Dr. Ammar Haider
Assistant Professor
School of Computing

CS3002 Information Security

Security in TCP/IP

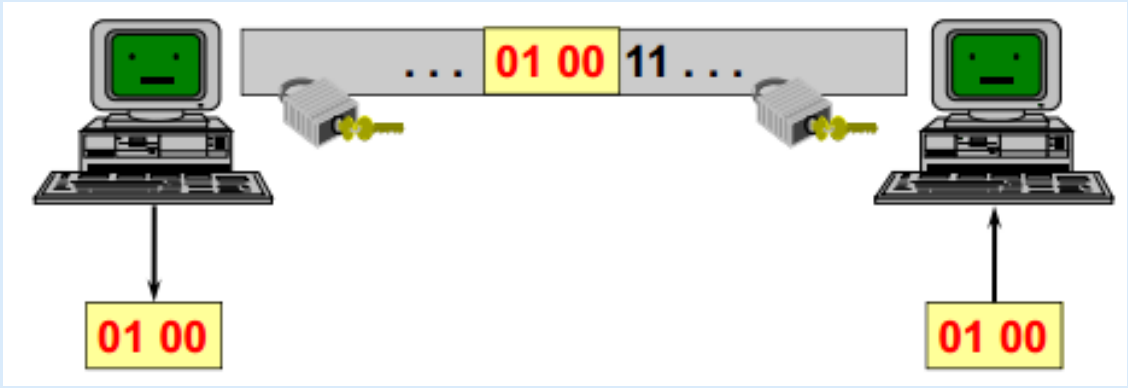
Reference: Stallings SPP chap 22

Implementation options

- Channel Security
- Message/Data Security
  - Security internal to applications
  - Security external to applications

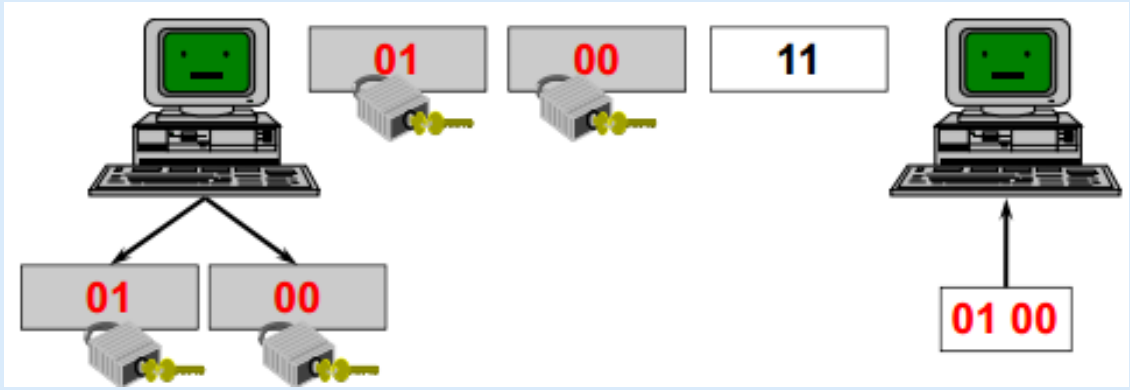
Channel Security

- authentication, integrity and secrecy only during the transit inside the communication channel
- requires no (or small) modification of applications



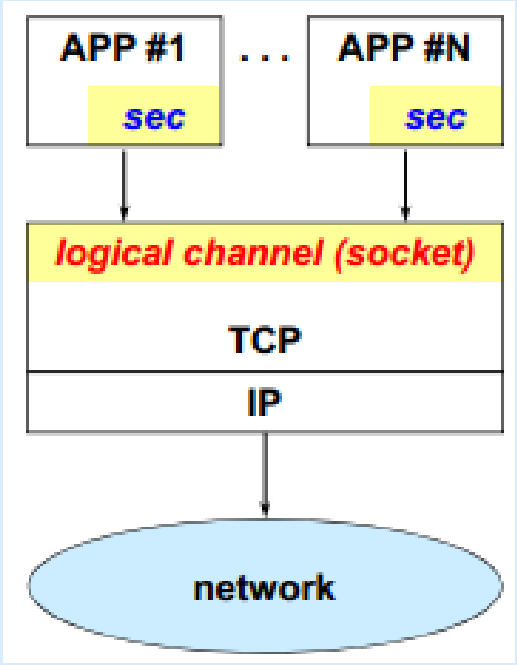
Message/data security

- authentication, integrity and secrecy self contained in the message
- requires modification of applications



