IT-Security

Chapter 6: Network Security Protocols

on Network and Transport Layer

Prof. Dr.-Ing. Ulrike Meyer

Overall Lecture Context

- **In the past lectures we have learned how to**
  - Protect confidentiality with symmetric or asymmetric encryption
  - Protect integrity (including replay) with MACs or digital signatures
  - Establish session keys between authenticated entities
- **In this chapter we will learn how these mechanisms are used in network security protocols**
- **In particular, we will study and compare IPSec, and TLS**

Overview

IPSec

- Primary use cases
- Security services offered
- Authentication and key agreement
- IP Payload of IP packet protection

TLS

- Primary use case
- Security services offered
- Authentication and key agreement
- TCP payload protection

Comparison of the protocols

- Differences
- Communalities in mechanisms used
- Overlaps in use cases

Overview

IPSec

- Main use case
- Security services offered
- Authentication and key agreement
- Payload or packet protection

TLS

- Main use case
- Security services offered
- Authentication and key agreement
- Payload protection

Comparison of the protocols

- Differences
- Communalities in mechanisms used
- Overlaps in use cases

Overview IPSec Part

Introduction

- Historical notes
- Security services offered by IPSec
- Transport Mode and Tunnel Mode
- Primary Use Cases

Authentication Header Protocol AH

- Integrity Protection in the two modes
- ESP Header
- MAC computation
- Supported algorithms in AH and ESP
- Replay protection in AH and ESP

Encapsulating Security Payload Protocol ESP

- Encryption and Integrity Protection
- ESP Header
- MAC computation

Authentication and Key Agreement with IKEv2

- The concept of security associations
- Overview on detailed discussion of IKEv2
- IP packet processing with IPSec
- Example use cases

IPSec over the Years

- **IPsec is a protocol family**
- **Originally comprising**
  - ISAKMP for transporting key management messages
  - IKEv1 for authenticated key agreement carried over ISAKMP
  - ESP/AH protocol for encryption and integrity protection
- **Recommended today**
  - IKEv2 for authentication and key agreement
  - ESP/AH protocol for encryption and integrity protection
- **We focus on the latest versions of these protocols**

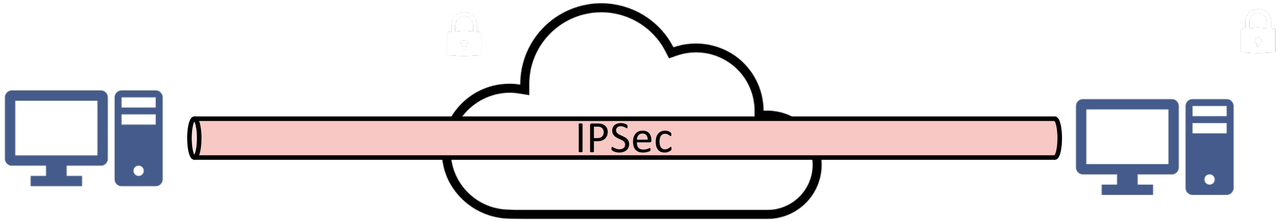
Security Services offered by IPSec

- **Authenticated Session Key Exchange**
  - Using the **Internet Key Exchange Protocol**
  - Based on **pre-shared keys** or based on **certificates**
- **IP packet level encryption and/or IP packet level integrity protection**
  - Including replay protection
  - Using the **Encapsulating Security Payload Protocol**
  - And/or using the **Authentication Header Protocol**
  - **Transport mode**
    - Protection of IP payload of all IP packets exchanged between two IPsec-enabled hosts
  - **Tunnel mode**
    - Protection of complete IP packets routed between IPsec-enabled gateways

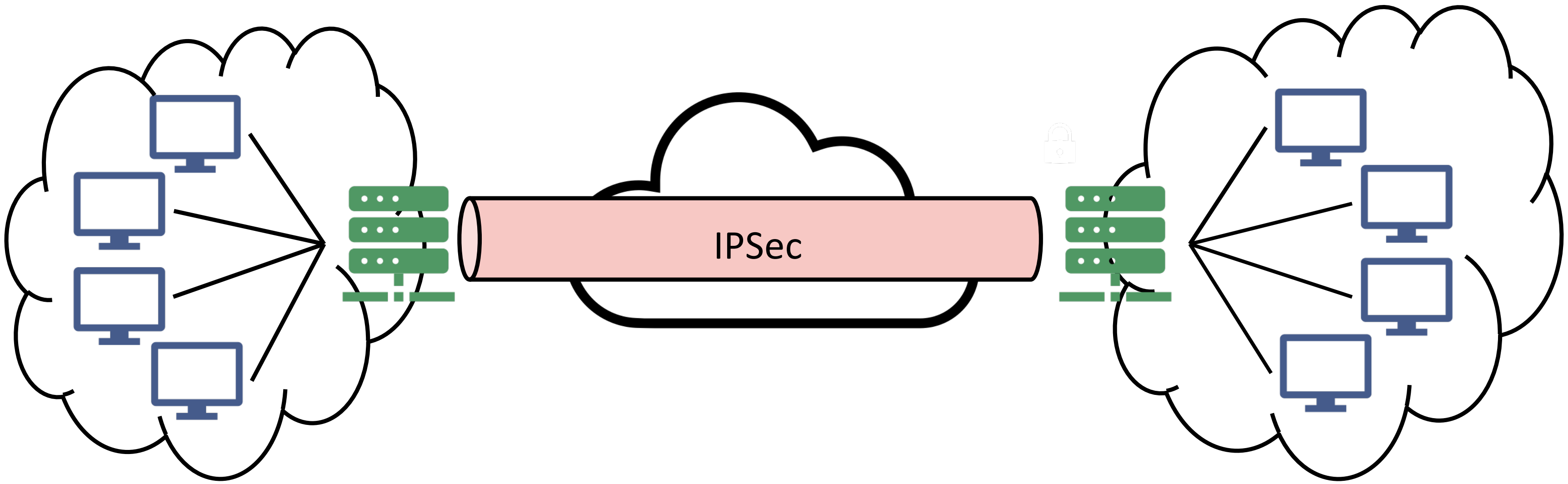
Usable on top of IPv4 and IPv6
Transparent to higher layer protocols

Tunnel Mode and Transport Mode and Primary Use Cases

Transport mode between any two individual nodes

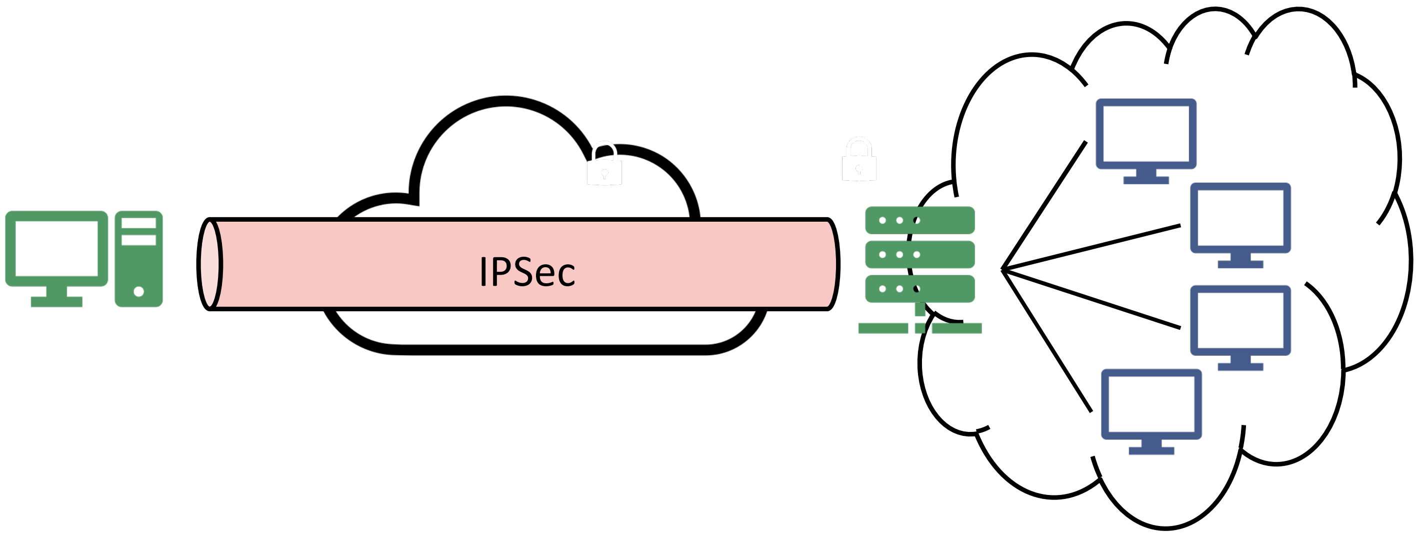


Tunnel mode, e.g., for securely connecting the networks of two branches of a company



