SDC 20

BY Developers FOR Developers

Storage Developer Conference
September 22-23, 2020

TLS for Storage Systems

Eric Hibbard, CISSP, CITP, CISA
PrivSec Consulting LLC

About the Speaker

Eric Hibbard, CISSP-ISSAP, ISSMP, ISSEP, CIPT, CISA, CCSK

Security/Privacy Professional
eric.Hibbard@gmail.com

Chair, SNIA Security Technical Work Group
Chair, INCITS TC CS1 Cyber Security
Chair, IEEE Computer Society, Cybersecurity & Privacy Standards Committee (CPSC)
Co-Chair, Cloud Security Alliance (CSA) – International Standardization Council (ISC)
Member, American Bar Association – Science & Technology (SciTech) Law Council
Member, American Bar Association – Cybersecurity Legal Task Force
Co-Chair, American Bar Association – SciTech Law – Internet of Things (IoT) Committee
ISO Editor: ISO/IEC 27040, ISO/IEC 27050 (multi-part), ISO/IEC 17788, ISO/IEC 22123 (multi-part), ISO/IEC 20648
IEEE Editor: IEEE Std 1619 (XTS-AES)

Abstract

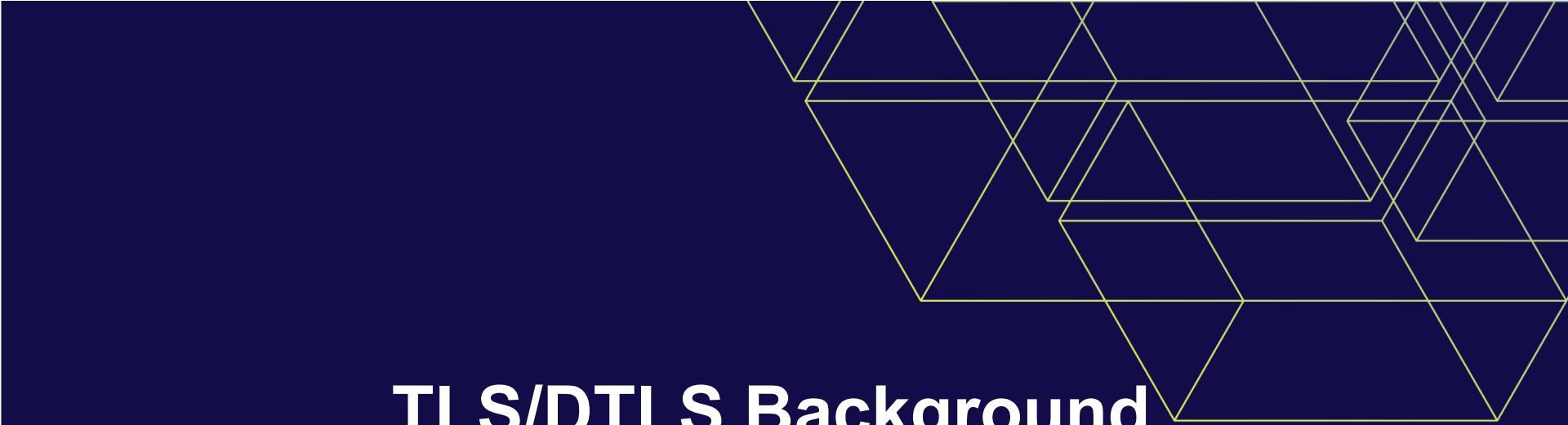
Transport Layer Security (TLS), sometimes referred to as SSL (deprecated predecessor), is an important mechanism for preventing eavesdropping, tampering, and message forgery of network-based communications between clients and servers. The stream-oriented TLS is designed to run on top of a reliable transport protocol (e.g., TCP); however, the Datagram Transport Layer Security (DTLS) provides similar security guarantees for datagram-based applications. To fully exploit the security protections of TLS and DTLS, care must be exercised in selecting certain options and features (e.g., cipher suites) as well as correctly handling operational details (e.g., certificate validation and management). As with many aspects of security, TLS/DTLS must be adjusted to respond to changes in the threat landscape, so these adjustments need to be factored into TLS/DTLS implementations and use.

TLS and DTLS, to a lesser degree, are important security protocols used with many storage systems, which increasingly use RESTful APIs and Web-based management interfaces (e.g., SMI-S, CDMI, and Swordfish). This session highlights important TLS/DTLS details that are relevant to storage systems. In addition, information will be provided on recent changes and anticipated changes that could have an impact on storage infrastructures.

