# International Standard

**ISO/IEC 20648**

# Information technology — TLS specification for storage systems

*Technologies de l'information — Spécification TLS pour systèmes de stockage*

Second edition
2024-07

# Contents

# Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by SNIA (as TLS Specification for Storage Systems, Version 2.1) and drafted in accordance with its editorial rules. It was adopted, under the JTC 1 PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

This second edition cancels and replaces the first edition (ISO/IEC 20648:2016), which has been technically revised.

The main changes are as follows:

— a statement has been added regarding the relevance of ISO/IEC 20648 to Datagram Transport Layer Security (DTLS) implementations;

— a statement has been added regarding quantum computing and TLS;

— a statement has been added encouraging the use of TLS 1.3;

— a recommendation has been added to guard against replay attacks on zero round-trip time (0-RTT) for TLS version 1.3;

— the recommended cipher suites have been aligned with the RFC 7525 recommendations on forward secrecy;

— the requirements concerning 112 bits of security strength have been changed to 128 bits of security strength;

— the requirements for the maximum certificate validity period have been changed from 3 years to 398 days;

— the requirements associated with the ECDSA signature certificate have been clarified;

— a requirement has been added for including the TLS 1.3 extension for pre-shared key (PSK) support.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

# Introduction

Within information and communications technology, one of the best defences against telecommunications attacks is to deploy security services implemented with mechanisms specified in standards that are thoroughly vetted in the public domain and rigorously tested by third party laboratories, by vendors, and by users of commercial off-the-shelf products. Three services that most often address network user security requirements are confidentiality, message integrity and authentication.

The Internet Engineering Task Force (IETF) with its Transport Layer Security (TLS) has a standard that supports preventing tampering, message forgery, and eavesdropping by encrypting data units, or segments, from one end of the transport layer to the other. In addition, TLS is application protocol independent, which means higher-level protocols like the Hypertext Transfer Protocol (HTTP) can layer on top of the TLS protocol transparently.

Additional details beyond the basic TLS protocol specification are necessary to ensure both security and interoperability. This document provides detail in the form of specific requirements and guidance for using TLS in conjunction with storage systems.

# Information technology —TLS specification for storage systems

## 1   Scope

This document details the requirements for use of the Transport Layer Security (TLS) protocol in conjunction with data storage technologies. The requirements set out in this document are intended to facilitate secure interoperability of storage clients and servers as well as non-storage technologies that may have similar interoperability needs.

This document is relevant to anyone involved in owning, operating or using data storage devices. This includes senior managers, acquirers of storage products and service, and other non-technical managers or users, in addition to managers and administrators who have specific responsibilities for information security and/or storage security, storage operation, or who are responsible for an organization's overall security program and security policy development. It is also relevant to anyone involved in the planning, design and implementation of the architectural aspects of storage security.

## 2   Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 27000, *Information technology — Security techniques — Information security management systems — Overview and vocabulary*

IETF RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, IETF, May 2008

IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*, IETF, August 2008

IETF RFC 5746, *Transport Layer Security (TLS) Renegotiation Indication Extension*, IETF, February 2010

IETF RFC 8446, *The Transport Layer Security (TLS) Protocol Version 1.3*, IETF, August 2018

## 3   Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 27000 and the following apply.

**3.1**
**cipher suite**
named combination of authentication, encryption, and message authentication code algorithms used to negotiate the security settings for a network connection

Note 1 to entry:   Cipher suites are typically used with the Transport Layer Security (TLS) and the Secure Sockets Layer (SSL) network protocols.

**3.2**
**digital certificate**
data structure signed with a digital signature that is based on a public key and which asserts that the key belongs to a subject identified in the structure

**3.3**
**perfect forward secrecy**
security condition in which a leaving entity cannot obtain any subsequent shared secret keys

[SOURCE: ISO/IEC 11770-5:2011, 3.24]

**3.4**
**proxy**
intermediary that acts as both a server and a client for the purpose of making requests on behalf of other clients

**3.5**
**self-signed certificate**
*digital certificate* (3.2) that is signed by the same entity whose identity it certifies

Note 1 to entry:    A self-signed certificate is one signed with its own private key.