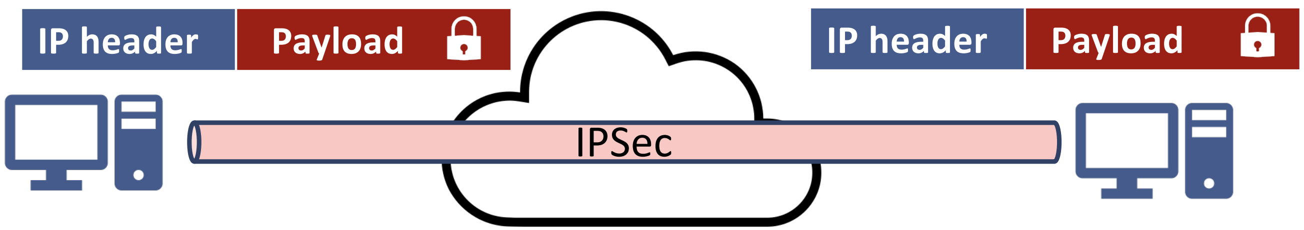
VPN Use Case

IPSec in tunnel mode is also used to connect remote hosts to an internal network

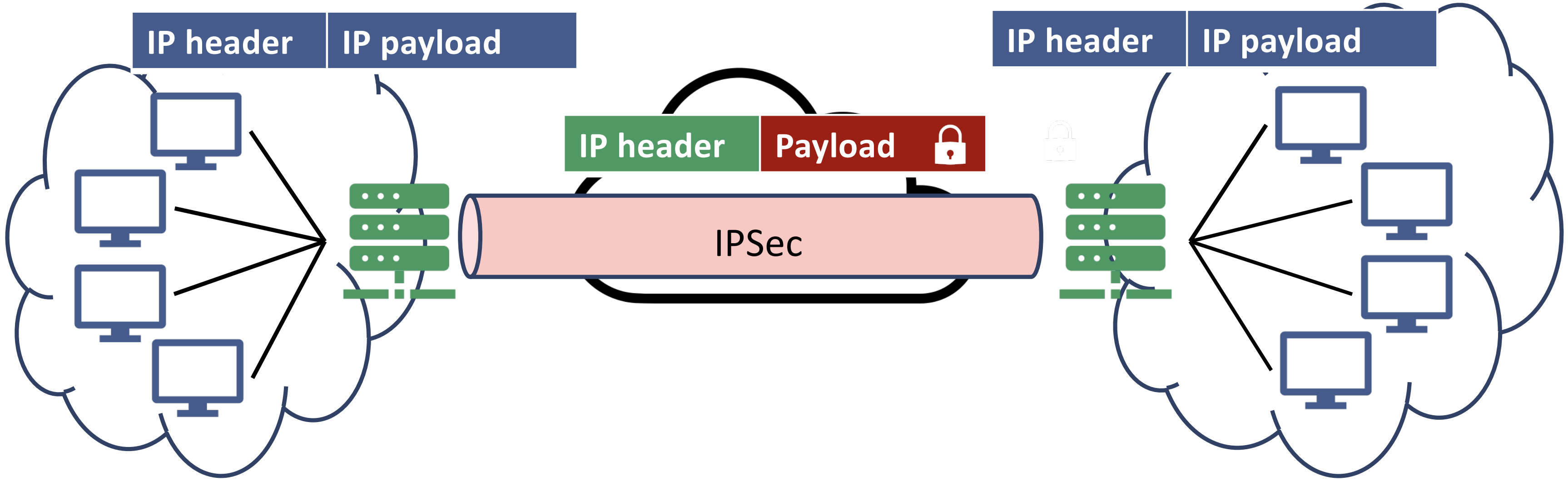


Tunnel Mode and Transport Mode and Primary Use Cases

Transport mode

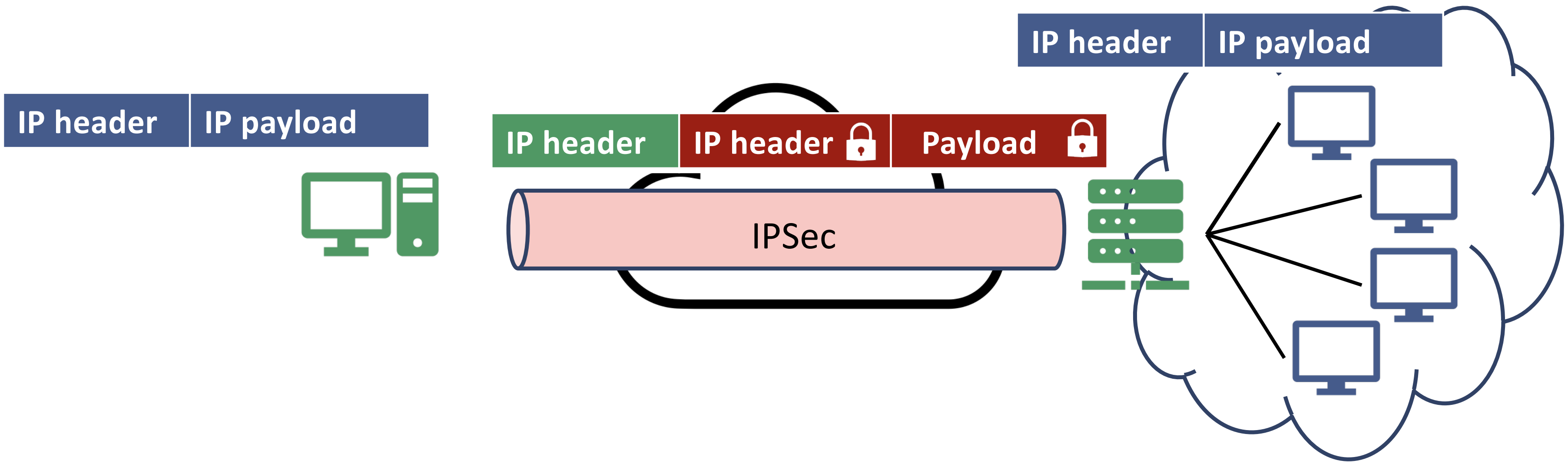


Tunnel mode, e.g., for securely connecting the networks of two branches of a company



VPN Use Case

IPSec in tunnel mode is also used to connect remote hosts to an internal network



Overview IPSec Part

Introduction

- Historical notes
- Security services offered by IPSec
- Transport Mode and Tunnel Mode
- Primary Use Cases

Authentication Header Protocol AH

- Integrity Protection in the two modes
- ESP Header
- MAC computation
- Supported algorithms in AH and ESP
- Replay protection in AH and ESP

Encapsulating Security Payload Protocol ESP

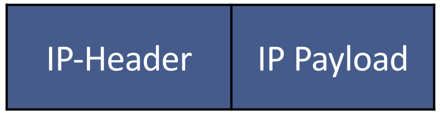
- Encryption and Integrity Protection
- ESP Header
- MAC computation

Authentication and Key Agreement with IKEv2

- The concept of security associations
- Overview on detailed discussion of IKEv2
- IP packet processing with IPSec
- Example use cases

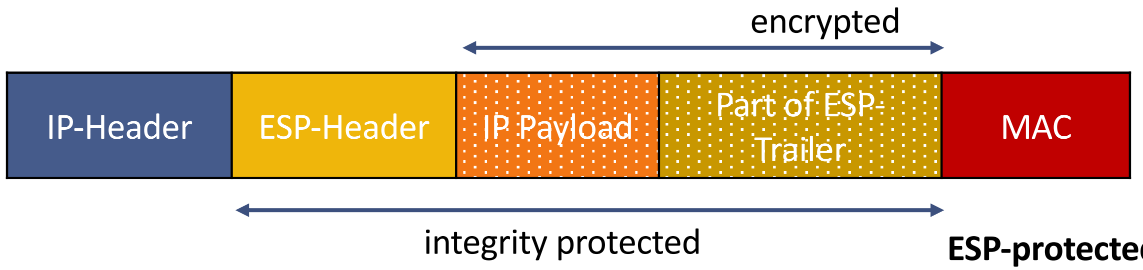
Encryption and Integrity Protection offered by ESP

- **Encapsulating Security Payload protocol**



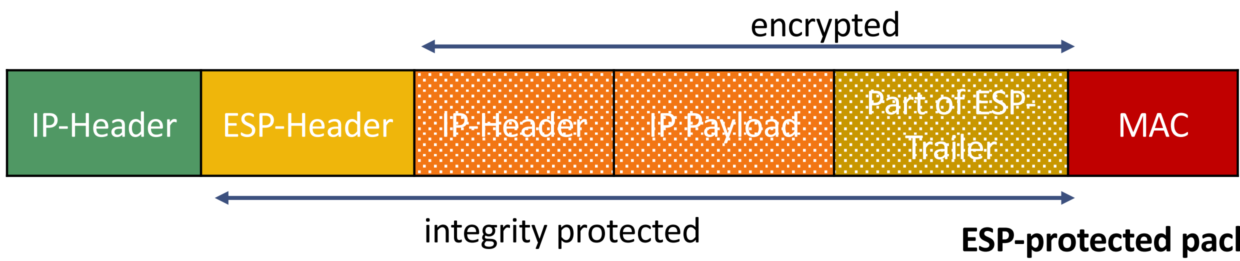
IP packet to be protected

- ▶ Transport mode between two IPsec-enabled hosts
  - Encryption of IP payload
  - Integrity protection of IP payload
  - Replay protection of IP packets



ESP-protected packet

- ▶ Tunnel mode between two IPsec-enabled gateway
  - Encryption of IP packets including IP headers routed through the gateway
  - Integrity protection of IP packets including IP headers routed through the gateway
  - Replay protection of IP packets



ESP-protected packet

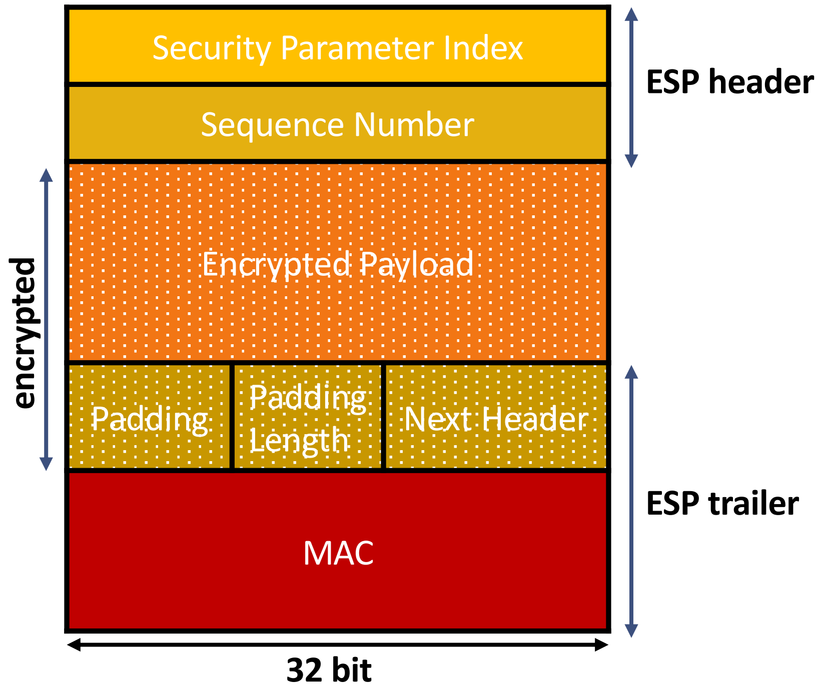
Payload of an ESP Protected IP Packet

ESP header

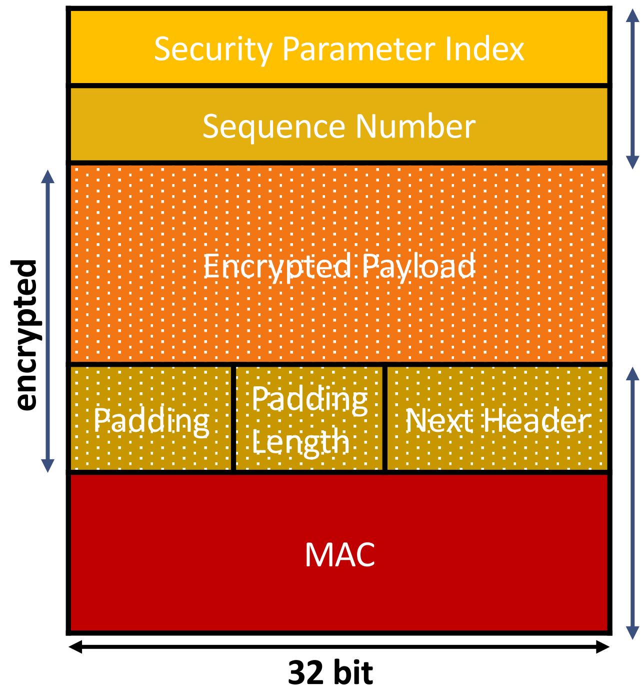
- **Security Parameter Index** (SPI): 32-bit
  - Identifies a security association (SA)
  - Specified what keys and algorithms to use
- **Sequence number**: 32-bit number per packet
  - Used for replay protection

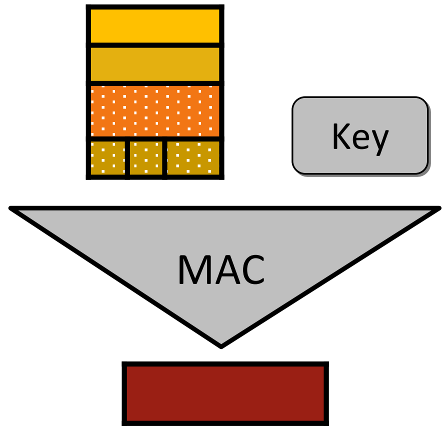
ESP trailer

- **Padding field**: 0-255 padding bits
- **Padding length:** 8-bit length field
- **Next header:** 8-bit filed indicating type of payload encrypted in the encrypted payload
- **MAC:** message authentication code



MAC Computation





MAC not computed on IP header

Overview IPSec Part

Introduction

- Historical notes
- Security services offered by IPSec
- Transport Mode and Tunnel Mode
- Primary Use Cases

Authentication Header Protocol AH

- Integrity Protection in the two modes
- ESP Header
- MAC computation
- Supported algorithms in AH and ESP
- Replay protection in AH and ESP

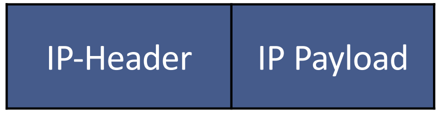
Encapsulating Security Payload Protocol ESP

- Encryption and Integrity Protection
- ESP Header
- MAC computation

Authentication and Key Agreement with IKEv2

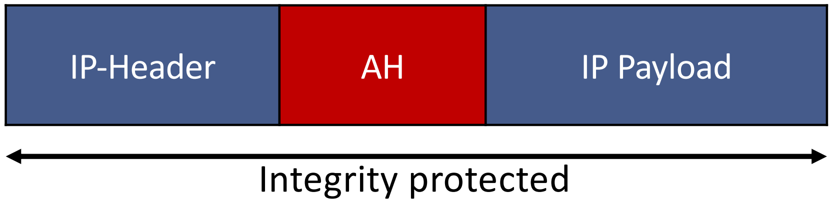
- The concept of security associations
- Overview on detailed discussion of IKEv2
- IP packet processing with IPSec
- Example use cases

Integrity Protection offered by AH



IP packet to be protected

- **Authentication Header Protocol**
  - **Transport mode** between two IPsec enabled hosts
    - Integrity protection of the complete IP packet, including the header



AH-protected packet

  - **Tunnel mode** between two IPsec-enabled gateway
    - Integrity protection of the complete IP packet, including the new IP header

AH-protected packet

Authentication Header

Next header field

- 8-bit field, indicates type of header following the AH header
  - IP header in tunnel mode, first header in IP payload in transport mode