Agenda

- Background and key aspects of TLS/DTLS
- Storage issues associated with TLS/DTLS use
- Common security expectations and requirements for TLS/DTLS

- Note: Unless stated otherwise, the TLS materials presented apply to DTLS as well.



TLS/DTLS Background

A Few Terms

- SSL = Secure Socket Layer
- TLS = Transport Layer Security
- DTLS = Datagram Transport Layer Security

TLS Overview

- TLS is a protocol that provides authenticated, confidentiality and integrity-protected communication between two endpoints.
- 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 some reliable transport protocol (e.g., TCP), and it is used for encapsulation of various higher-level protocols (e.g., HTTP).

A Little TLS History

- Netscape developed SSL 2.0 (1995) and SSL 3.0 (1996)
- Internet Engineering Task Force (IETF) develops and maintains TLS
  - TLS WG is under the IETF Security Area
  - Specified as Requests for Comment (RFC)
- Internet Assigned Numbers Authority (IANA) documents registries and important TLS parameters
  - https://www.iana.org/assignments/tls-parameters/tls-parameters.xml

IETF RFCs for TLS

- IETF RFC 2246, *The TLS Protocol Version 1.0*, January 1999
- IETF RFC 4346, *The Transport Layer Security (TLS) Protocol Version 1.1*, April 2006
- IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*, August 2008
- IETF RFC 8446, *The Transport Layer Security (TLS) Protocol Version 1.3*, August 2018
- Numerous TLS related RFC

Noteworthy

- TLS and SSL names often used interchangeably, but technically, they are different, since each describes a different version of the protocol
- The various versions of SSL and TLS do not interoperate; they are not backwards compatible
- All versions of TLS prior to TLS 1.2 have security defects or vulnerabilities that make them unacceptable for most organizations

Known Attacks (RFC 7457)

- SSL Stripping
- STARTTLS Command Injection Attack
- BEAST
- Padding Oracle Attacks (Lucky 13 & POODLE)
- Compression Attacks (CRIME, TIME, BREACH)
- Certificate and RSA-Related Attacks
- Renegotiation
- Triple Handshake
- Denial of Service

TLS Record Protocol

- Layered on top of a reliable transport protocol (e.g., TCP)
- Provides connection security by using symmetric cryptography for confidentiality, data origin authentication, and integrity protection
- Used for encapsulation of various higher-level protocols