**3.6**
**security strength**
number associated with the amount of work that is required to break a cryptographic algorithm or system and specified in bits such that security strength $s$ bits implies the required number of operations is $2^s$

Note 1 to entry:    Common values of security strength are 80, 112, 128, 192, and 256.

[SOURCE: ISO/IEC 9797-2:2021, 3.13, modified — Note to entry 1 has been replaced.]

# 4   Symbols and abbreviated terms

AEAD          Authenticated Encryption with Additional Data

AES           Advanced Encryption Standard

ASCII         American Standard Code for Information Interchange

CA            certificate authority

CBC           cipher block chaining

CDMI          Cloud Data Management Interface

CRL           certificate revocation list

DER           distinguished encoding rules

DHE           Ephemeral Diffie-Hellman

DSA           Digital Signature Algorithm

DTLS          Datagram Transport Layer Security

ECDHE         Elliptic Curve Ephemeral Diffie–Hellman

ECDSA         Elliptic Curve Digital Signature Algorithm

| GCM | Galois/Counter Mode |
|---|---|
| HKDF | HMAC key derivation function |
| HMAC | Hash Message Authentication Code |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| IANA | Internet Assigned Numbers Authority |
| IETF | Internet Engineering Task Force |
| IP | Internet Protocol |
| MAC | message authentication code |
| MD5 | Message Digest 5 |
| OCSP | Online Certificate Status Protocol |
| PEM | Privacy Enhanced Mail |
| PKCS | Public-Key Cryptography Standards |
| PKI | public key infrastructure |
| PRF | pseudorandom function |
| PSK | pre-shared key |
| RFC | Request For Comment |
| RSA | Rivest, Shamir, and Adelman algorithm |
| SHA | Secure Hash Algorithm |
| SMI-S | Storage Management Initiative – Specification |
| SSL | Secure Socket Layer |
| TCP | Transmission Control Protocol |
| TLS | Transport Layer Security |

# 5    Overview and concepts

## 5.1 General

Data storage systems and infrastructure increasingly use technologies such as protocols over TCP/IP to manage the systems and data as well as to access the data. In many situations, the historical reliance on isolated connectivity, specialized technologies, and the physical security of data centers are not sufficient to protect data, especially when the data is considered sensitive and/or high value. Thus, there is a need to include security at the transport layer and at the same time, ensure interoperability.

The objectives for this document are to:

— Specify the TLS elements necessary to secure storage management and data access

— Facilitate timely updates and enhancements to the security for the storage specifications

— Ensure storage clients and systems can interoperate securely

— Support non-storage technologies that may have similar TLS interoperability needs

While many elements of this document can be relevant to Datagram Transport Layer Security (DTLS), no provisions have been added to address DTLS conformance.

## 5.2 Storage specifications

As a starting point, the original TLS requirements described herein were extracted from the following specifications:

— ISO/IEC 17826:2012, Information technology — Cloud Data Management Interface (CDMI)

— SNIA Storage Management Initiative – Specification (SMI-S), Version 1.6.1

These original requirements were then harmonized, eliminating minor differences.

## 5.3 Overview of TLS

### 5.3.1 TLS background

TLS is a protocol that provides communications security over networks. It allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery. TLS is layered on top of a reliable transport protocol (e.g., TCP), and it is used for encapsulation of various higher-level protocols (e.g., HTTP).

Version 1.2 of TLS is specified in IETF RFC 5246. The more recent version 1.3 of TLS is specified in IETF RFC 8446. Earlier, and less secure, versions of TLS are also specified and in use; TLS versions 1.0 is specified in IETF RFC 2246, and TLS versions 1.1 is specified in IETF RFC 4346. The predecessor to TLS, The Secure Sockets Layer (SSL), and in particular, version 3.0 is also in use, but also considered less secure; SSL 3.0 is documented in the historical IETF RFC 6101, *The Secure Sockets Layer (SSL) Protocol Version 3.0*.