Payload length

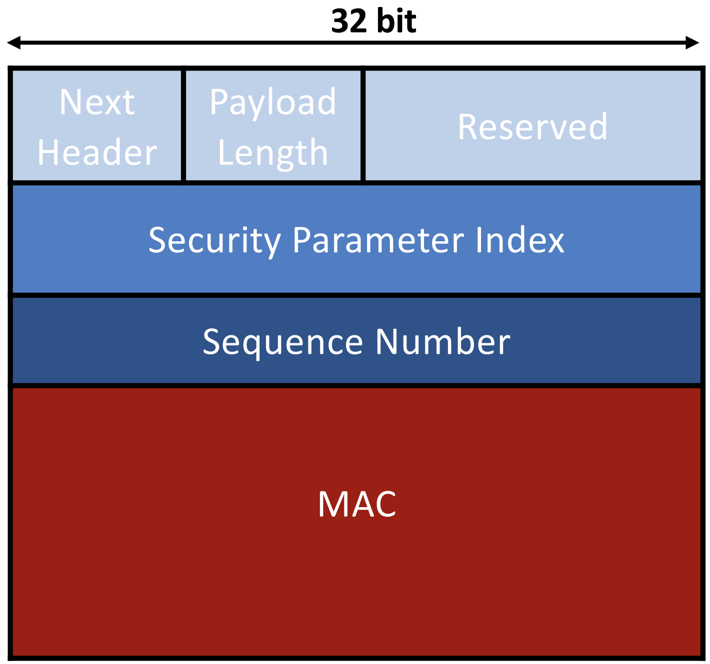
- 8-bit field defining length of authentication header
  - Depending on MAC algorithm, length of authentication data varies

Security Parameter Index (SPI)

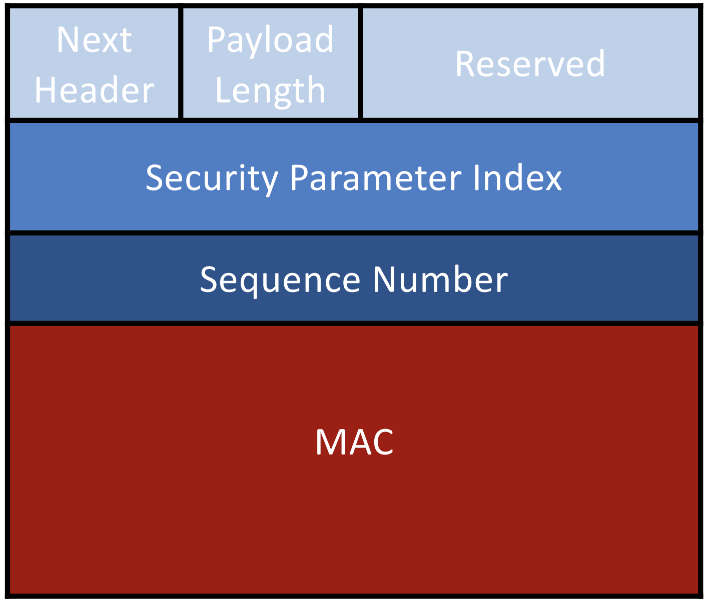
- 32-bit identifier of a security association (SA)
- Specified what keys and algorithms to use

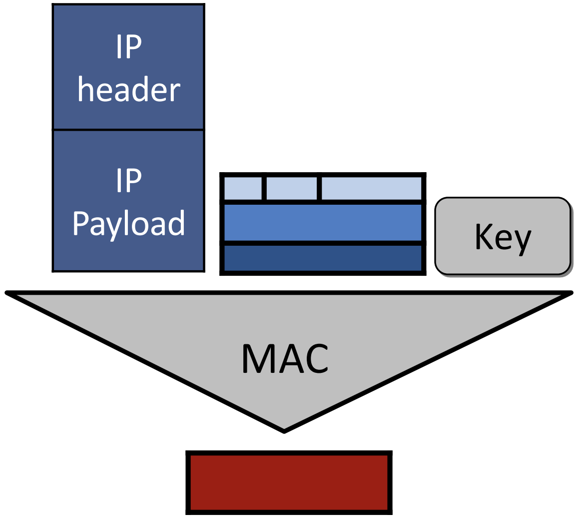
Sequence number

- 32-bit sequence number incremented with each packet, used for replay protection



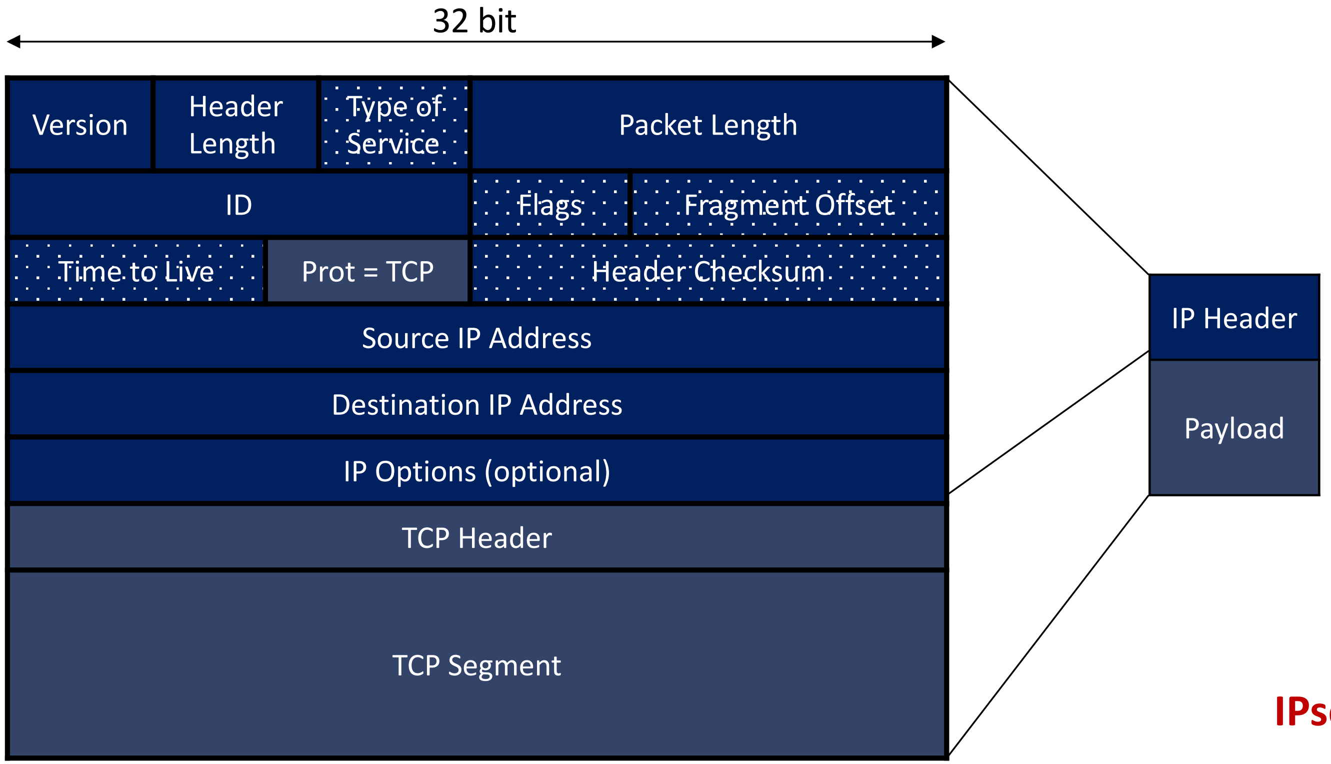
MAC Computation





- **Authentication header fields included in MAC-computation**
- **Non-mutable fields of outer IP header included in MAC-computation**
  - Mutable fields such as TTL, Header Checksum, Fragment Offset etc. can and should not be protected

Recap: Mutable Fields in the IPv4 Header



IPsec supports both IPv4 and IPv6!

Mutable fields  are set to zero for the MAC calculation in AH

Most recent MAC algorithm support for AH RFC 8221

Name	Status
AUTH_NONE	MUST NOT
AUTH_HMAC_MD5_96	MUST NOT
AUTH_HMAC_SHA1_96	MUST- (=expected to be phased out soon)
AUTH_DES_MAC	MUST NOT
AUTH_KPDK_MD5	MUST NOT
AUTH_AES_XCBC_96	SHOULD for IoT / MAY otherwise
AUTH_AES_128_GMAC	MAY
AUTH_AES_256_GMAC	MAY
AUTH_HMAC_SHA2_256_128	MUST
AUTH_HMAC_SHA2_512_256	SHOULD

Recommendations change over time, latest ones currently from 2017

XCBC is a predecessor of CMAC that differs in the generation of the masking keys

Most recent MAC algorithm support for ESP RFC 8221

Name	Status
AUTH_NONE	MUST (in comb. with combined enc/integ.) / MUST NOT
AUTH_HMAC_MD5_96	MUST NOT
AUTH_HMAC_SHA1_96	MUST- (=expected to be phased out soon)
AUTH_DES_MAC	MUST NOT
AUTH_KPDK_MD5	MUST NOT
AUTH_AES_XCBC_96	SHOULD for IoT / MAY otherwise
AUTH_AES_128_GMAC	MAY
AUTH_AES_256_GMAC	MAY
AUTH_HMAC_SHA2_256_128	MUST
AUTH_HMAC_SHA2_512_256	SHOULD

Recommendations change over time, latest ones currently from 2017

Same as for AH except for the first one

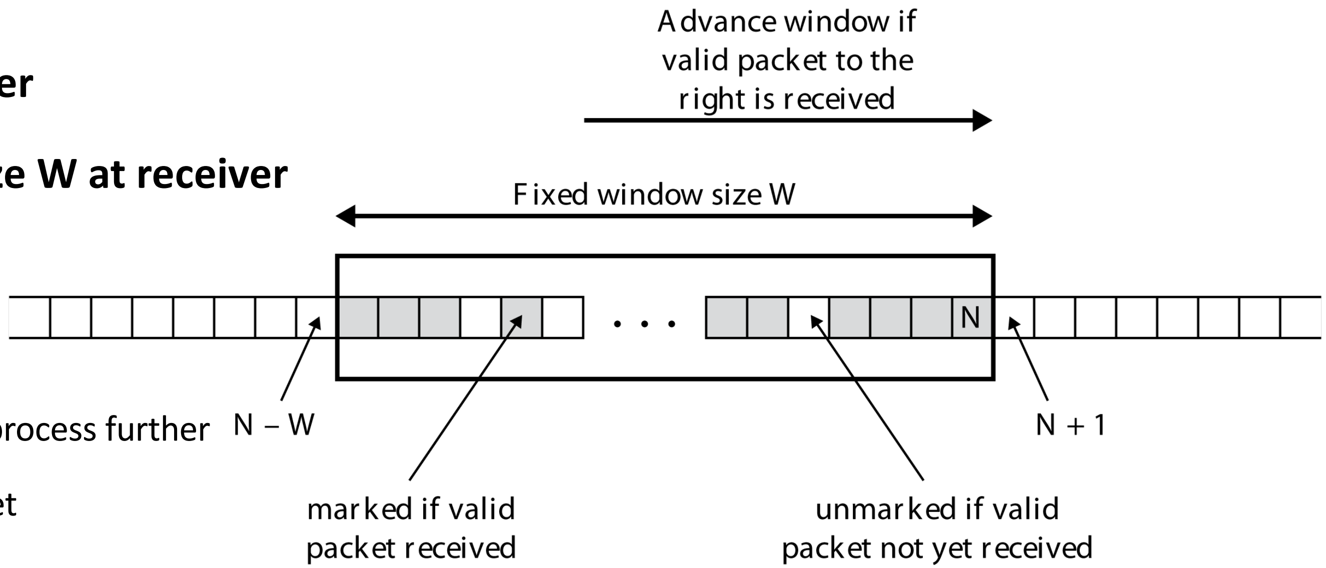
Most recent Encryption algorithm support for ESP RFC 8221

Name	Status
ENCR_DES_IV64	MUST NOT
ENCR_DES	MUST NOT
ENCR_3DES	SHOULD NOT
ENCR_BLOWFISH	MUST NOT
ENCR_3IDEA	MUST NOT
ENCR_DES_IV32	MUST NOT
ENCR_NULL	MUST
ENCR_AES_CBC	MUST
ENCR_AES_CCM	SHOULD (provides integrity as well)
ENCR_AES_GCM	MUST (provides integrity as well)
ENCR_CHACHA20_POLY1305	SHOULD (provides integrity as well)

Recommendations change over time, latest ones currently from 2017

Replay Protection in ESP and AH

- **SQN included in ESP and AH header**
- **Window of acceptable SQNs of size W at receiver**
- **SQN checking at receiver**
  - If SQN is in current window
    - **and not yet marked**: mark and process further
    - **and already marked:** drop packet
  - If SQN is lower than left boarder of window
    - drop packet, log event
  - If SQN is higher than current right boarder
    - mark as received, move window to include SQN as right boarder, process packet further



Recommended window size

- Should be ≥ 32
- Recommends default 64

Overview IPSec Part

Introduction

- Historical notes
- Security services offered by IPSec
- Transport Mode and Tunnel Mode
- Primary Use Cases

Authentication Header Protocol AH

- Integrity Protection in the two modes
- ESP Header
- MAC computation
- Supported algorithms in AH and ESP
- Replay protection in AH and ESP