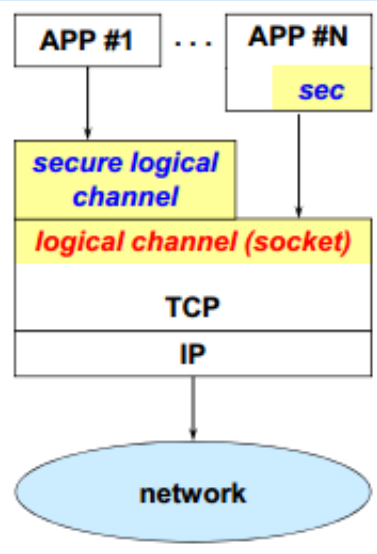
Security internal to applications

- each application implements security internally
- the common part is limited to the communication channels (socket)
- possible implementation errors (inventing security protocols is not simple!)
- does not guarantee interoperability



Security external to applications

- the session level would be the ideal one to be used to implement many security functions
- … but it does not exist in TCP/IP!
- a "secure session" level was proposed:
  - it simplifies the work of application developers
  - it avoids implementation errors
  - it is up to the application to use it (or not)



SSL
Secure Sockets Layer

and

TLS
Transport Layer Security

SSL/TLS Timeline

SSL was a commercial protocol, it evolved into TLS, an IETF standard.

Version	Published	Deprecated
SSL 1.0	Never published	
SSL 2.0	1995	2011
SSL 3.0	1996	2015
TLS 1.0	1999	2021
TLS 1.1	2006	2021
TLS 1.2	2008	in use
TLS 1.3	2018	in use

now considered insecure, should no longer be used

SSL: what is it?

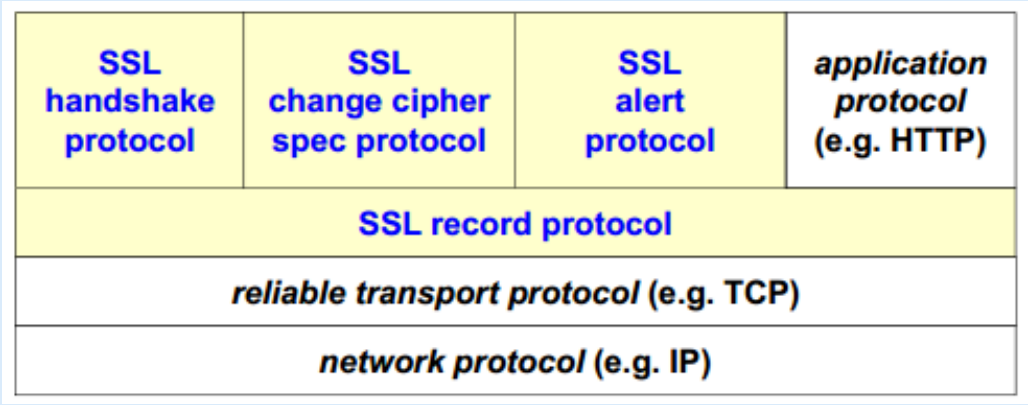
- **S**ecure **S**ockets **L**ayer: Security between layer 4 (transport) and layer 5
- secure transport channel (session level):
  - peer authentication (server only, or server + client)
  - message confidentiality
  - message authentication and integrity
  - protection against replay attacks
- Easily applicable to all protocols based on TCP:
  - HTTP, SMTP, FTP, TELNET, ...
  - e.g. the famous secure HTTP (https://....) = 443/TCP

SSL: what is it?

- Philosophy: Easier to deploy something if no changes in OS required
- Application's API (Socket) is interface to SSL
  - hence secure socket layer
- API to SSL is the superset of API to TCP
- SSL/TLS operate above TCP. OS doesn't change applications do!