### 5.3.2 TLS functionality

TLS provides endpoint authentication and communications privacy over the network using cryptography. Typically, only the server is authenticated (i.e., its identity is ensured) while the client remains unauthenticated; this means that the end user (whether an individual or an application) has a measure of assurance with whom they are communicating. Mutual authentication (the identities of both endpoints are verified) requires, with few exceptions, the deployment of digital certificates on the client.

TLS involves three basic phases:

— Peer negotiation for algorithm support

— Key exchange and authentication

— Symmetric cipher encryption and message authentication

During the first phase, the client and server negotiate cipher suites (see 5.3.3), which determine the ciphers to be used, the key exchange, authentication algorithms, and the Message Authentication Codes (MACs). The key exchange and authentication algorithms are typically public key algorithms. The MACs are made up from a keyed-Hash Message Authentication Code (HMAC).

## 5.3.3 Summary of cipher suites

TLS cipher suite names consist of a set of mnemonics separated by underscores (i.e., "_"). A registered[1] 16-bit (4 hexadecimal digit) number, called the cipher suite index, is assigned for each defined cipher suite. The naming convention in TLS 1.3 differs from the convention shared in TLS 1.0, 1.1, and 1.2. In all TLS cipher suites, the first mnemonic is the protocol name (i.e., "TLS"). Cipher suite names in TLS 1.0. 1.1, and 1.2 have the following form:

TLS_*KeyExchangeAlg*_WITH_*EncryptionAlg_MessageAuthenticationAlg*

where:

*KeyExchangeAlg* consists of one (e.g., RSA, PSK, etc.) or two (e.g., ECDHE_ECDSA) mnemonics.

*EncryptionAlg* indicates the symmetric encryption algorithm and associated mode of operations.

*MessageAuthenticationAlg* is generally the hashing algorithm to be used for HMAC, if applicable.

The following examples illustrate how to interpret the cipher suite names:

— TLS_DHE_RSA_WITH_AES_256_CBC_SHA256: Ephemeral DH is used for the key exchange. The server's ephemeral public key is authenticated using the server's RSA public key. Once the handshake is completed, the messages are encrypted using AES-256 in CBC mode. SHA-256 is used for both the PRF and HMAC computations.

— TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384: Ephemeral ECDH is used for the key exchange. The server's ephemeral public key is authenticated using the server's ECDSA public key. Once the handshake is completed, the messages are encrypted and authenticated using AES-256 in GCM mode, and SHA-384 is used for the PRF. Since an authenticated encryption mode is used, messages neither have nor require an HMAC message authentication code.

TLS 1.3 cipher suites have the following form:

TLS_*AEAD_HASH*

where:

*AEAD indicates the AEAD algorithm that is used for confidentiality, integrity, and message authentication.*

*HASH* indicates the hashing algorithm that is used with the HKDF during key derivation.

The following examples illustrate how to interpret TLS 1.3 cipher suite names:

— TLS_AES_256_GCM_SHA384: Messages are encrypted and authenticated with AES-256 in GCM mode, and SHA-384 is used with the HKDF.

— TLS_AES_128_CCM_SHA256: Messages are encrypted and authenticated with AES-128 in CCM mode, and SHA-256 is used with the HKDF.

The negotiation of the key exchange method is handled elsewhere in the TLS 1.3 handshake.

---

[1]    The Internet Assigned Numbers Authority (IANA) documents registries and important TLS parameters, which can be found at: https://www.iana.org/assignments/tls-parameters/tls-parameters.xml.

To ensure a measure of interoperability between clients and servers, each version of TLS specifies a mandatory cipher suite[2] that all compliant applications are required to implement. The following is the mandatory cipher suite associated with TLS 1.2:

⎯ TLS_RSA_WITH_AES_128_CBC_SHA          {0x00, 0x2F}

The client always initiates the TLS session and starts cipher suite negotiation by transmitting a handshake message that lists the cipher suites (by index value) that it will accept. The server responds with a handshake message indicating which cipher suite it selected from the list or an "abort." Although the client orders its list of cipher suite by preference, starting with the most preferred, the server may choose any of the cipher suites proposed by the client. Therefore, there is no guarantee that the negotiation will select the strongest suite. If no cipher suites are mutually supported, the connection is aborted.

