no client certificate message, and instead send a client key exchange message in an attempt to continue the handshake. The evaluator shall verify that the handshake is not finished successfully and no application data flows.*
- ***Test 3:*** *The evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the handshake is not finished successfully and no application data flows.*
- ***Test 4:*** *The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.*
- ***Test 5:*** *The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not in fact correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not in fact terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.*
- ***Test 6:*** *The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.*
- ***Test 7:*** *The evaluator shall perform the following modifications to the traffic: a) Configure the server to require mutual authentication and then modify a byte in the client's certificate. The evaluator shall verify that the server rejects the connection. b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message. The evaluator shall verify that the server rejects the connection.*

If the product implements mutual authentication, the evaluator shall verify that the TSS describes how the DN and SAN in the certificate is compared to the expected identifier. If the DN is not compared automatically to the domain name, IP address, username, or email address, the evaluator shall ensure that the AGD guidance includes configuration of the expected identifier or the directory server for the connection.

- ***Test 1:*** *The evaluator shall send a client certificate with an identifier that does not match any of the expected identifiers and verify that the server denies the connection. The matching itself might be performed outside the TOE (e.g. when passing the certificate on to a directory server for comparison).*

TSS
Guidance
Tests

TLS Server Support for Renegotiation

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

.1

The product shall support the "renegotiation_info" TLS extension in accordance with RFC 5746.

.2

The product shall include the renegotiation_info extension in ServerHello messages.

Application Note: RFC 5746 defines an extension to TLS that binds renegotiation handshakes to the cryptography in the original handshake.

Evaluation Activities

The following tests require connection with a client that supports secure renegotiation and the "renegotiation_info" extension.

- ***Test 1:*** *The evaluator shall use a network packet analyzer/sniffer to capture the traffic between the two TLS endpoints. The evaluator shall verify that the “renegotiation_info” field is included in the ServerHello message.*
- ***Test 2:*** *The evaluator shall modify the length portion of the field in the ClientHello message in the initial handshake to be non-zero and verify that the server sends a failure and terminates the connection. The evaluator shall verify that a properly formatted field results in a successful TLS connection.*
- ***Test 3:*** *The evaluator shall modify the "client_verify_data" or "server_verify_data" value in the ClientHello message received during secure renegotiation and verify that the server terminates the connection.*

Tests

DTLS Client Protocol

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

.1

The product shall implement DTLS 1.2 (RFC 6347) and [**selection**: *DTLS 1.0 (RFC 4347), no earlier DTLS versions*] as a client that supports the cipher suites [**selection**:

- *TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,*
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,*
- *TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,*
- *TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,*
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,*
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,*
- *TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,*
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,*
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*

] and also supports functionality for [**selection**:

- *mutual authentication,*
- *none*

].

Application Note: If any ECDHE or DHE cipher suites are selected, then is required.

If *mutual authentication* is selected, then the ST must additionally include the requirements from . If the TOE implements mutual authentication, this selection must be made.

Differences between DTLS 1.2 and TLS 1.2 are outlined in RFC 6347; otherwise the protocols are the same. All application notes listed for that are relevant to DTLS apply to this requirement.

.2

The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

.3

The product shall not establish a trusted channel if the server certificate is invalid [**selection**: *with no exceptions*, *except when override is authorized*].

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

.4

The product shall [**selection**: *terminate the DTLS session*, *silently discard the record*] if a message received contains an invalid MAC or if decryption fails in the case of GCM and other AEAD ciphersuites.

Evaluation Activities

The evaluator shall perform the evaluation activities listed for , but ensuring that DTLS (and not TLS) is used in each evaluation activity.

For tests which involve version numbers, note that in DTLS the on-the-wire representation is the 1's complement of the corresponding textual DTLS version numbers. This is described in Section 4.1 of RFC 6347 and RFC 4347. For example, DTLS 1.0 is represented by the bytes 0xfe 0xff, while the undefined DTLS 1.4 would be represented by the bytes 0xfe 0xfb. The evaluator shall perform the evaluation activities listed for . The evaluator shall perform the evaluation activities listed for . The evaluator shall verify that the TSS describes the actions that take place if a message received from the DTLS Server fails the MAC integrity check. The evaluator shall establish a connection using a server. The evaluator will then modify at least one byte in a record message, and verify that the client discards the record or terminates the DTLS session.
TSS
Tests

DTLS Client Support for Mutual Authentication

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

.1

The product shall support mutual authentication using X.509v3 certificates.

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

Evaluation Activities

The evaluator shall perform the evaluation activities listed for .
Tests

DTLS Server Protocol

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

.1

The product shall implement DTLS 1.2 (RFC 6347) and [**selection**: *DTLS 1.0 (RFC 4347)*, *no earlier DTLS versions*] as a server that supports the ciphersuites [**selection**:

- *TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*,
- *TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC*

5289,
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*,
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*

] and no other cipher suites, and also supports functionality for [**selection**:

- *mutual authentication*,
- *none*

].

Application Note: If *mutual authentication* is selected, then the ST must additionally include the requirements from . If the TOE implements mutual authentication, this selection must be made.

All application notes listed for that are relevant to DTLS apply to this requirement.

.2

The product shall deny connections from clients requesting [**assignment**: *list of DTLS protocol versions*].

