Lecture 14: TLS 1.3 and IPsec

TTM4135

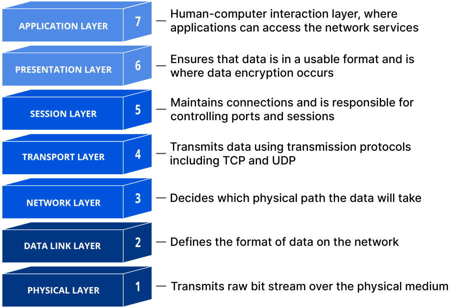
Relates to Stallings Chapters 17 and 20

Spring Semester, 2025

Motivation

- TLS 1.3 is the latest version of the Transport Layer Security protocol and has significant changes from earlier versions affecting both security and efficency
- Internet Protocol security (IPsec) is a framework for ensuring secure communications over Internet Protocol (IP) networks
- IPsec provides similar security services as TLS, but at a lower layer in the communications protocol stack

OSI model – TLS/ IPsec



TLS operates at the application layer, IPsec operates at the network layer.

Outline

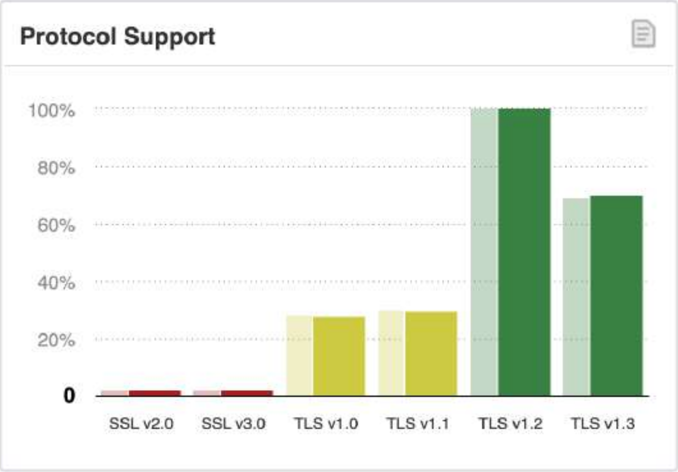
Why was TLS 1.3 needed?

Efficiency: In earlier TLS versions need at least two round trip times (RTT) before data can be sent

Security: Many security problems in earlier TLS versions
- Protocol was too complex
- Protocol supported old and weak ciphersuites

- New protocol designed to support sound cryptographic principles and aims to achieve *provable security*
- First drafted in 2014 in close cooperation between academics, practitioner community and developers
- Internet proposed standard RFC 8446 January 2018
- Today supported in around 66% of popular web servers according to SSL Pulse alongside earlier TLS versions

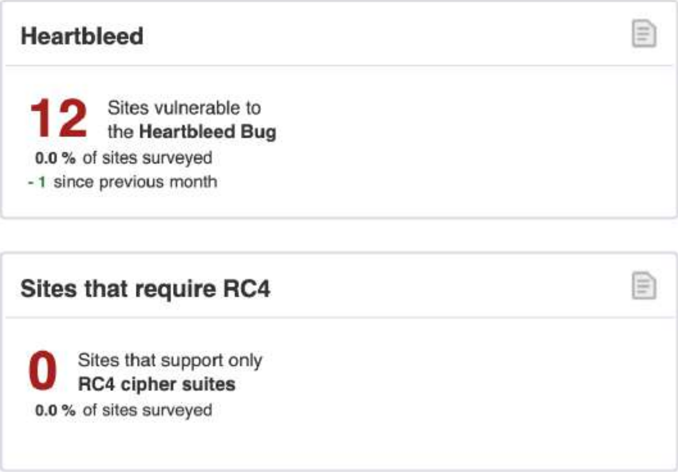
Protocol support



Poodle


POODLE TLS
Not vulnerable
134,331 99.9%
+ 0.0 %
Vulnerable and exploitable
24 < 0.1%
+ 0.0 %
Unknown
140 0.1%
+ 0.0 %

Heartbleed – RC4



Some concrete changes from TLS 1.2 to TLS 1.3

Some items removed:

- static RSA and DH key exchange (why?)
- renegotiation
- SSL 3.0 negotiation
- DSA in finite fields
- data compression
- non-AEAD cipher suites

Some items added:

- zero round-trip time (0-RTT) mode from pre-shared keys
- post-handshake client authentication through “certificate verify” signature
- more AEAD ciphersuites

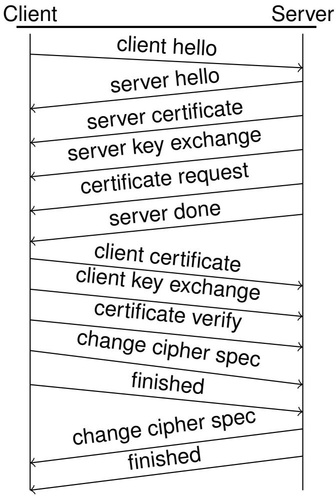
TLS 1.3 handshake protocol: hello messages

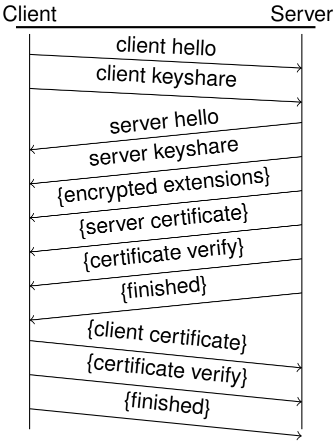
- Client sends keyshare field in client hello for one or more anticipated ciphersuites
- Server can obtain session key on receipt of client hello if:
  - server accepts one the clients ciphersuites
  - client keyshare matches the accepted ciphersuite
- If the above conditions fail then:
  - server sends an optional Hello Retry Request
  - client responds if there is an acceptable alternative ciphersuite
- Usually this results in saving a *whole round trip* of communication

TLS 1.3 handshake protocol: other messages

- Only client and server hello/keyshare messages are not cryptographically protected — all later parts of the protocol use handshake traffic keys
- Key calculation now uses the standard HKDF (hash key derivation function) to derive the individual keys instead of the ad hoc PRF used in TLS 1.2
- Several different key types derived from master secret:
  - *handshake traffic keys* to protect handshake protocol
  - *application traffic keys* for client-server traffic
  - *early data keys* for 0-RTT data (see below)
- Various "tricks" used to allow interoperability with devices that only accept earlier TLS versions

Handshake: TLS 1.2 (left) to TLS 1.3 (right)





{} protected by handshake traffic keys

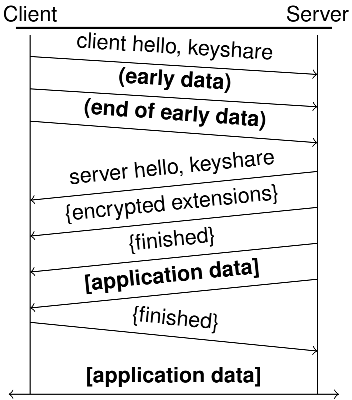
Client authentication

- In TLS 1.2 and 1.3 it is optional for the client to send a certificate and authenticate using a `CertificateVerify` message
- The `CertificateVerify` message includes a signature (with the secret key corresponding to the public key in the certificate) which can be verified using the public key in the certificate
- TLS 1.3 adds a *post-handshake client authentication* extension; if this is used then the server may request client authentication *at any time* after the handshake completed
- The client responds with its certificate and a signature in the form of `CertificateVerify`

0-RTT in TLS 1.3

- In a 0-RTT key establishment parties can start sending application data immediately, so-called early data
- 0-RTT in TLS 1.3 is based on a pre-shared key (PSK)
- PSK can either be agreed outside TLS or from an earlier TLS session
- At the end of the handshake protocol the server can send to the client one or more *new session tickets* as PSKs
- A client may start a new PSK session without negotiating version and ciphersuite

0-RTT in TLS 1.3



() protected by early data keys
{} protected by handshake traffic keys
[] protected by further traffic keys

- Only possible with pre-shared key
- Pre-shared key is used to authenticate DH
- Early data is optional and lacks forward secrecy

TLS 1.3 ciphersuites

- Handshake always uses Diffie-Hellman option so ciphersuite specifies only:
  - which AEAD cipher to use in Record layer
  - hash function to use for KDF
- TLS 1.2 and lower ciphersuite values cannot be used with TLS 1.3 and *vice versa*
- Mandatory to implement ciphersuite: `TLS_AES_128_GCM_SHA256`
- Other recommended ciphersuites: `TLS_AES_256_GCM_SHA384`, `TLS_CHACHA20_POLY1305_SHA256`, `TLS_AES_128_CCM_SHA256`, `TLS_AES_128_CCM_8_SHA256`