NOTE     When the negotiated options, including optional public key certificates and random data for developing keying material to be used by the cryptographic algorithms, are complete, messages are exchanged to place the communications channel in a secure mode.

## 5.3.4   X.509 digital certificates

TLS uses X.509 version 3 public key certificates that are conformant with the *Certificate and Certificate Extension Profile* defined in Section 4 of IETF RFC 5280. This certificate and certificate revocation list (CRL) profile specifies the mandatory fields included in the certificate as well as optional fields and extensions that may be included in the certificate. These X.509 certificates use a digital signature to bind together a public key with an identity. These signatures will often be issued by a certificate authority (CA) associated with an internal or external public key infrastructure (PKI); however, an alternate approach uses self-signed certificates (the certificate is digitally signed by the very same key-pair whose public part appears in the certificate data). The trust models associated with these two approaches are very different.

NOTE     Self-signed certificates can be used to form a web of trust (trust decisions are in the hands of individual users/administrators) that is considered less secure as there is no central authority for trust (e.g., no identity assurance or revocation). This reduction in overall security, which may still offer adequate protections for some environments, is accompanied by an easing of the overall complexity of implementation.

Section 6 of IETF RFC 5280 identifies the need for clients and servers to perform basic path validation, extension path validation, and CRL validation. These validations include, but are not limited to, the following:

⎯ The certificate is a validly constructed certificate

⎯ The signature is correct for the certificate

⎯ The date of its use is within the validity period (i.e., it has not expired)

⎯ The certificate has not been revoked (applies only to PKI certificates)

⎯ The certificate chain is validly constructed considering the peer certificate plus valid issuer certificates up to the maximum allowed chain depth (applies only to PKI certificates)

X.509 digital certificates come in various formats that involve either binary or textual (ASCII) encoding. These encoded certificates can be stored in file types that have different structures that can include server certificate, the intermediate certificate, the private key (e.g., in a PKCS#12 package/file), etc. as well as possibly including a passphrase to protect a private key. It is usually possible to convert between one certificate file type to another, so support for all file types is typically not needed.

---

2     Section 9 - *Mandatory Cipher Suites* of each of the corresponding TLS IETF RFCs is where these mandatory cipher suites are specified. The mandatory cipher suite is required to be implemented and supported in the absence of an application profile.

### 5.3.5 Quantum computing and TLS

Quantum computing is expected to have significant impacts on the security protections offered by commonly used cryptographic algorithms. In the "post-quantum" era (i.e. when real, working, large-scale quantum computers can be built), the following asymmetric encryption algorithms are expected to be vulnerable to quantum computing:

— Rivest, Shamir, and Adelman (RSA) algorithm

— Elliptic Curve Digital Signature Algorithm (ECDSA)

— Elliptic Curve Diffie-Hellman (ECDH)

— Diffie-Hellman (DH)

While the general guidance for symmetric encryption algorithms (e.g. AES) is to double the key size, this is not always an option.

The development of specifications for quantum resilient algorithms is an active area of work. Pending completion of this work, the IETF RFCs associated with TLS make no provisions for quantum computing at this time. As such, this document does not address the potential requirements for post-quantum cryptography.

## 6  Requirements

## 6.1 TLS protocol requirements

Storage systems functioning as servers shall implement the TLS protocol; however, its use by clients is optional. TLS version 1.2 (specified in IETF RFC 5246) shall be implemented, and TLS version 1.3 should be implemented. Servers shall not support SSL (i.e., versions 1.0, 2.0, and 3.0) or TLS protocol versions prior to 1.2.

NOTE      Implementation of TLS version 1.3 is already mandatory, disallowing all earlier versions, for one or more storage specifications. In addition, the IETF is encouraging all new TLS enabled protocols to be developed to only support TLS version 1.3.

Storage systems shall guard against renegotiation attacks (as outlined in IETF RFC 5746) for versions of TLS prior to version 1.3, using one of the following approaches:

— Option 1: Disable renegotiations[3]

— Option 2: Implement the TLS Renegotiation Indication Extension specified in IETF RFC 5746

Storage systems should guard against replay attacks on zero round-trip time (0-RTT) for TLS version 1.3 by disabling the 0-RTT functionality if session restarts are not required.