7	Application Layer
Secure Sockets Layer	
4	Transport Layer
3	Network Layer
2	Data Link Layer
1	Physical Layer

SSL/TLS Protocol Stack

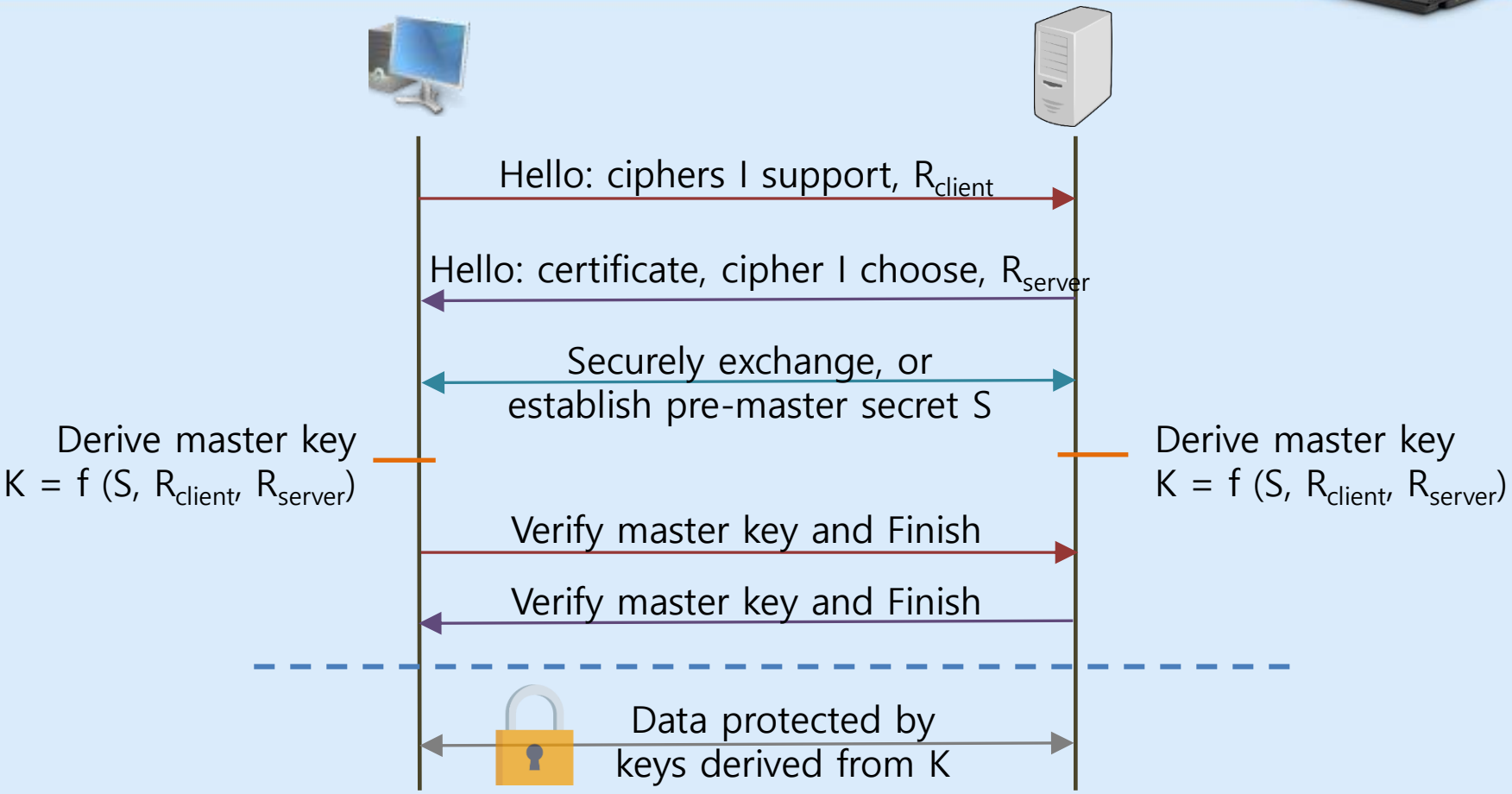


- Handshake: enables the client and server to negotiate cipher algorithms and establish the secret keys with which they will communicate
- Change-cipher-spec: indicates the confirmation of secret key usage for data communication
- Alert: signal problems with SSL connection, give current status
- Record protocol: securely carries the data of higher layers