Encapsulating Security Payload Protocol ESP

- Encryption and Integrity Protection
- ESP Header
- MAC computation

Authentication and Key Agreement with IKEv2

- The concept of security associations
- Overview on detailed discussion of IKEv2
- IP packet processing with IPSec
- Example use cases

Authentication and Key Agreement

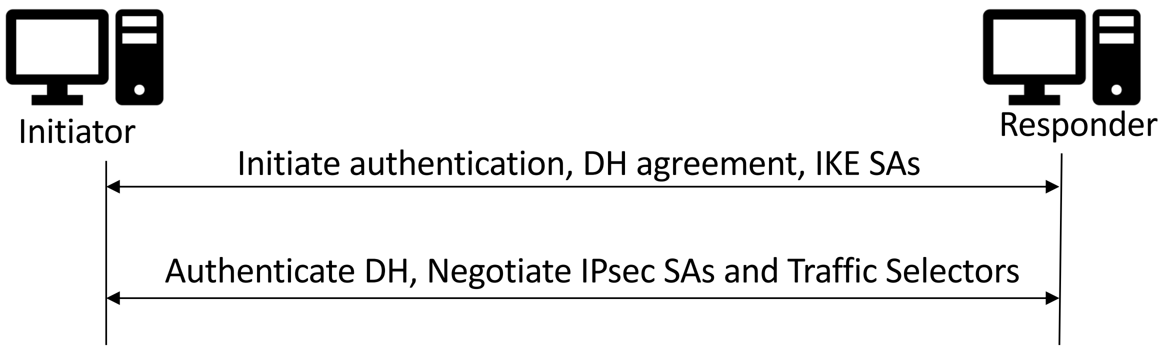
Security Association

- Identified uniquely by a 32-bit Security Parameter Index **SPI**
- **Security protocol type:** determines if the SA is for IKE, AH or ESP usage
- **Algorithm information:** Encryption and / or MAC algorithms, keys
- **Replay Window:** Current start point and size of replay window
- **SQN:** Current Value of the sequence number SQN
- **IPSec Mode:** Indicates if SA is usable for transport mode, tunnel mode or both
- **SA lifetime:** Lifetime of the security association
  - lifetime can be based on time, byte count, or both

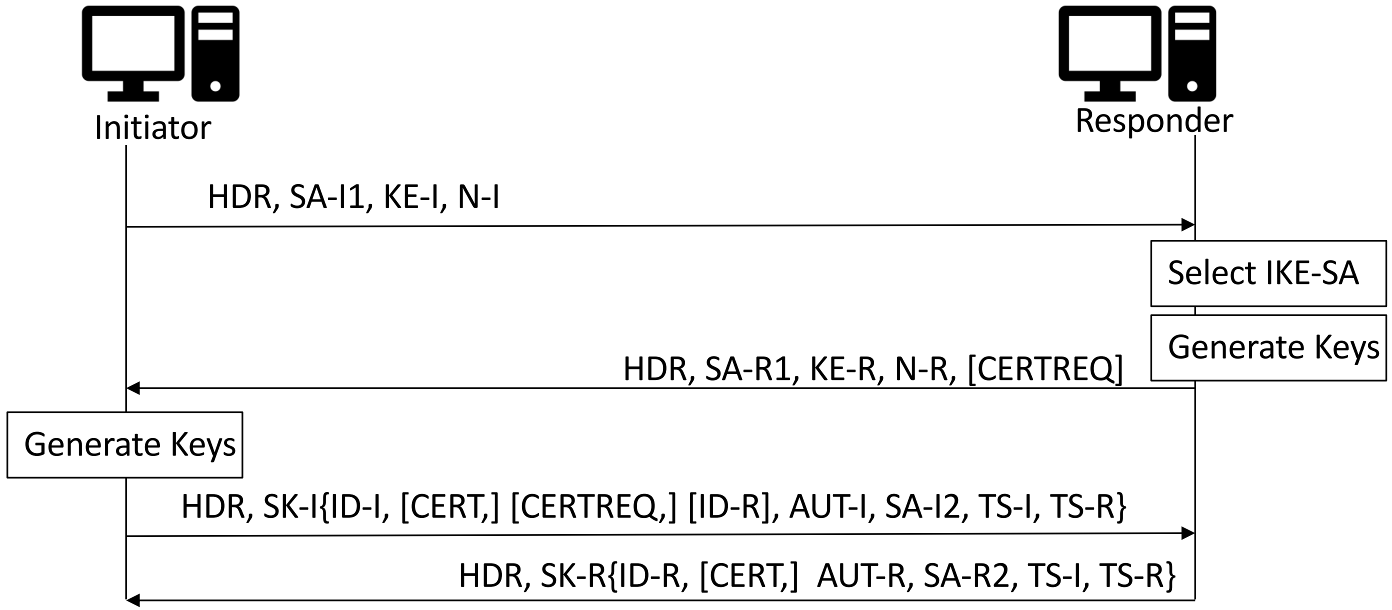
Authentication and Key Agreement

The Internet Key Management Protocol IKEv2

- Supports authentication and key agreement between two IPsec-enabled peers
  - Establishes at least two pairs of **security associations** (SAs) between the peers
  - One **IKE-SA** pair to protect the authentication and key agreement itself
  - One **IPSec-SA** pair to use with ESP and / or AH in tunnel or transport mode later
- Peer starting the protocol is called the **initiator**, the other peer is called **responder**



IKE v2 Exchange: Complete Overview



HDR: header, contains SPIs
SA-I1: SAs offered for IKE
SA-R2: SA selected
KE-I, KE-R: public DH values
N-I, N-R: nonces

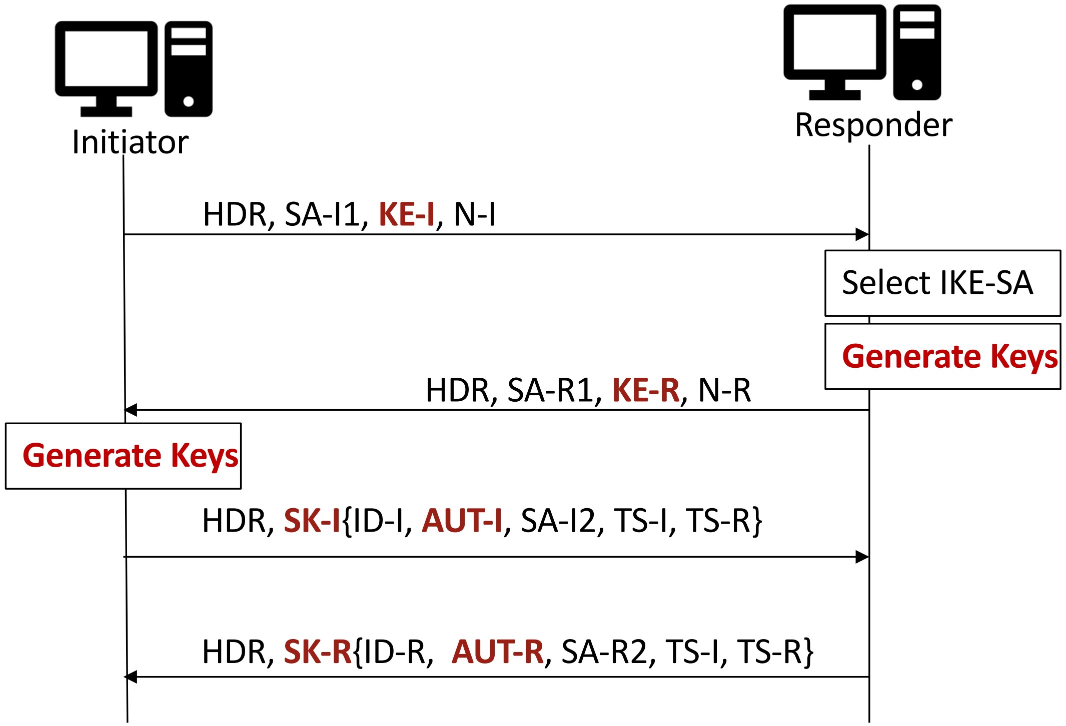
CERTREQ: certificate request
CERT: certificate
ID-I, ID-R: identifier
AUT-I, AUT-R: sign. or MAC
SA-I2: SA offered for IPsec

SA-R2: SA selected for IPsec
TS-I, TS-R: traffic selectors
SK-I, SK-R: encrypted with SKe and integrity protected with SK-a

IKE v2 Exchange: Authentication and Key Agreement

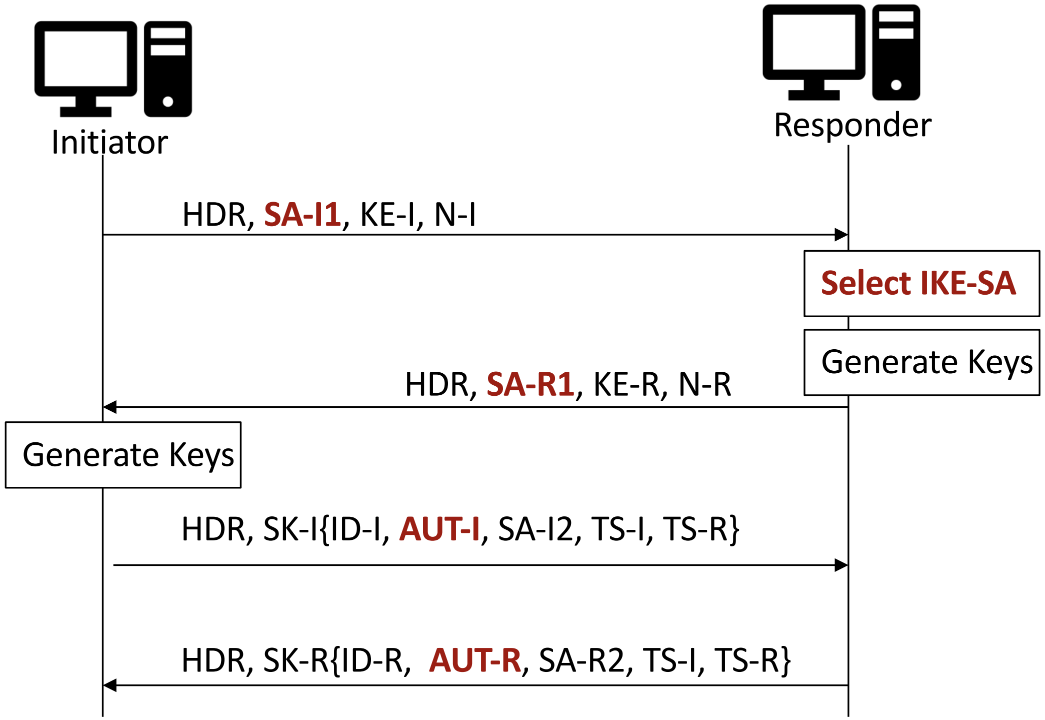
Variant of the secure authenticated DH

- DH values **KE-I** and **KE-R** exchanged in the clear
- Keys for encryption and integrity protection during IKE exchanged and further key derivation for IPSec SAs derived from nonces and DH values
- Authenticated by **AUT-I** and **AUT-R** using digital signatures or pre-shared keys
  - AUT-I = sign(h(message 1 || N-R || $MAC_{SKp\text{-}i}$(ID-I)) or AUT-I = MAC(message 1 || N-R || $MAC_{SKp\text{-}i}$(ID-I))
  - AUT-R = sign(h(message 2 || N-I || $MAC_{SKp\text{-}r}$(ID-R)) or AUT-R = MAC(message 2 || N-I || $MAC_{SKp\text{-}r}$(ID-R))
- Message 3 and 4 encrypted and integrity protected