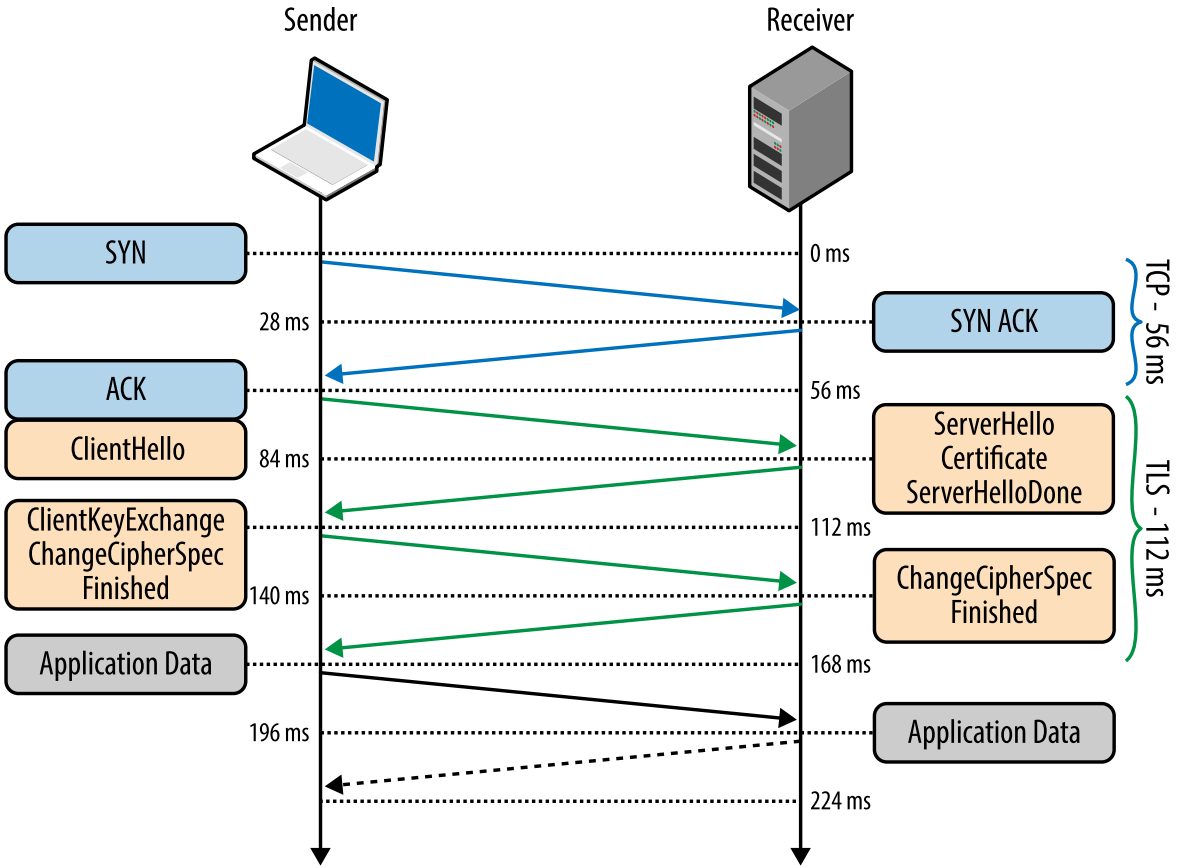
TLS Handshaking Protocols

Handshake protocol **-** allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives data

Change Cipher Spec protocol **-** exists to signal transitions in ciphering strategies

Alert protocol **-** used to convey errors (fatal and warnings) and specifies a long list of error types

TLS Handshake Protocol



Understanding Cipher Suites

- *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.
- TLS 1.3 uses a different convention

Mandatory Cipher Suites

- Each version of TLS employs a different mandatory cipher suite
  - TLS 1.0: TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
  - TLS 1.1: TLS_RSA_WITH_3DES_EBE_CBC_SHA
  - TLS 1.2: TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS 1.3: TLS_AES_128_GCM_SHA256

- In the absence of a “profile” these mandatory cipher suites prevail
- IETF has a draft RFC that deprecates SHA-1, which complicates the situation for pre-TLS 1.3

Digital Certificates

- TLS uses X.509 version 3 public key certificates (conformant with Section 4 of IETF RFC 5280)
- Certificates use a digital signature (typically CA-issued) to bind together a public key with an identity
- Clients and servers need to perform basic path validation
- Options for various encoding formats

Certificate Validation

- Certificate is a validly constructed X.509 certificate
- Signature is correct for the certificate
- The date of its use is within the validity period
- The certificate has not been revoked
- The certificate chain is validly constructed considering the peer certificate plus valid issuer certificates up to the maximum allowed chain depth
- Use Certificate Revocation List (CRL) or Online Certificate Status Protocol (OCSP) for revocation information

DTLS Overview

- DTLS is an “adaptation” of TLS for UDP
  - DTLS 1.0 is equivalent to TLS 1.1 (no RC4)
  - DTLS 1.2 is equivalent to TLS 1.2
  - Draft DTLS 1.3 is equivalent to TLS 1.3
- Unlike TLS, the application has to deal with
  - packet reordering
  - loss of datagram
  - data larger than a datagram network packet

Storage Related Issues

SNIA TLS Specification for Storage (ISO/IEC 20648)

- Specifies the TLS elements necessary to secure storage management and data access
- Facilitates timely updates and enhancements to the security for the storage specifications
  - SMI-S, CDMI, Swordfish, etc.
- Ensures storage clients and systems can interoperate securely
- Supports non-storage technologies that may have similar TLS interoperability needs

TLS 1.2 Configurations

- Full control over enabled versions
  - Default: TLS 1.2 Enabled (and preferred)
  - Default: All SSL+TLS 1.0+1.1 Disabled
- Full control over crypto (affects cipher suites)
  - Default: NULL+anon+EXPORT Disabled
  - Default: RC2+RC4+DES+3DES Disabled
  - Default: MD5+SHA Disabled
  - Support: TLS_RSA_WITH_AES_128_CBC_SHA256

TLS 1.2 Configurations (cont.)