ChaCha algorithm

- Stream cipher defined in RFC 8439 together with a message authentication code (MAC) called Poly1305
- Available in a TLS ciphersuite (RFC 7905)
- Designed by D. J. Bernstein in 2008
- Faster than AES, except for processors with AES hardware support (most modern desktop computers)
- Combines $\oplus$, addition modulo 2^{32} and rotation operations over 20 rounds to produce 512 bits of keystream. An example of an *add-rotate-xor* or ARX cipher.
- 256-bit key

TLS 1.3 main improvements

Efficiency
- Saving of one round trip time in handshake
- Can set up follow-on session with 0-RTT

Security
- Only forward-secret key exchange now allowed
- Many legacy cipher suites no longer allowed
- Renegotiation option removed
- Formal security proofs

Selfie Attack on TLS 1.3

- Published in March 2019 by Drucker and Gueron
- Breaks mutual authentication in PSK mode
- Suppose Alice shares a PSK with Bob
- Attacker reflects messages back to herself so client Alice believe she is talking to Bob while she is actually talking with server Alice
- Case is not covered in formal analysis of TLS 1.3
- Can be prevented by forbidding to share PSK between more than one server and one client

IPsec: Introduction

- Standardised in RFCs 4301-4304 (2005) with crypto algorithms updated in subsequent RFCs
- Provides protection for any higher layer protocol
- Uses encryption, authentication and key management algorithms
- Most commonly used to provide Virtual Private Networks (VPNs)
- Provides a security architecture for both IPv4 and IPv6

Security services

Message confidentiality Protects against unauthorised data disclosure by the use of encryption

Message integrity Detects if data has been changed by using a message authentication code (MAC) or authenticated encryption

Limited traffic analysis protection Eavesdropper on network traffic should not know which parties communicate, how often, or how much data is sent

Message replay protection The same data is not replayed and data is not delivered badly out of order

Peer authentication Each IPsec endpoint confirms the identity of the other IPsec endpoint

Gateway-to-gateway architecture

- Provides secure network communications between two networks
- Network traffic is routed through the IPsec connection, protecting it appropriately
- Only protects data between the two gateways
- Most often used when connecting two secured networks, such as linking a branch office to headquarters over the Internet
- Can be less costly than private wide area network (WAN) circuits

Host-to-gateway architecture

- Commonly used to provide secure remote access.
- The organization deploys a virtual private network (VPN) gateway onto their network
- Each remote access user establishes a VPN connection between the local computer (host) and the gateway
- VPN gateway may be a dedicated device or part of another network device
- Most often used when connecting hosts on unsecured networks to resources on secured networks

Host-to-host architecture

- Typically used for special purpose needs, such as system administrators performing remote management of a single server
- Only model that provides protection for data throughout its transit (end-to-end)
- Resource-intensive to implement and maintain in terms of user and host management
- All user systems and servers that will participate in VPNs need to have VPN software installed and/or configured
- Key management is often accomplished through a manual process

IPsec protocol types

Encapsulating Security Payload (ESP) Can provide confidentiality, authentication, integrity and replay protection

Authentication Header (AH) Authentication, integrity and replay protection, but no confidentiality and is now deprecated

Internet Key Exchange (IKE) negotiate, create, and manage session keys in so-called *security associations*

Setting up an IPsec connection

- Key exchange uses IKEv2 protocol specified in RFC 7296 (2014)
- IKEv2 uses Diffie–Hellman protocol authenticated using signatures with public keys in X.509 certificates
- Includes *cookies* to mitigate denial-of-service attacks:
  - client must return a time-dependent cookie value before the server proceeds
  - they provide *proof of reachability* before any expensive cryptographic processing is completed

Security associations

- A security association (SA) contains info needed by an IPsec endpoint to support an IPSec connection
- Can include cryptographic keys and algorithms, key lifetimes, security parameter index (SPI), and security protocol identifier (ESP or AH)
- SPI is included in the IPSec header to associate a packet with the appropriate SA
- SA tells the endpoint how to process inbound IPSec packets or how to generate outbound packets
- SAs are needed for each direction of connection
- IKEv2 is used to establish keys to use in SAs