## 6.2 Cipher suites

### 6.2.1 Required cipher suites for interoperability with TLS 1.2

Storage systems shall not use MD5 in calculating the default HMAC, which is different than what is specified in the cipher suite. In addition, storage systems shall support:

— selection and use of signature/hash algorithm pairs, using the *supported_signature_algorithms* mechanism in TLS 1.2

— use of SHA-256 or greater strength hashes

---

[3]  This approach may also prevent the use of client-side certificates in certain scenarios.

Storage systems shall use cipher suites that have at least 128 bits of security strength; IETF RFC 7525 suggests that implementations not negotiate cipher suites that use algorithms offering less than 128 bits of security strength. In addition, the following cipher suites shall be supported by storage systems and clients accessing them:

—  TLS_RSA_WITH_AES_128_CBC_SHA             {0x00, 0x2F}[4]

—  TLS_RSA_WITH_AES_128_CBC_SHA256          {0x00, 0x3C}

NOTE     The use of CBC mode encryption carries the theoretical risks associated with padding oracle attacks[5]. Other encryption modes like GCM do not carry these risks.

For some environments, the use of RSA or SHA-1 may be prohibited (see NIST SP 800-52 Revision 2). While the cipher suites listed above are required to be implemented, their use may not be allowed, so consideration should be given to supporting cipher suites listed in 6.2.2.

## 6.2.2   Recommended cipher suites for enhanced security with TLS 1.2

Clause 6.2.1 requires the use of algorithms that have at least 128 bits of security strength with a preference for at least 128 bits of security strength. Several cipher suites are acceptable, based on these criteria. However, there are additional considerations and a need to facilitate interoperability between clients and servers, so the recommendations in this subclause should be implemented.

Forward secrecy (sometimes called "perfect forward secrecy") prevents the recovery of information that was encrypted with older session keys, thus limiting the amount of time during which attacks can be successful. Many defined TLS cipher suites do not feature forward secrecy (e.g., TLS_RSA_WITH_AES_128_CBC_SHA256). IETF RFC 5116 defines authenticated encryption with additional data (AEAD) algorithms that can be used to achieve forward secrecy.

In alignment with the IETF RFC 7525 recommendations on forward secrecy, the following cipher suites should be supported by storage systems and clients accessing them:

—  TLS_DHE_RSA_WITH_AES_128_GCM_SHA256        {0x00,0x9E}

—  TLS_DHE_RSA_WITH_AES_256_GCM_SHA384        {0x00,0x9F}

—  TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256    {0xC0,0x2B}

—  TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384    {0xC0,0x2C}

—  TLS_ECDHE_EDDSA_WITH_AES_128_GCM_SHA256    {0xCC, 0xB1}

—  TLS_ECDHE_EDDSA_WITH_AES_256_GCM_SHA384    {0xCC, 0xB2}

Since the use of digital certificates can add complexity, especially for isolated networks that prevent certificate path verification or for mutual authentication, an alternate approach using pre-shared keys (PSK) is permitted. As such, the following PSK cipher suites should be supported by storage systems and clients accessing them:

—  TLS_DHE_PSK_WITH_AES_128_GCM_SHA256        {0x00,0xAA}[6]

—  TLS_PSK_WITH_AES_128_GCM_SHA256            {0x00,0xA8}

—  TLS_PSK_WITH_AES_256_GCM_SHA384            {0x00,0xA9}

---

[4]   In the absence of an application profile standard this is the mandatory cipher suite for TLS v1.2.

[5]   Additional information on padding oracle attacks can be found in the *Practical Padding Oracle Attacks* paper by Juliano Rizzo and Thai Duong.

[6] This cipher suite is specified in the *TCG Storage Opal SSC Feature Set: PSK Secure Messaging*, Specification Version 1.00, August 2015.

It is possible that compliance with specialized requirements (e.g., NIST SP 800-52 Revision 2) or specific TLS profiles (e.g., IETF RFC 6460) can further restrict or expand the recommended cipher suites an implementation needs to implement.