SSL/TLS Handshake

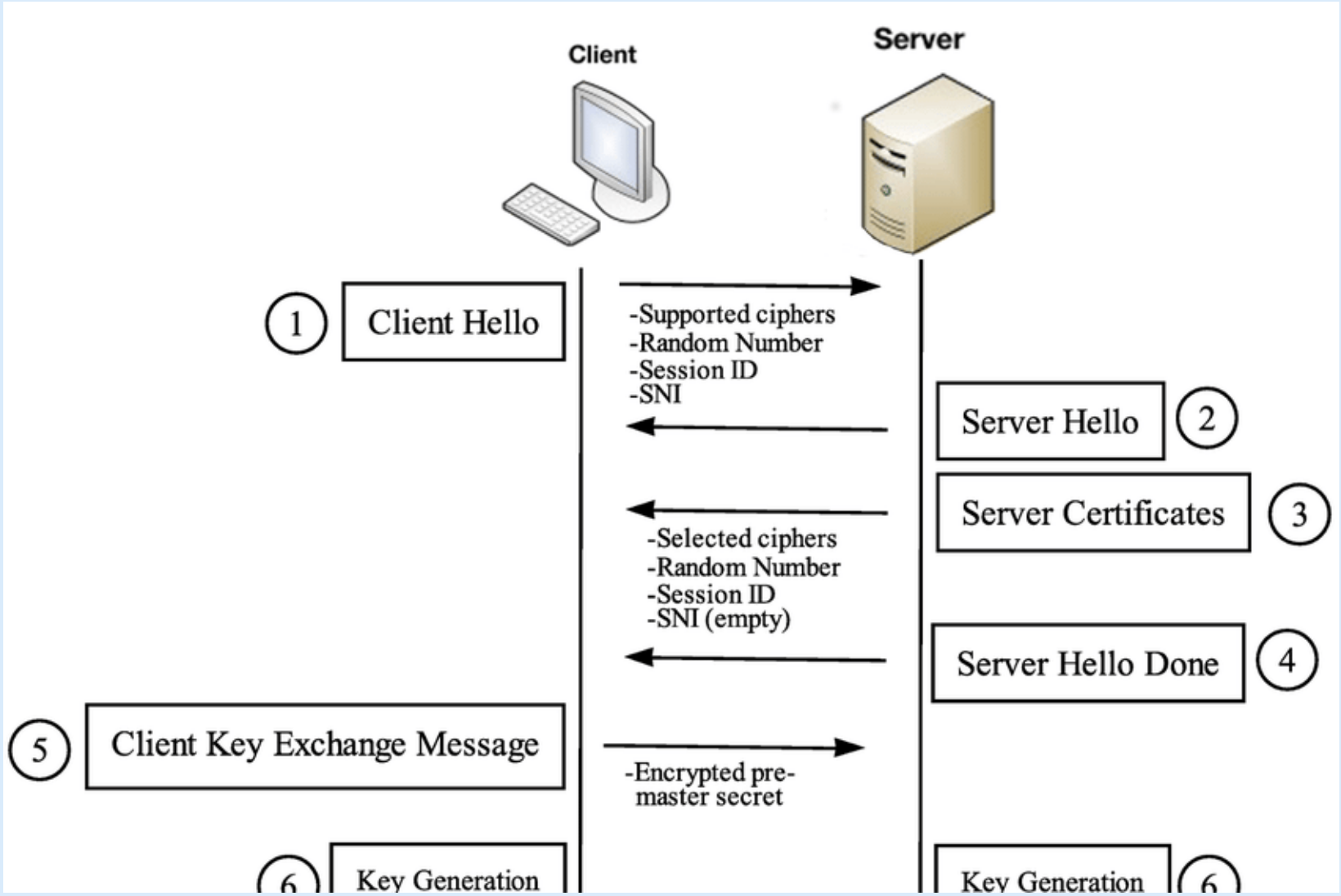
- agree on a set of algorithms for confidentiality, integrity and authentication
- exchange random numbers between the client and the server to be used for the subsequent generation of the keys
- establish a symmetric key by means of public key (asymmetric) operations, e.g. RSA, Diffie Hellman
- negotiate the session-id
- exchange the necessary certificates