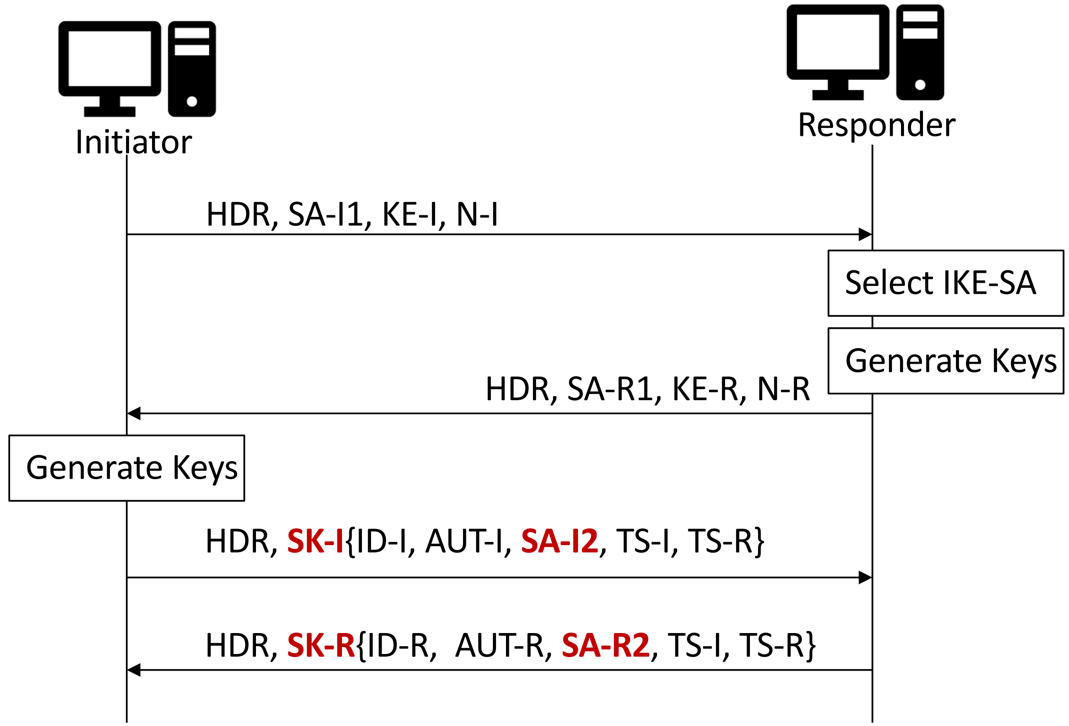
IKE v2 Exchange: IKE-SA Negotiation

- Initiator sends proposals for IKE-SAs in message 1
- Responder selects IKE-SA and includes selection in message 2
- Proposal and selection protected against manipulation with **AUT-I** and **AUT-R**



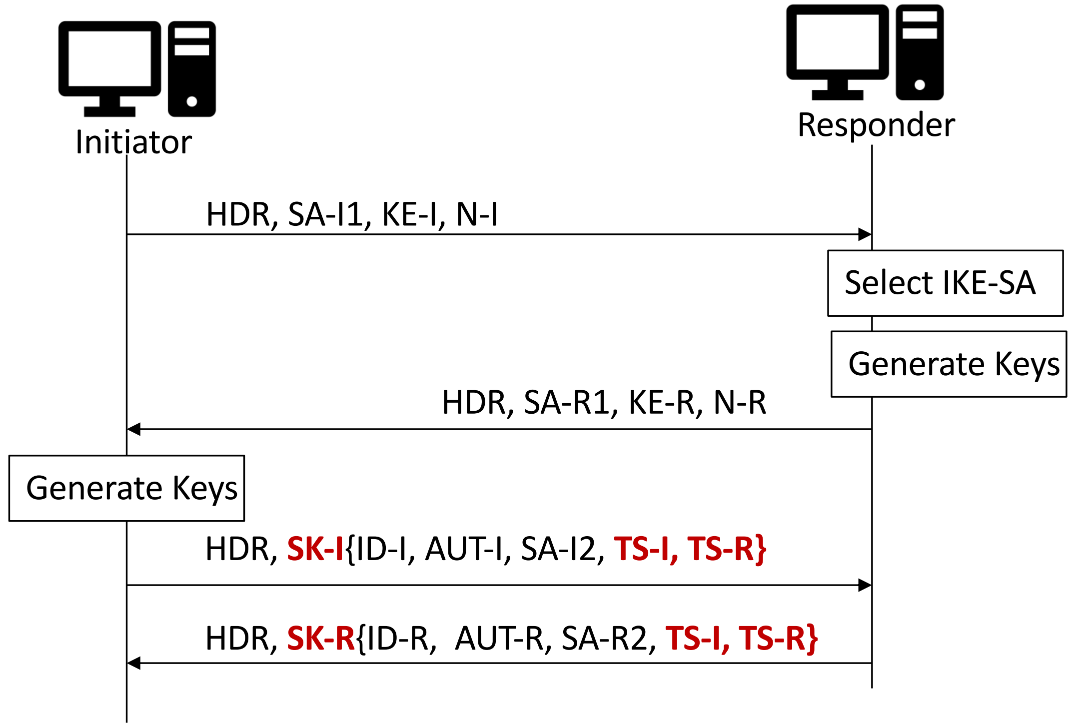
IKE v2 Exchange: IPSec-SA Negotiation

- Initiator sends proposals for IPSec-SAs in message 3
- Responder selects IPSec-SA, includes selection in message 4
- Proposal and selection protected against manipulation with integrity protection (and encryption) by **SK-I** and **SK-R**



IKE v2 Exchange: Negotiation of Traffic Selectors

- Initiator sends proposals for Traffic Selectors in message 3
- Responder includes selected Traffic Selectors in message 4
- Proposal and selection protected against manipulation with **SK-I** and **SK-R**



Traffic Selectors and Security Policy Database

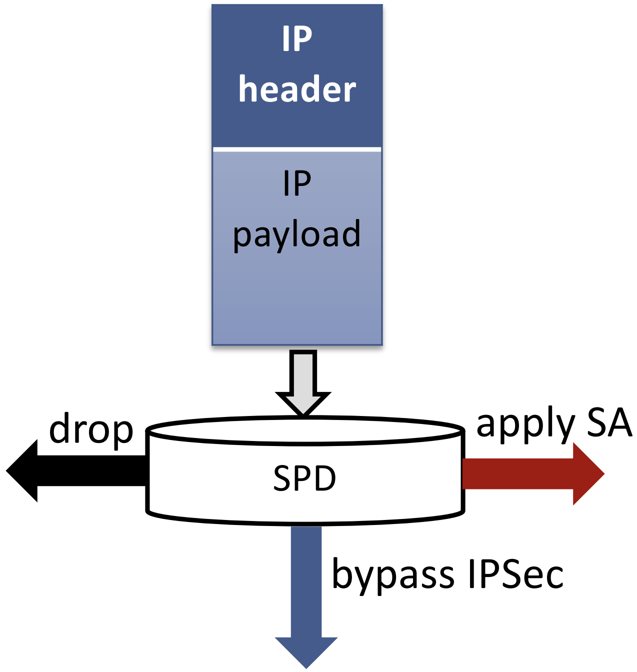
Traffic selectors are stored in a Security Policy Database

Traffic selectors specify

- Set of source IP addresses (one, list, range, wildcard)
- Set of destination IP addresses (one, list, range, wildcard)
- Transport layer protocol number (one, list, range, wildcard)
- Source and destination port (one, list, or wildcard)

Traffic selectors determine

- Whether inbound and outbound IP packets are protected, bypassed, or dropped
- If packet is to be protected, corresponding traffic selector points to the SA to use, if non exists yet, a new one is generated with IKE



Supported Algorithms

Encryption algorithms currently recommended for IKEv2 (RFC 8247)

- ENCR_AES_CBC MUST
- ENCR_AES_CCM SHOULD (supports integrity protection simultaneously)
- ENCR_AES_GCM SHOULD (supports integrity protection simultaneously)
- ENCR_CHACHA20_POLY1305 SHOULD (supports integrity protection simultaneously)

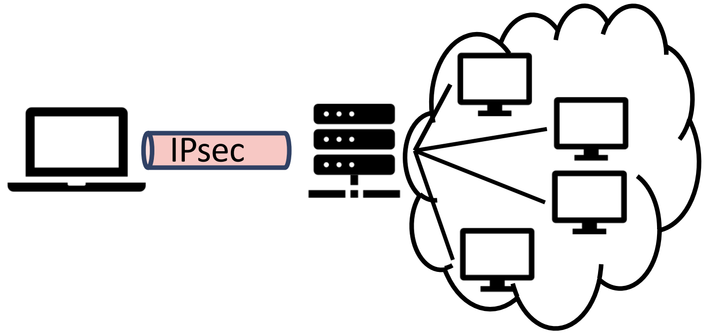
Integrity protection algorithms currently recommended for IKEv2 (RFC8247)

- AUTH_HMAC_SHA2_512_256 SHOULD
- AUTH_HMAC_SHA2_256_128 MUST

Recommendations change over time!

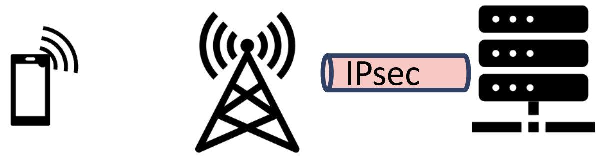
Examples of where else IPsec is used today

- **Many VPNs use IPsec between the VPN Client and Server**
  - Including the Cisco AnyConnect VPN Client used by RWTH



- **Connections between WLAN access points and authentication servers**
  - E.g., in Eduroam IPSec is used to protect the transfer of session keys from the authentication server to the WLAN access point

- **Connections between backbone components in mobile systems**
  - E.g., between base stations and backbone components or between backbone components that exchange subscriber information



Overview

IPSec

- Main use case
- Security services offered
- Authentication and key agreement
- Payload or packet protection

TLS 1.3

- Main use case
- Security services offered
- Handshake Protocol
- Payload protection with record protocol

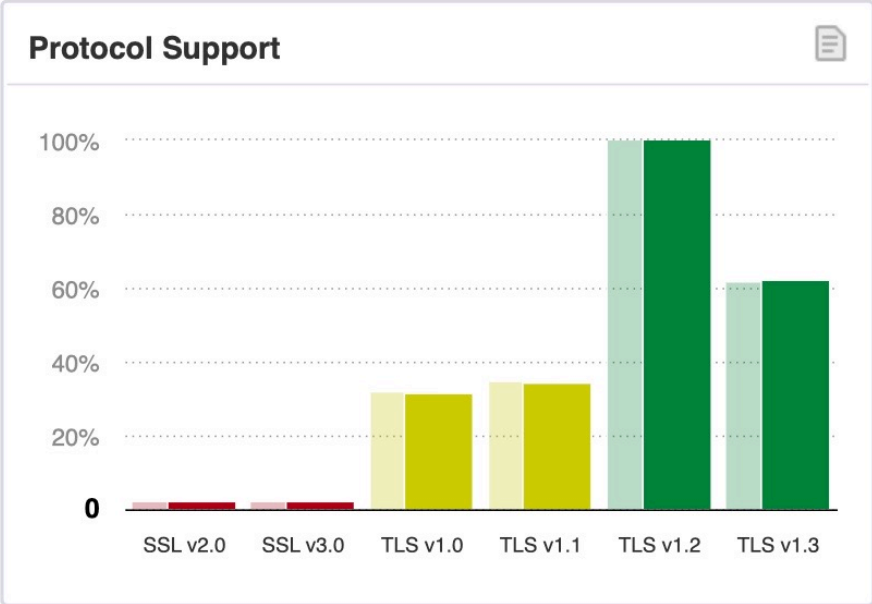
Comparison of the protocols

- Differences
- Communalities in mechanisms used
- Overlaps in use cases

Transport Layer Security Protocols over the Years

- **Secure Socket Layer SSL**
  - Predecessor of TLS, first version developed by Netscape in 1994
- **Transport Layer Security TLS**
  - Standardized by the IETF
  - TLS 1.0 and TLS 1.1 should not be used any more
  - TLS 1.2 still in use but has many weakness and only very few unbroken configurations
- **TLS 1.3 standardized in RFC 8446 in 2018**
- **We focus on TLS 1.3**

TLS version support of the top 150 000 visited websites according to the Alexa list (May 2023)