### 6.2.3   Recommended cipher suites and extensions with TLS 1.3

If TLS 1.3 is supported, the following IETF RFC 8446 mandatory cipher suite shall be implemented:

— TLS_AES_128_GCM_SHA256          {0x13, 0x01}

The following TLS 1.3 cipher suites, recommended by IETF RFC 8446, should also be supported by storage systems and clients accessing them:

— TLS_AES_256_GCM_SHA384          {0x13, 0x02}

— TLS_CHACHA20_POLY1305_SHA256     {0x13, 0x03}

NOTE:  The U.S. Government (i.e., NIST SP 800-52 Revision 2, Section 1.3) has indicated that Agencies shall support TLS 1.3 by January 1, 2024. After this date, servers shall support TLS 1.3 for both government-only and citizen or business-facing applications. In general, servers that support TLS 1.3 should be configured to use TLS 1.2 as well. However, TLS 1.2 may be disabled on servers that support TLS 1.3 if it has been determined that TLS 1.2 is not needed for interoperability.

NOTE       TLS 1.3 cipher suites cannot be used with earlier versions of TLS. TLS 1.3 cannot use cipher suites from earlier versions of TLS.

Per IETF RFC 8446, TLS 1.3 compliant application shall support digital signatures with rsa_pkcs1_sha256 (for certificates), rsa_pss_rsae_sha256 (for CertificateVerify and certificates), and ecdsa_secp256r1_sha256. In addition, clients/servers shall support key exchange with secp256r1 (NIST P-256) and should support key exchange with X25519 (default for popular open source implementations of TLS).

Clients/servers shall implement and use the following TLS extensions:

— The Supported Versions ("supported_versions")

— Cookie ("cookie")

— Signature Algorithms ("signature_algorithms")

— Signature Algorithms Certificate ("signature_algorithms_cert")

— Negotiated Groups ("supported_groups")

— Key Share ("key_share")

— Server Name Indication ("server_name")

— Pre-shared Key Support ("pre_shared_key")

## 6.3 Digital certificates

### 6.3.1   Certificate profile requirements

Storage systems (TLS servers) shall be capable of supporting one or more public key certificates and the associated private keys. The use of public key certificates is optional, but they should be used for management operations and data access.

When public key certificates are used by storage systems and clients that access these systems, the supported certificates shall be X.509 version 3 public key certificates that are conformant with the *Certificate and Certificate Extension Profile* defined in Section 4 of IETF RFC 5280. Both the public key contained in the certificate and the signature shall provide at least 128 bits of security strength.

At a minimum, TLS servers conforming to this document shall be configured with an RSA signature certificate or an ECDSA signature certificate. For RSA server X.509 certificates, key sizes of 3072 bits or greater (evenly divisible by 8) shall be used. If the server is configured with an ECDSA signature certificate, either curve, NIST P-384 with SHA-384, or NIST P-521 with SHA-512 shall be used for the public key in the certificate; for interoperability, clients and servers should support the curve NIST P-384

### 6.3.2   Certificate validity and path validation requirements

The certificate validity period is the time interval during which the CA warrants that it will maintain information about the status of the certificate. Certificates used for storage systems shall not have a validity period greater than 398 days.[7]

TLS clients/servers shall validate certificates in accordance with certificate path validation rules specified in Section 6 of IETF RFC 5280. The revocation status of each certificate in the certification path (certificate chain) shall be checked.

Revocation information shall be obtained by the TLS client/server from one or more of the following locations:

1   CRL or Online Certificate Status Protocol (OCSP), as described IETF RFC 6960, response in the server's local store;

2   OCSP response from a locally configured OCSP responder;

3   OCSP response from the OCSP responder location identified in the OCSP field in the Authority Information Access extension in the certificate; or

4   CRL from the CRL distribution points extension in the certificate.

When the local store does not have the current or a cogent[8] CRL or OCSP response and the OCSP responder and the CRL distribution point are unavailable or inaccessible at the time of TLS session establishment, either deny the connection or accept a potentially revoked or compromised certificate. The decision to accept or reject a certificate in this situation should be made according to organizational policy.