Handshake in brief

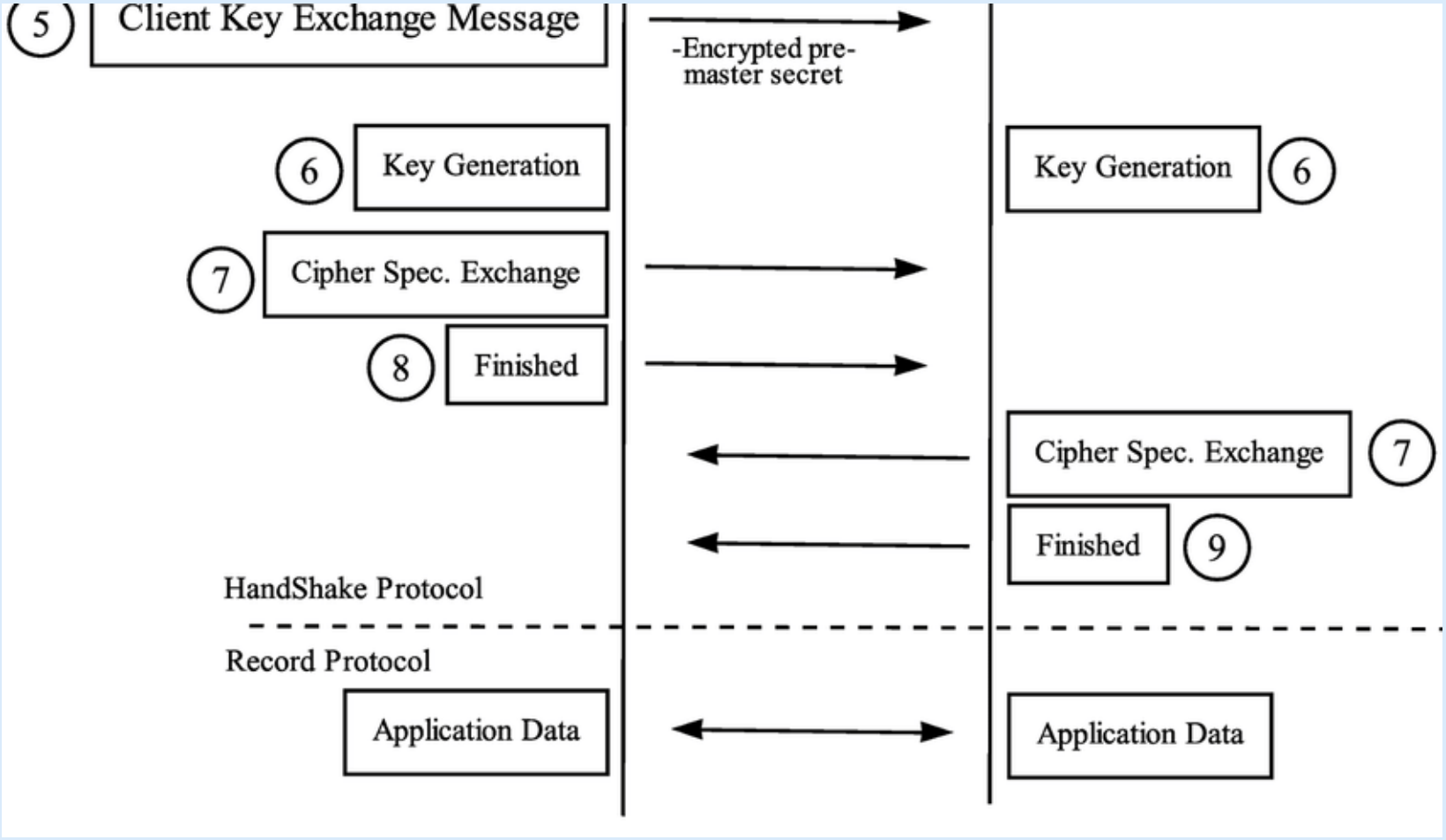


- Secrets are: Pre-master secret S, Master Key K
- Server authentication via certificate, client authentication optional

Handshake in detail



Handshake in detail



Key Handshake Terms

- HELLO Extensions: request extended functionality by sending data in the extensions field.
  - e.g. max_fragment_length, status request
  - The server may not oblige
  - Client may abort the handshake
- Pre-master secret, S:
  - EITHER generated by client and sent to server using RSA encryption
  - OR derived via a key exchange algorithm e.g. Diffie Hellman (g^{ab} mod p)

Key Handshake Terms

- Master key:
  generated by applying a Pseudo-Random Function (PRF).

$$\text{Master key} = \text{PRF}(S, R_{client}, R_{server})$$

- Master key may also be the called the session key.
- Subsequently, multiple cryptographic operations (like encryption and MAC calculation) will be applied on the application data. Keys for all these operations are derived from the session key.