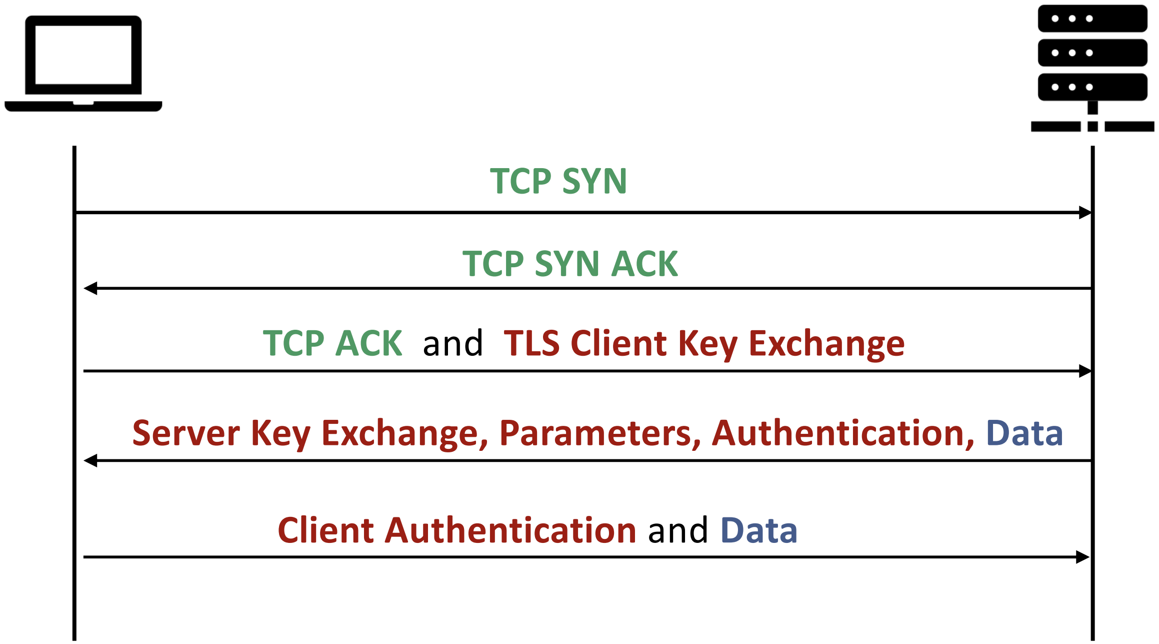


https://www.ssllabs.com/ssl-pulse/

Primary Use Case of TLS

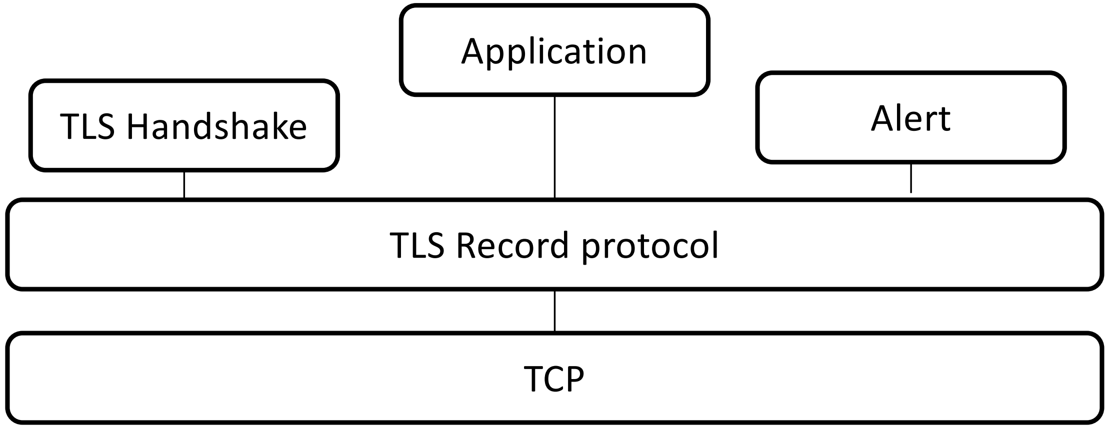
- **Transport layer protection of application traffic between a client and a server**
- **Most important use case**
  - HTTP over TLS = HTTPs
- **Other uses include**
  - SMPT over TLS = SMTPs
  - DNS over TLS = DoT

Application (HTTP,…)
TLS
TCP
IP
Data Link
Physical



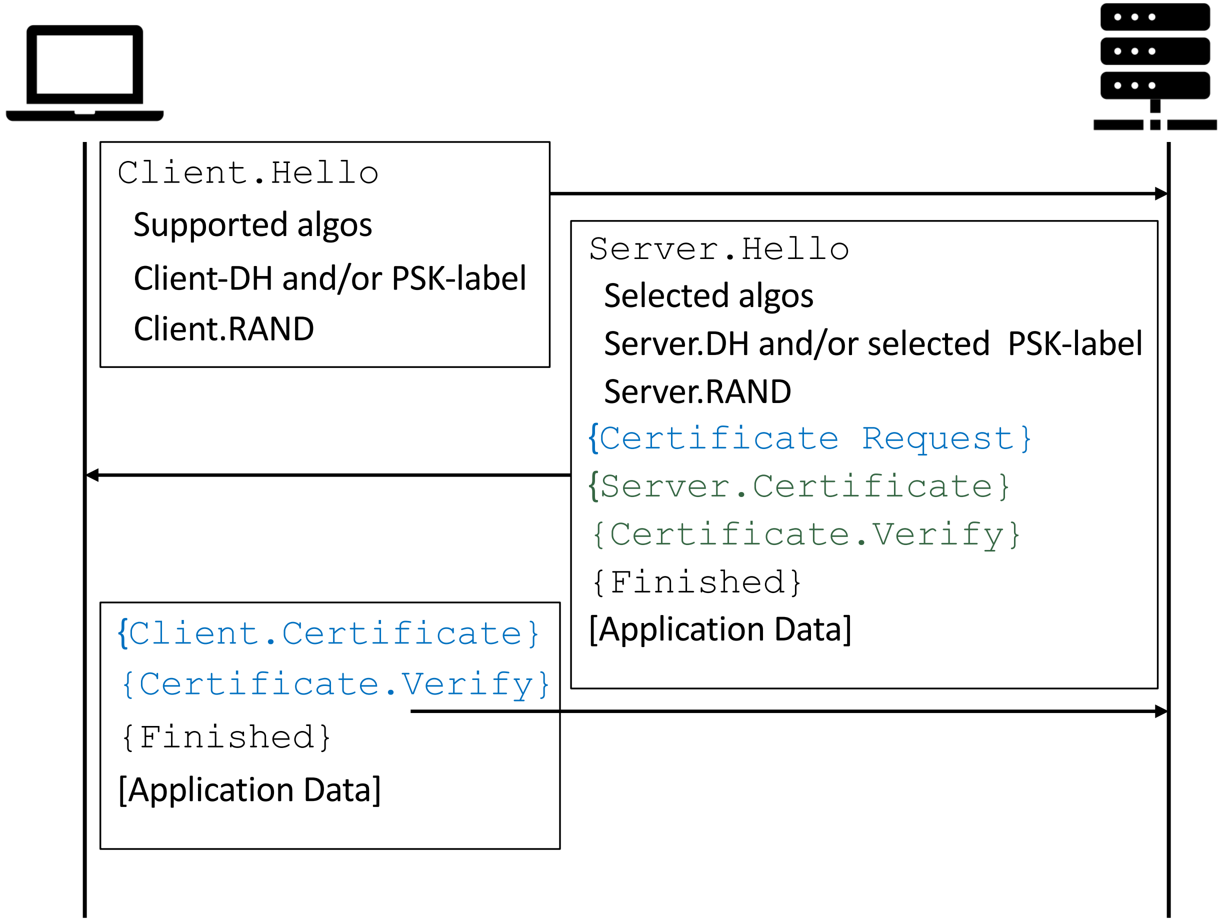
Security Services offered by TLS 1.3

- **Authenticated session key agreement**
  - Using the **TLS Handshake protocol**
  - Supports three key agreement methods
    - PSK-only
    - PSK-authenticated DH
    - Signature authenticated DH
- **Encryption and integrity protection**
  - Of application data, part of the handshake, alert and change cipher spec messages
  - Using the **TLS Record Protocol**



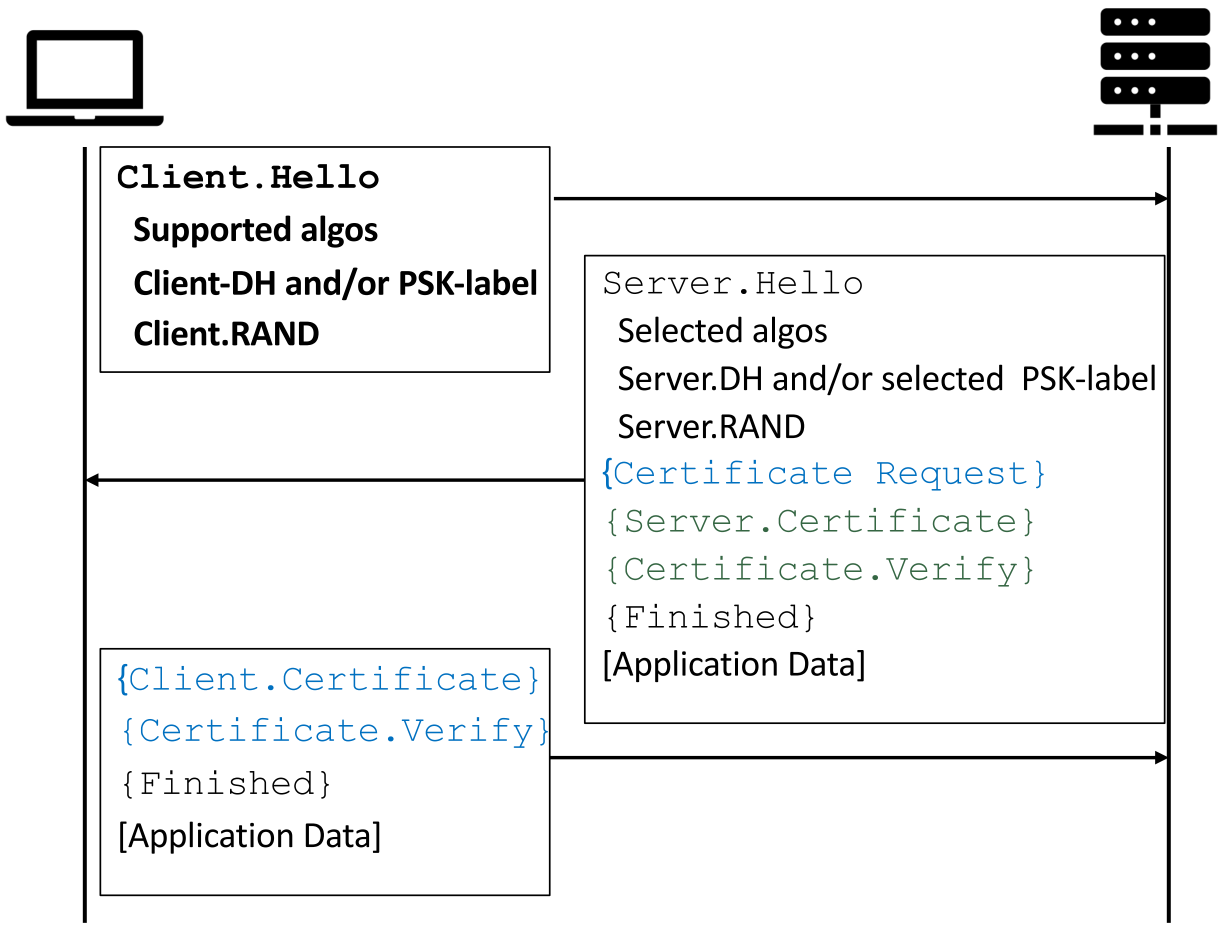
Note: we focus on **TLS 1.3** here
Most resources on the Web are still on TLS 1.2

Authentication and Key Agreement: TLS 1.3 Handshake Overview



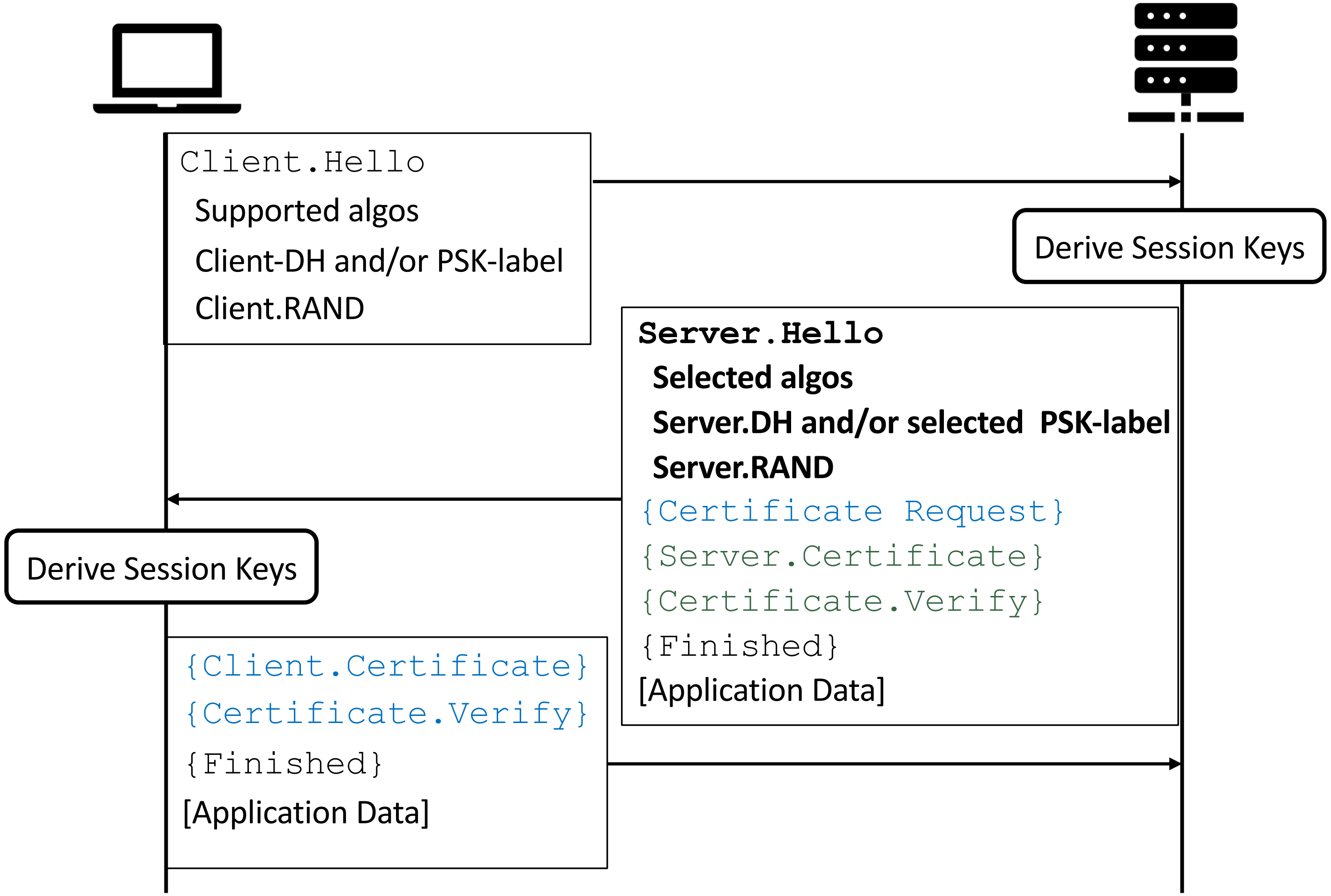
- **{} encrypted and integrity protected handshake messages**
- **[] encrypted and integrity protected application data (different keys used)**
- **Only sent if certificate-based client authentication required**
- **Only sent if DH is authenticated with server signature**