### 6.3.3   Certificate encoding requirements

Several security-related standards used on the Internet define ASN.1 data formats that are normally encoded using the Basic Encoding Rules (BER) or Distinguished Encoding Rules (DER)[9], which are binary, octet-oriented encodings. Alternatively, the binary encoding can be substituted with a Base64 (ASCII) encoding; a common example is Privacy Enhanced Mail (PEM) file that are just Base64 encoded DER files.

Storage clients/servers shall support both binary DER and Base64 (ASCII) DER encodings. This support shall include the ability to input the corresponding file types (e.g., ".der" and ".cer" for binary DER as well as ".pem" and ".crt" for Base64 DER). In addition, clients/servers shall support input of PKSC#12 files (e.g., ".pfx" and ".p12").

## 6.4 Compression methods

The use of compression may enable attackers to perform attacks using compression-based side channels. To defend against these attacks, the null compression method shall be enabled, and all other compression methods shall be disabled.

---

[7] Based on the CA/Browser Forum, *Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates*, Version 1.7.2.

[8] A CRL is considered "cogent" when the "CRL Scope" (see IETF RFC 5280) is appropriate for the certificate in question.

[9] See ITU-T X.690.

# 7 Guidance for the implementation and use of TLS in data storage

## 7.1 Digital certificates

### 7.1.1 Certificate model

Digital certificates are used to identify servers (or less commonly clients) and provide cryptographic keys for use in communication. These certificates can either be public key certificates or self-signed certificates (as noted in 5.3.4). Public key certificates typically provide more reliable identity assurances but require prior planning and supporting infrastructure (e.g., certificate authorities). Self-signed certificates are very easy to deploy but do not provide reliable identity assurance, so they should not be used on anything but test systems. Organizations should decide between the two models based on their risk profile and available resources.

### 7.1.2 Chain of trust

The confidence in the identity assurance provided by a certificate depends on confidence in the entity (i.e., CA) issuing the certificate. Often a trusted CA will issue a certificate to an organization, which will then issue its own certificates (this creates the "chain of trust"). When using public key certificates, organizations should explicitly identify the certificate authorities that are allowed to issue certificates for use within the organization. These trusted CAs should be configured in the client (e.g., web browsers).

### 7.1.3 Certificate lifecycle

Certificates need to be issued, installed, replaced and ultimately removed/revoked. Effective governance of this certificate lifecycle is dependent on the organization developing sound policies and procedures. A commonly overlooked decision is the lifetime of the certificate. Certificates have expiration dates to specify the maximum time a certificate is valid (much like other forms of identity assurances) and at the end of their lifetime, they have to be replaced to avoid "certificate expired" errors. When setting the lifetime, carefully consider the risk, complexity of the certificate request and installation, and the number of certificates involved. For example, if the certificate/request installation process requires 1.5 person hours and there are 10,000 certificates in the organization, setting the certificate lifetime to 1 year would require 8 full-time employees (10,000 x 1.5 divided by 8 hours per workday divided by 230 workdays per year) just to replace the certificates.

Removal of certificates from devices being repurposed or leaving organization control is an essential measure to protect the organization against unauthorized access.

If a certificate is compromised (e.g., its private key is revealed to an unauthorized person) or if the certificate is no longer needed, the certificate should be revoked (see 7.1.4) to prevent its further use.

### 7.1.4 Revocation

Certificates need to be invalidated (revoked) when they are no longer useful or they have been compromised (e.g., the private key associated with the certificate has come into the possession of an unauthorized party). Certificate revocation is simply the process of adding a certificate to the CRL. As described in 6.3.2, clients and other certificate users are required to check the validity of a certificate (i.e., in part to verify that it has not been revoked) before making use of a certificate.

The following important certificate revocation issues should be addressed:

— The sources of the revocation information should be relevant and trustworthy

— The revocation information should be as "fresh" as possible, especially when high value or sensitive data is involved (e.g., a CRL can have lengthy expiration dates that masks the need to retrieve a more current version of the CRL)

— CRLs, similar to certificate chains, can be large so adequate provisions should be made to store and process CRLs when they are used for validation