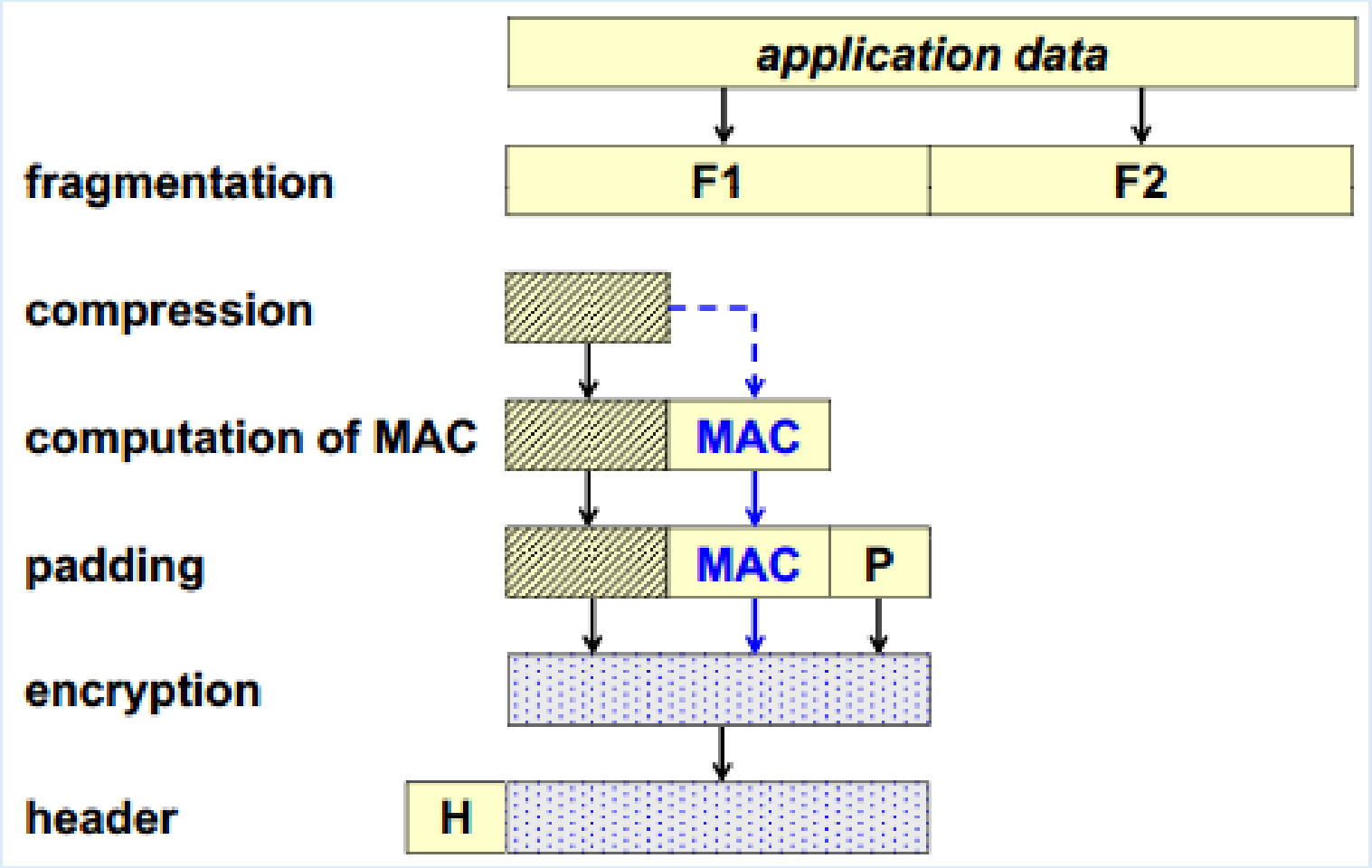
Pre-Master vs Master

Why not use pre-master secret (S) as the session key?

- There are various reasons, but most importantly, it is for prevention of replay attacks.
- An attacker could intercept client's packets, record them, and send to server again at a later time.
  - Server needs to be able to detect such attempts.
- Including unique random values from both parties (R_{client}, R_{server}) helps prevent these attacks.
  - Even if an attacker intercepts and tries to reuse a valid handshake request, the server's random values in a new session would be different.