Authentication and Key Agreement: TLS Handshake Key Exchange Phase (1)



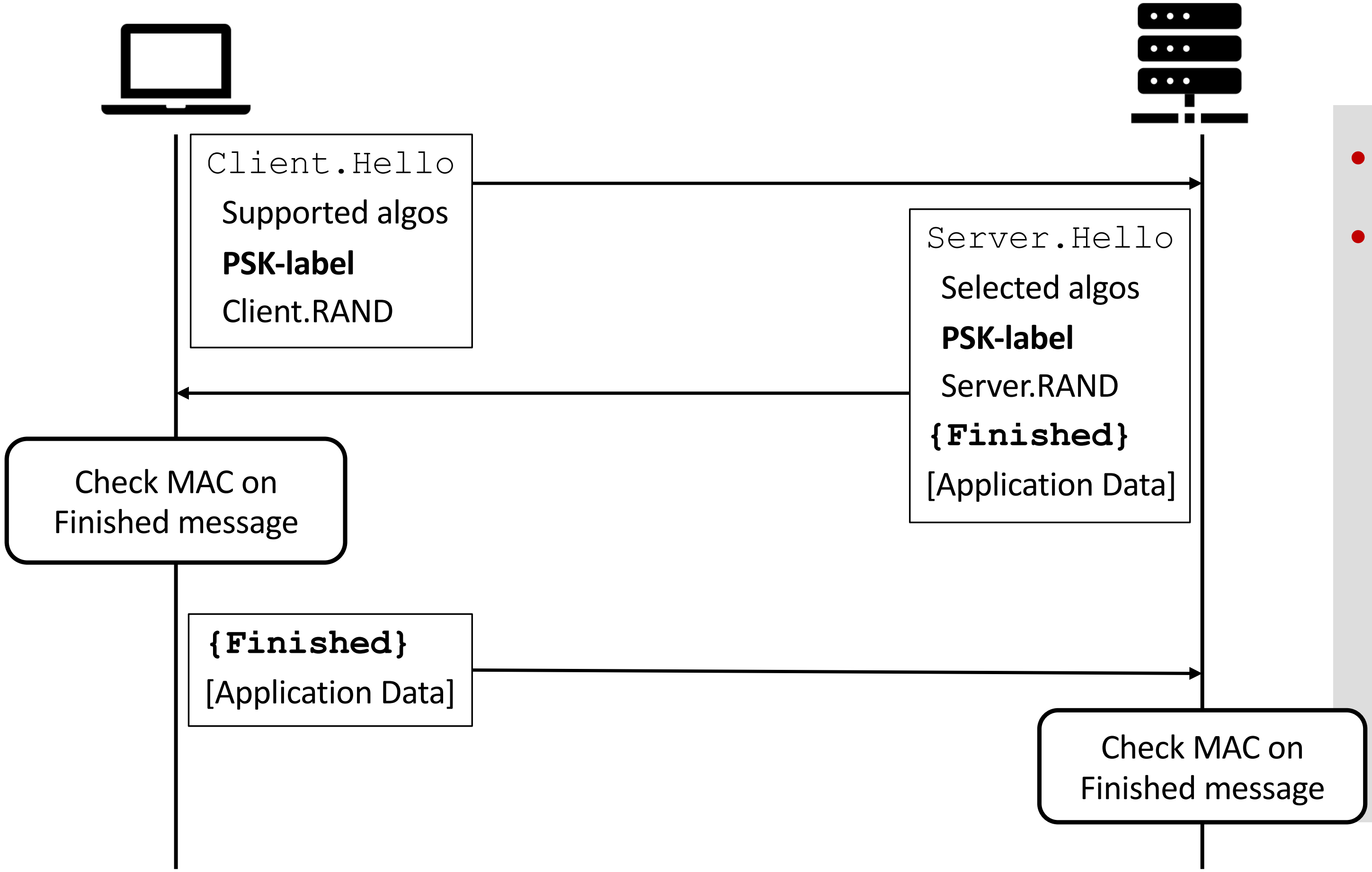
- **In Client.Hello, client offers**
  - several DH-values for several groups and/or
  - several PSK-labels identifying PSKs
  - Encryption and integrity protection algorithms it supports
- **and includes**
  - a fresh random number Client.RAND

Authentication and Key Agreement: TLS Handshake Key Exchange Phase (2)



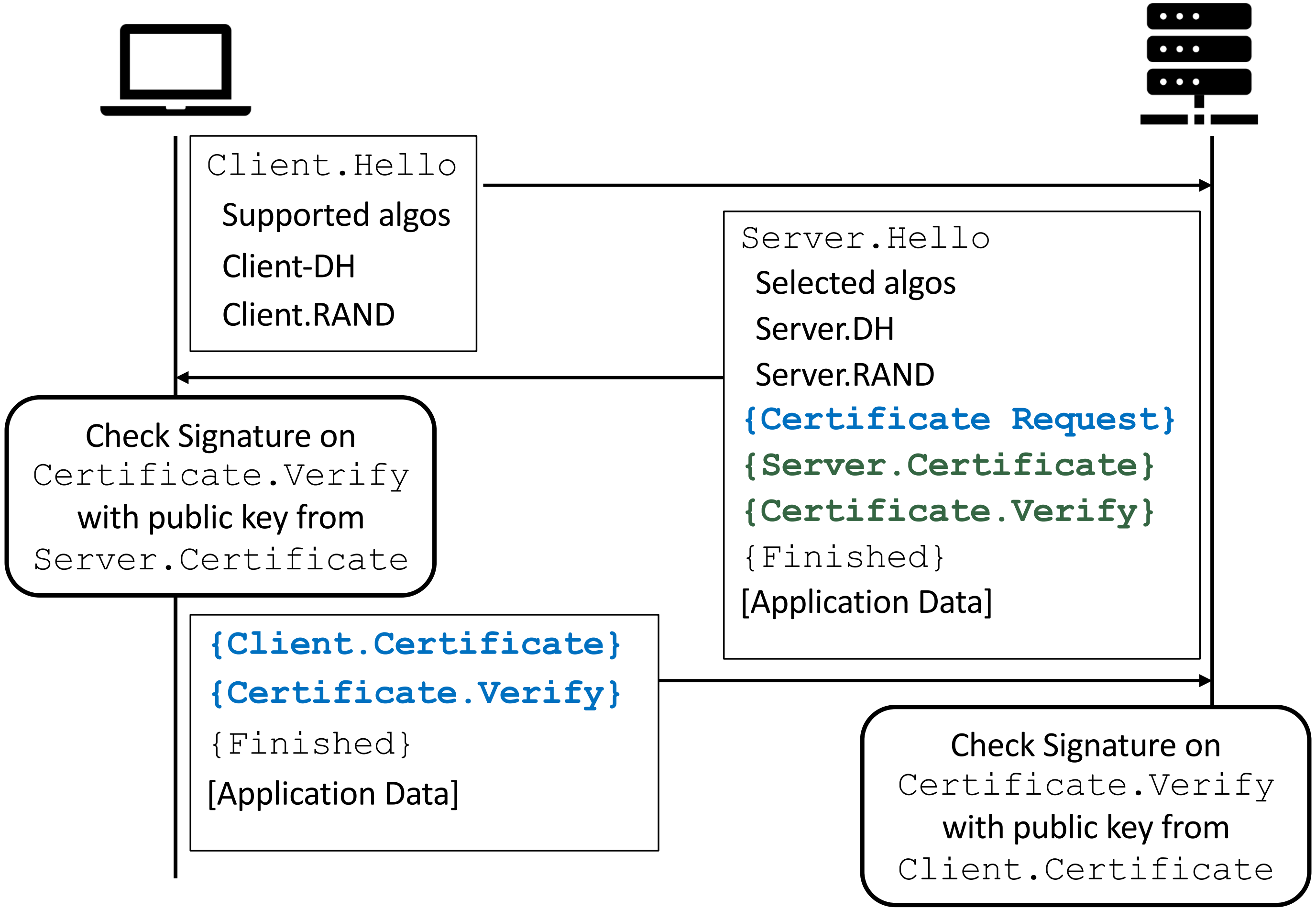
- **In Sever.Hello, server includes**
  - DH-values for the selected group and/or
  - PSK-label of selected PSK
  - Selected enc. and int. algos
- **Client and Server can now**
  - Compute the DH-Key and/or
  - Identify and retrieve the PSK to use
  - Derive session keys from DH-Key and / or PSK

TLS 1.3 Handshake PSK-Only Key Exchange



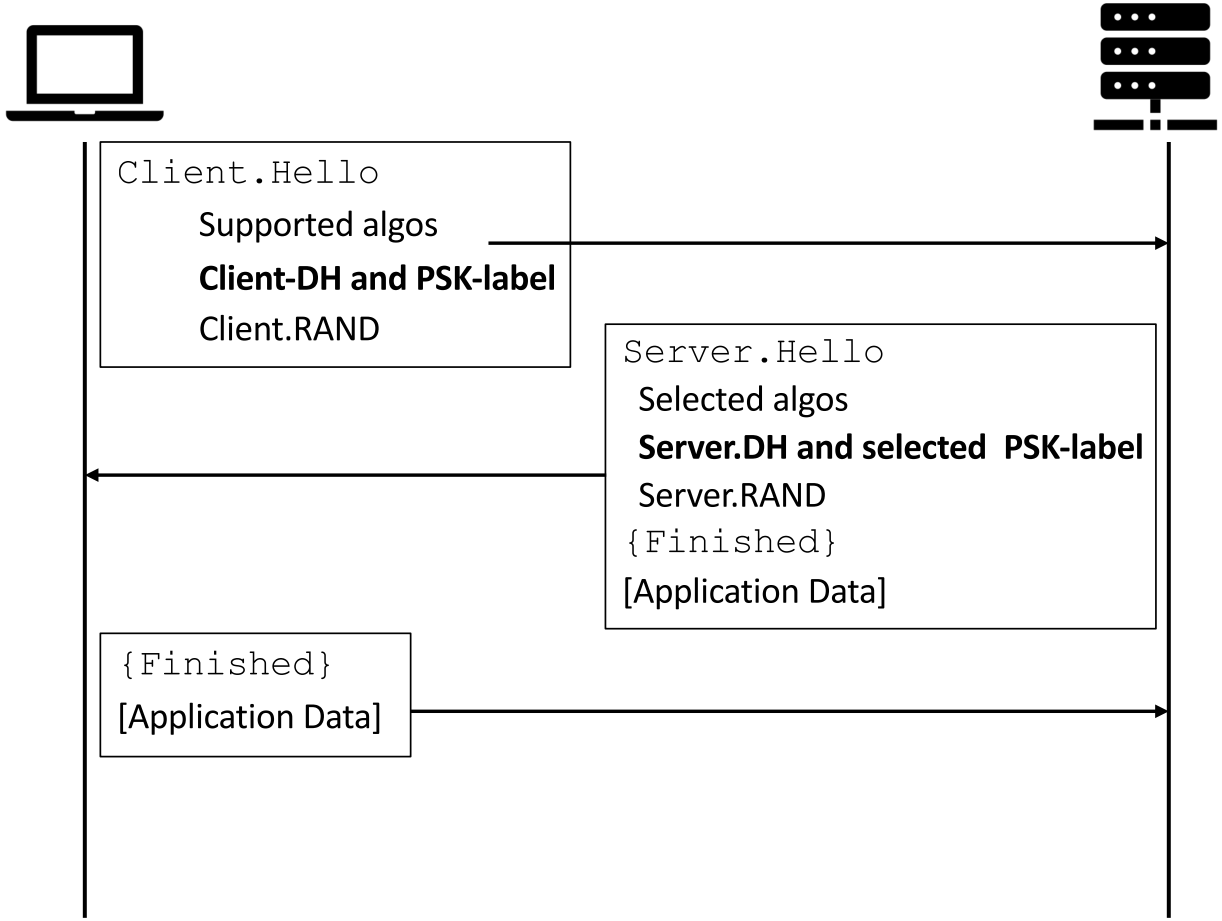
- **PSK-label identifies PSK to use**
- **Each Finished message**
  - Includes MAC on hash of all handshake messages so far
  - MAC is computed with session key derived in last step
  - MAC ensures that any changes to data in the Hello messages are protected against manipulation