Application Note: Any specific DTLS version not selected in should be assigned here. This version of the PP does not require the server to deny DTLS 1.0, and if the TOE supports DTLS 1.0 then "none" can be assigned. In a future version of this PP, DTLS 1.0 will be required to be denied.

.3

The product shall not proceed with a connection handshake attempt if the DTLS Client fails validation.

Application Note: The process to validate the IP address of a DTLS client is specified in section 4.2.1 of RFC 6347 (DTLS 1.2) and RFC 4347 (DTLS 1.0). The server validates the DTLS client during Connection Establishment (Handshaking) and prior to sending a Server Hello message. After receiving a ClientHello, the DTLS Server sends a HelloVerifyRequest along with a cookie. The cookie is a signed message using a keyed hash function. The DTLS Client then sends another ClientHello with the cookie attached. If the DTLS server successfully verifies the signed cookie, the Client is not using a spoofed IP address.

.4

The product shall perform key establishment for DTLS using [**selection**:

- *RSA with size [**selection**: 2048 bits, 3072 bits, 4096 bits, no other sizes]* ,
- *Diffie-Hellman parameters with size [**selection**: 2048 bits, 3072 bits, 4096 bits, 6144 bits, 8192 bits, no other size]* ,
- *Diffie-Hellman groups [**selection**: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, no other groups]* ,
- *ECDHE parameters using elliptic curves [**selection**: secp256r1, secp384r1, secp521r1] and no other curves*,
- *no other key establishment methods*

].

Application Note: If the ST lists an RSA cipher suite in , the ST must include the RSA selection in the requirement.
If the ST lists a DHE cipher suite in , the ST must include either the Diffie-Hellman selection for parameters of a certain size, or for particular Diffie-Hellman groups.
If the ST lists an ECDHE cipher suite in , the ST must include the NIST curves selection in the requirement.

.5

The product shall [**selection**: *terminate the DTLS session*, *silently discard the record*] if a message received contains an invalid MAC or if decryption fails in the case of GCM and other AEAD ciphersuites.

Evaluation Activities

The evaluator shall perform the evaluation activities listed for , but ensuring that DTLS (and not TLS) is used in each stage of the evaluation activities.

For tests which involve version numbers, note that in DTLS the on-the-wire representation is the 1's complement of the corresponding textual DTLS version numbers. This is described in Section 4.1 of RFC 6347 and RFC 4347. For example, DTLS 1.0 is represented by the bytes 0xfe 0xff, while the undefined DTLS 1.4 would be represented by the bytes 0xfe 0xfb. The following evaluation activities shall be conducted unless "none" is assigned.

The evaluator shall verify that the TSS contains a description of the denial of old DTLS versions consistent relative to selections in . The evaluator shall verify that the AGD guidance includes any configuration necessary to meet this requirement.

- ***Test 1:*** *The evaluator shall send a Client Hello requesting a connection with each version of DTLS specified in the selection and verify that the server denies the connection.*

The evaluator shall verify that the TSS describes how the DTLS Client IP address is validated prior to issuing a ServerHello message. Modify at least one byte in the cookie from the Server's HelloVerifyRequest message, and verify that the Server rejects the Client's handshake message. The evaluator shall perform the evaluation activities listed for . The evaluator shall verify that the TSS describes the actions that take place if a message received from the DTLS client fails the MAC integrity check. The evaluator shall establish a connection using a client. The evaluator will then modify at least one byte in a record message, and verify that the server discards the record or terminates the DTLS session.
TSS
Guidance
Tests

DTLS Server Support for Mutual Authentication

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

.1

The product shall support mutual authentication of DTLS clients using X.509v3 certificates.

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

.2

The product shall not establish a trusted channel if the client certificate is invalid.

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

.3

The product shall not establish a trusted channel if the Distinguished Name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match one of the expected identifiers for the client.

Application Note: All application notes listed for that are relevant to DTLS apply to this requirement.

Evaluation Activities

The evaluator shall perform the evaluation activities listed for . The evaluator shall perform the evaluation activities listed for . The evaluator shall perform the evaluation activities listed for .
Tests

Appendix G - Acronyms

AES Advanced Encryption Standard CA Certificate Authority CBC Cipher Block Chaining CN Common Name DHE Diffie-Hellman Ephemeral DN Distinguished Name DNS Domain Name Server DTLS Datagram Transport Layer Security EAP Extensible Authentication Protocol ECDHE Elliptic Curve Diffie-Hellman Ephemeral ECDSA Elliptic Curve Digital Signature Algorithm GCM Galois/Counter Mode HTTP Hypertext Transfer Protocol IETF Internet Engineering Task Force IP Internet Protocol LDAP Lightweight Directory Access Protocol NIST National Institute of Standards and Technology RFC Request for Comment (IETF) RSA Rivest Shamir Adelman SAN Subject Alternative Name SCSV Signaling Cipher Suite Value SHA Secure Hash Algorithm SIP Session Initiation Protocol TCP Transmission Control Protocol TLS Transport Layer Security UDP User Datagram Protocol URI Uniform Resource Identifier URL Uniform Resource Locator

Appendix H - Acronyms

Acronym	Meaning
Base-PP	Base Protection Profile
CC	Common Criteria
CEM	Common Evaluation Methodology
OE	Operational Environment
PP	Protection Profile
PP-Configuration	Protection Profile Configuration
PP-Module	Protection Profile Module
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
TSS	TOE Summary Specification

Appendix I - Bibliography

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.
[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.