https://security.stackexchange.com/q/218491/245722

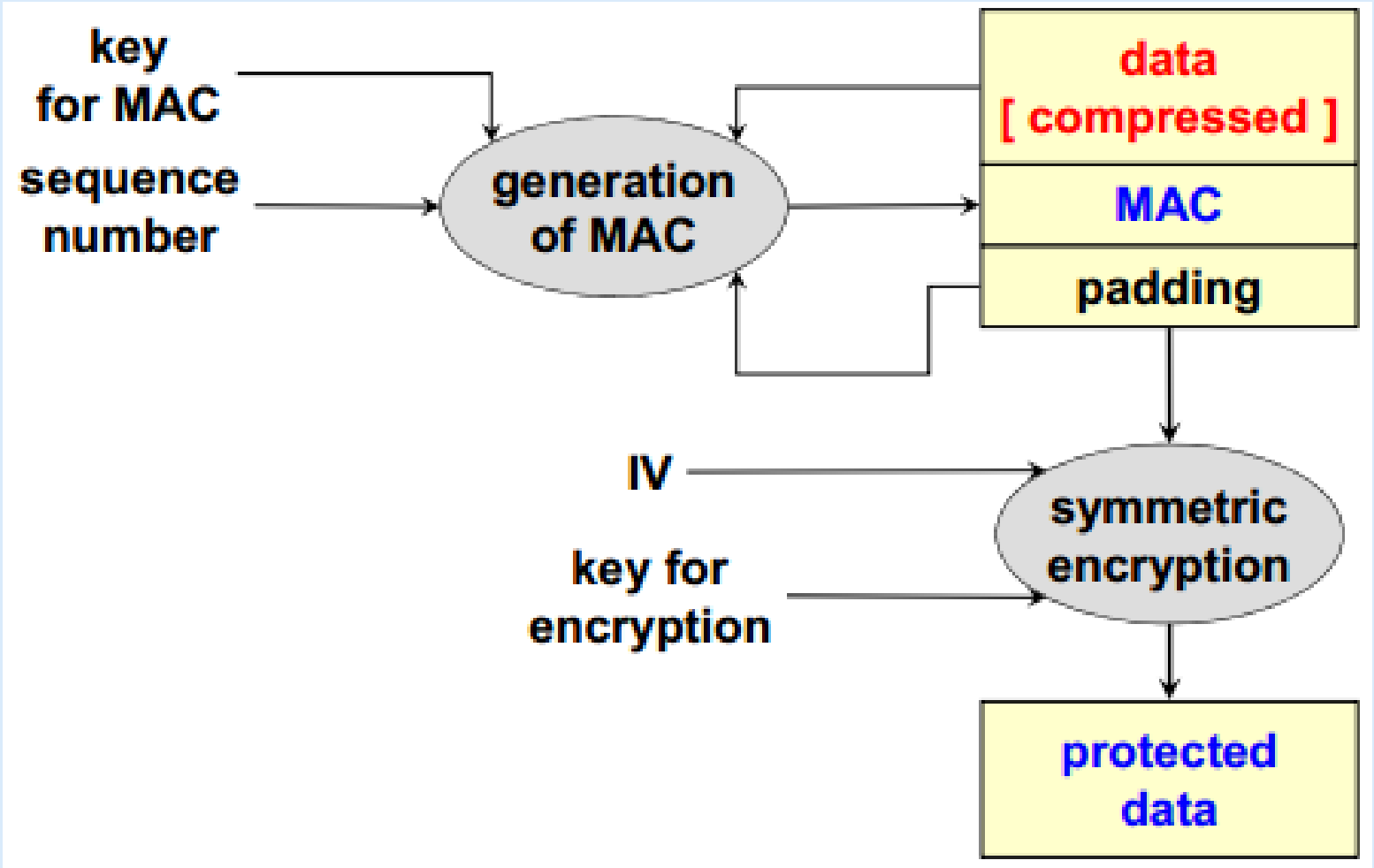
SSL/TLS record protocol



MAC computation

- MAC = message_digest (key, seq_number | type | version | length | fragment)
- message_digest: the chosen hash algorithm
- key: sender-write-key or receiver-read-key
- seq_number: 32-bit integer
- type: type of record
  - change cipher spec (20)
  - alert (21)
  - handshake (22)
  - application data (23)
- length: length of the fragment/plaintext

Record Protocol Workflow



TLS 1.3 important changes

- Removed unnecessary features like compression
- Removed ability to ‘renegotiate’ cipher specs
- Removed broken hash functions like MD5 and SHA-224
- Mandated Diffie-Hellman for key exchange (replacing RSA) in order to ensure forward secrecy
- Reordered and merged handshake messages for a faster negotiation. Whole handshake can be completed in 1 round trip.