TLS 1.3 Handshake DHE (with signatures) Key Exchange



- **Server sends**
  - Certificate request if client is to be authenticated with a certificate
  - its own Server.Certificate including a chain of certificates
  - Certificate.Verify message with a signature on the hash of all handshake messages with server's private key
  - Finished message as before
- **If requested Client sends**
  - Client.Certificate and Certificate.Verify

TLS1.3 Handshake PSK with DHE Key Exchange

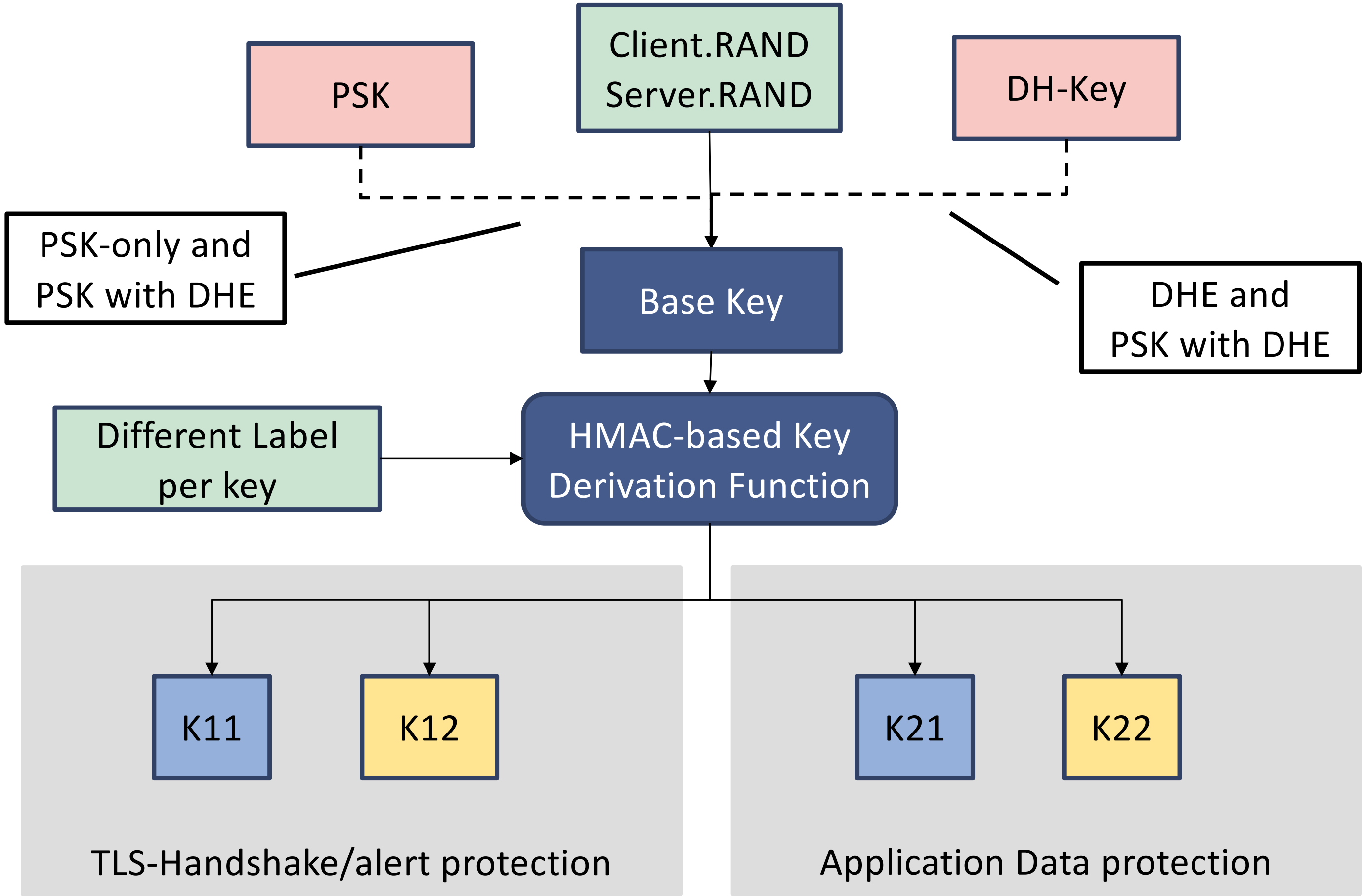


- **Same as PSK-only**
- **But session keys derived from PSK and DH-key**

PSK-only and PSK with DHE-key

- can also be used after a full handshake with DHE and signatures to resume
- In this case, the first message from the client may already contain data
  - Referred to as 0-RTT

Session Key Generation



Separate Keys for the different directions

- Counters, IV etc. can be selected independently

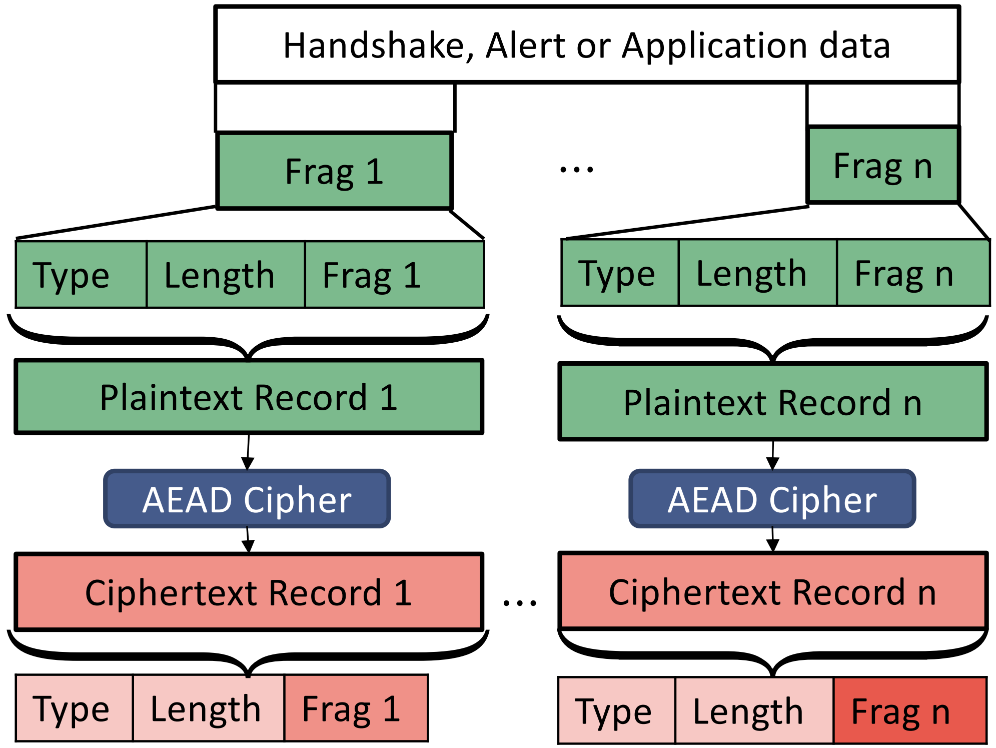
Client to Server Keys

Server to Client Keys

Kxy: Session key for AEAD ciphers

TCP Payload Protection with the TLS Record Protocol

- **The record protocol is responsible for**
  - Taking messages to be transmitted and fragmenting data into blocks of 2^{14} bytes or less
    - Called TLS Plaintext records
  - Protecting the records and transmitting them
  - Verifying integrity protection on received data, decrypting received data
  - Reassembling and delivering data to higher layers
- **Supports three main content types for the plaintext records**
  - handshake , application-data, alert



Alert protocol

- **Specifies two different types of alerts**
  - Closure alerts
    - Closure-notify: Notifies receiver that sender will close connection now, receiver should ignore any traffic received after this message
    - user-canceled
  - Error alerts
    - unexpected-message
    - bad-record-mac: MAC on record layer did not check out correctly
    - handshake-failure: parameters could not be agreed upon
    - ....

TLS and Certificate Validation

- **The TLS RFC itself only specifies that**
  - TLS servers and clients need to check that the signature provided in the Certificate.Verify message can be verified with the public key in the certificate
- **Verifying the certificates received additionally requires the receiver to**
  - Check if the root CA is trusted in the context of the application invoking TLS
  - Check that the identity included in the certificate corresponds to the identity of the server
  - Verifying the signatures on all certificates in the provided chain up to a trusted root certificate
  - Verifying that each certificate in the chain is currently valid and has not been revoked

Supported Algorithms in Handshake and Data Protection

- **All Ciphers supported by TLS 1.3 are AEAD ciphers**

Supported AEAD Ciphers
TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_SHA256
TLS_AES_128_CCM_SHA256
TLS_AES_128_CCM_8_SHA256

Overview

IPSec

- Main use case
- Security services offered
- Authentication and key agreement
- Payload or packet protection

TLS

- Main use case
- Security services offered
- Authentication and key agreement
- Payload protection

Comparison of the protocols

- Differences
- Communalities in mechanisms used
- Overlaps in use cases

Comparison of IPSec and TLS

IPSec	TLS
IP-packet level protection	Protection of TCP Segments
Host-to-host protection of IP communication	Transport layer protection invoked by a specific application
Application independent protection of communication between individual hosts or complete networks	Communication between browser and web server and other client/server-style applications
Can be transparent to end users; no need to understand / configure IPsec	Requires end users to check if certificate has been issued to desired server
Highly configurable; Can be restricted to protect IP packets to / from individual host as well as complete networks	Invoked by a specific application running between client and server for all traffic of this application
Authentication and key agreement based on two-sided authenticated Diffie-Hellman	Authentication and key agreement based on a server-side only or mutually authenticated Diffie-Hellman
Authentication can be based on secret keys or public/private key pairs	Authentication based on public / private key pair of server and optional public / private key pair of client, alternatively a pre-shared secret key can be used since TLS 1.3

Base Specifications and References

IPSec

- Internet Key Exchange Protocol IKEv2
  - Specified in RFC RFC 7296
- Security Architecture for IP
  - Specified in RFC 4301
- Encapsulating Security Payload Protocol ESP
  - Specified in RFC 4303
- Authentication Header Protocol AH
  - Specified in RFC 4302

TLS 1.3

- TLS 1.3 RFC 8446
- Includes the handshake, record layer, and alert protocols

Book Chapter

- W. Stallings, Cryptography and Network Security: Principles and Practice, 8th edition, Pearson 2022
  - Chapter 17: Transport-Level Security
  - Chapter 20: IP Security

Summary

- **IPSec offers encryption and integrity protection for IP packets**
- **IPSec supports two modes**
  - Transport mode for IP-packet protection directly between packet origin and final destination
  - Tunnel mode for protection of IP-packets involving intermediate nodes such as security gateways
- **IPSec comprises**
  - The ESP protocol for encryption and integrity protection of the payload of the protected packet
  - The AH protocol for integrity protection of the entire protected packet (including the header)
- **IKEv2 offers authentication and key agreement for IPSec**
  - Based on a secure authenticated Diffie-Hellman key exchange (provides key confirmation)
  - Key exchange can be authenticated with the help of signatures or message authentication codes
  - Also negotiates which traffic is going to be protected with which protocols and algorithms

Summary

- **TLS 1.3 offers**
  - Server-side or mutual authentication between client and server
  - Session key establishment
  - Encryption and integrity protection of TCP segments
- **Handshake protocol in TLS 1.3**
  - Based on ephemeral DH exchange and signatures
  - Based on a pre-shared key alone
  - Based on ephemeral DH and pre-shared-key
- **Record protocol in TLS 1.3**
  - Supports only AEAD ciphers