- Select and use of signature/hash algorithm pairs (SHA-256 or greater strength hashes), using the *supported_signature_algorithms* mechanism
- Use CA-issued certificates whenever possible
  - Control over installing certificates and trust stores
  - Control over certificate validation (CRL/OCSP, chain depth, etc.)

TLS 1.2 Configurations (cont.)

- Guard against renegotiation attacks with either:
  - Option 1: Renegotiation Disabled
  - Option 2: Implement the TLS Renegotiation Indication Extension
- Control over using compression Disabled
- Support both binary Distinguished Encoding Rules (DER) and Base64 (ASCII) DER encodings

Certificate Signing Request (CSR)

- Used to apply for a digital identity certificate
- Requires:
  - applicant to have or generates a key pair, keeping the private key secret
  - information identifying the applicant (such as a distinguished name)
  - applicant's public key

Expectations & Considerations

Interoperability

- With all the options available, clients and servers may not successfully negotiate a connection
- Connections should utilize the most secure option available
- Protocols that use TLS may impose restrictions
- TLS profiles like ISO/IEC 20648 and NIST SP 800-62r1 can help ensure secure connections

Crypto Agility

- The threat landscape is constantly changing
- New attacks or discovered defects can render algorithms and mechanism unsafe to use
- Support the broadest set of options possible
- Support the latest TLS versions as soon as possible
- Include functionality to completely control allowed/disallowed options

Certificate Considerations

- Public key contained in the certificate and the signature need to provide at least 112 bits of security
- Validity period not greater than 2 years (possibly 1 year)
- Do not use self-signed certificates; pre-shared keys (PSK) are designed for isolated IT.

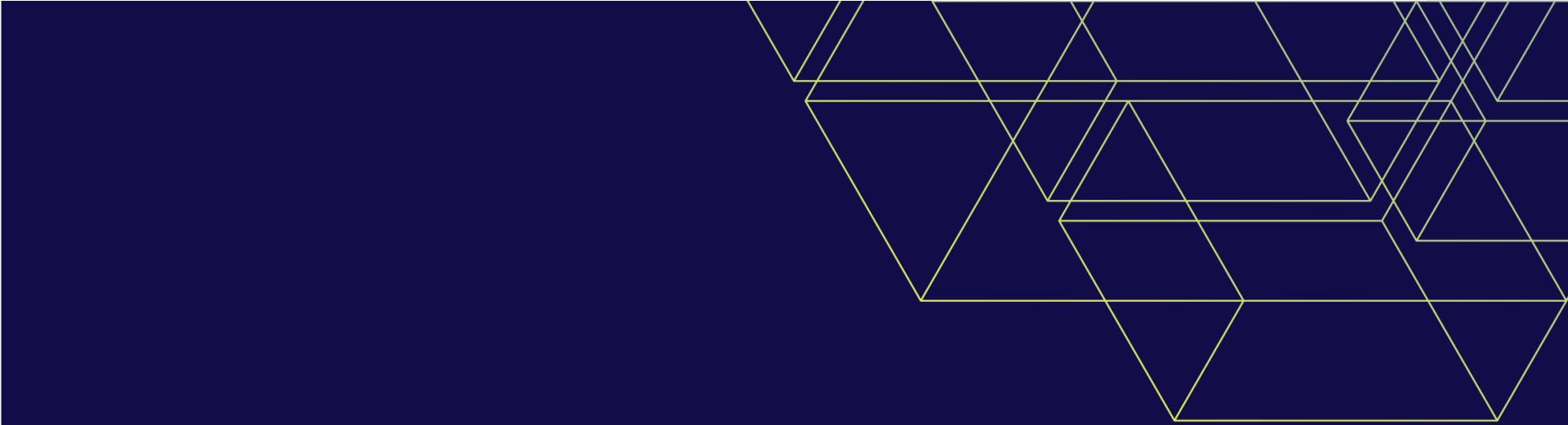
TLS & Security Scans

- All commercial security vulnerability scanning software will probe TLS
- Having older versions or weak options enabled will result in negative findings (flagged as vulnerabilities)

TLS 1.3

- TLS 1.3 was the first major rewrite (modernized)
- Eliminates a number of older algorithms that did nothing other than create vulnerabilities
- Lighter weight than its predecessor and uses fewer resources; reduced latency
- Uses the same certificates and keys as TLS 1.2
- A black swan event is likely to cause a rapid migration to TLS 1.3, so be ready

Thank You



Please take a moment
to rate this session.

Your feedback matters to us.