Cryptographic suites

- Similar to TLS ciphersuites, there are a number of standardised cryptographic suites, incorporating both public key and symmetric key algorithms
- Specific groups available for Diffie–Hellman, both in finite fields and on elliptic curves
- 3DES or AES can be used for encryption, either in CBC or GCM
- HMAC or CMAC (variant) is used for integrity if GCM mode is not used

IPsec modes of operation

- Each protocol (ESP or AH) can operate in transport or tunnel mode
- Transport mode: Maintains IP header of the original packet and protects payload — generally only used in host-to-host architectures
- Tunnel mode: Original packet encapsulated into a new one, payload is original packet — typical use is gateway-to-gateway architecture
- We show the pictures for IPv4 — there are slight differences for IPv6

IPsec protocol components

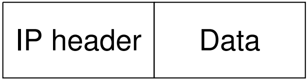
ESP header Contains the security parameter index (SPI) identifying the SA and sequence numbers

ESP trailer Contains padding and padding length — may also include extra padding to enhance traffic flow confidentiality

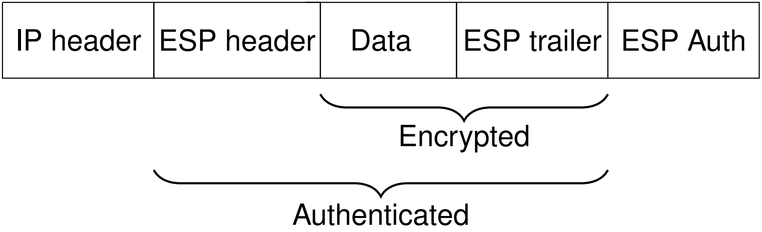
ESP Auth Contains MAC of the encrypted data and ESP header — may not be required if an authenticated encryption mode is used

Transport mode ESP

- Original IP packet



- IP Packet protected by Transport-ESP

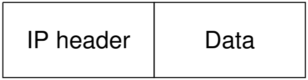


ESP in transport mode: Outbound packet processing

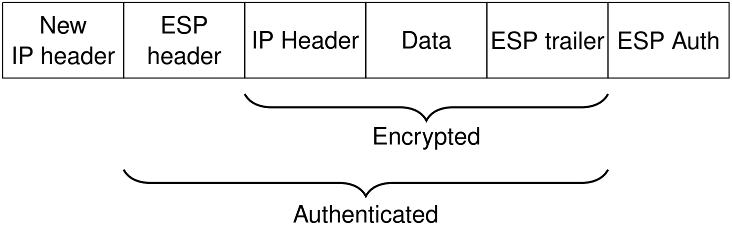
- Data after the original IP header is padded by adding an ESP trailer and result encrypted using the symmetric cipher and key in the SA
- An ESP header is prepended
- If an SA uses the authentication service, an ESP MAC is calculated over the data prepared so far and appended
- Original IP header is prepended, but some fields in the original IP header must be changed:
    - protocol field changes from TCP to ESP
    - total length field must be changed to reflect the addition of the ESP header
    - checksums must be recalculated

Tunnel mode ESP

- Original IP packet



- IP Packet protected by Tunnel-ESP



ESP in tunnel mode: Outbound packet processing

- Entire original packet is padded by adding an ESP trailer and the result encrypted using the symmetric cipher and key agreed in the SA
- ESP header is prepended
- If the SA uses the authentication service, an ESP MAC is calculated over the data prepared so far and appended
- New outer IP header is prepended
  - Inner IP header of the original IP packet carries the ultimate source and destination addresses
  - Outer IP header may contain distinct IP addresses such as addresses of security gateways
  - Outer IP header protocol field is set to ESP

IPsec security

- Active attacks have been demonstrated for *encryption-only* mode of ESP protocol — now widely understood that providing encryption without integrity is insecure
- Unlike earlier versions of IPsec, the 2005 version does not require implementations to support encryption-only mode, but still allows it
- ESP applies encryption before MAC in normal usage
- Using AH, a MAC can be applied before encryption, as in TLS. Attacks have been demonstrated on such configurations
- Formal analysis has shown that IPsec key exchange protocol (IKEv2) has no significant weaknesses