TLS Demos – for reference

Comic Explanation
https://howhttps.works/the-handshake/

TLS 1.2
https://tls12.xargs.org/

TLS 1.3
https://tls13.xargs.org/

Handshake Analysis in Wireshark
https://www.youtube.com/watch?v=MQg48n9lV0s

IPsec
Internet Protocol Security

IPsec

- Security at layer 3
- IPsec ensures:
  - confidentiality, integrity, and authenticity
- Allows secure communication on the Internet
- Independent from the application or higher protocols
- Network-layer security instead of application-layer security
  - Compatible with schemes providing security at the application layer (can be applied simultaneously)

IPsec

- Philosophy: implementing security within the operating systems automatically causes applications to be protected without changing applications
- IPsec is within the OS.
  - OS changes, applications and API to TCP don't.

7	Application Layer
4	Transport Layer
3	Network Layer
	IPSec
2	Data Link Layer
1	Physical Layer

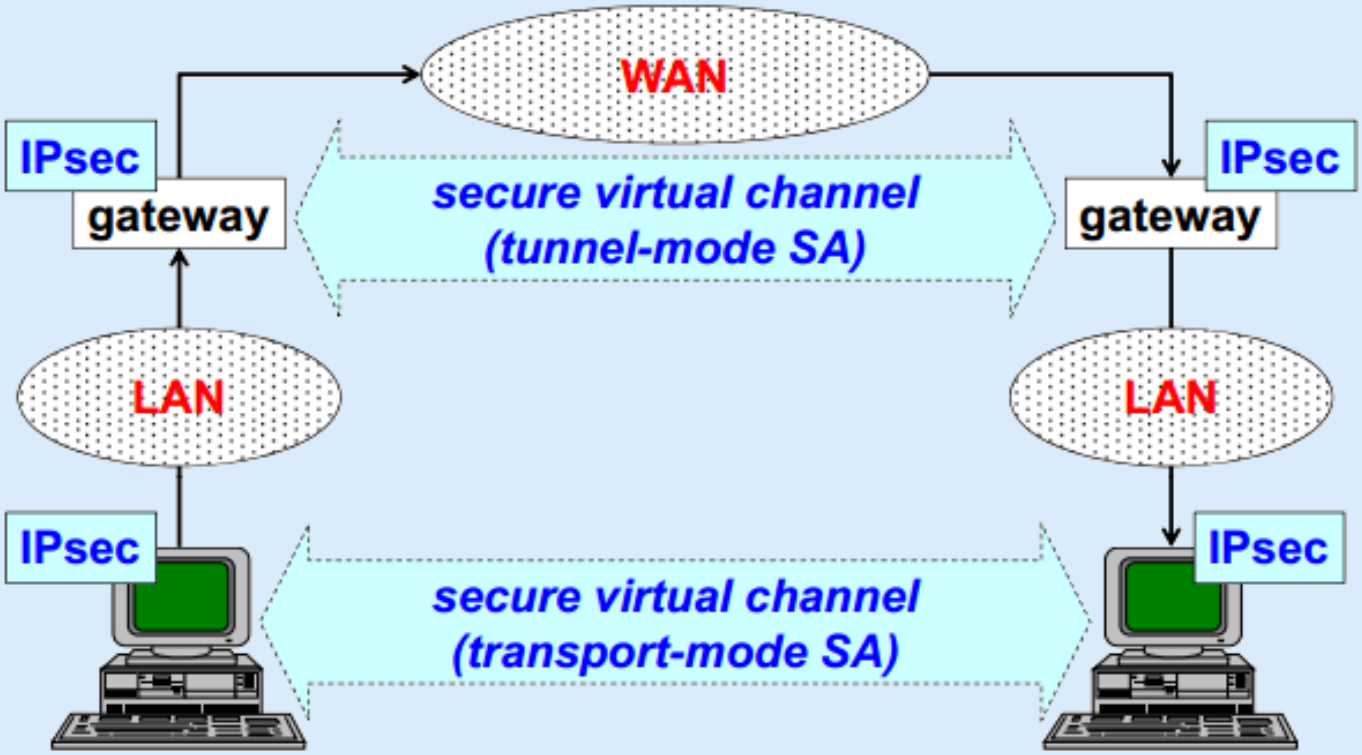
IPsec

- Further advantages:
  - Can be applied to all network traffic
  - Routers/firewalls vendors can implement it (can't implement SSL)
  - Transparent to the applications
  - Transparent to the users
- Limitations:
  - Limited to IP addresses
  - Has no concept of application users

Applications of IPsec

- Secure connection among different branches of the same company
  - Virtual Private Network (VPN)
- Secure remote access to an Intranet through the (insecure) Internet
  - Allows secure remote workers
- Secure communication between peers
- Secure inter-organizational communication
  - such as between customers and vendors/suppliers

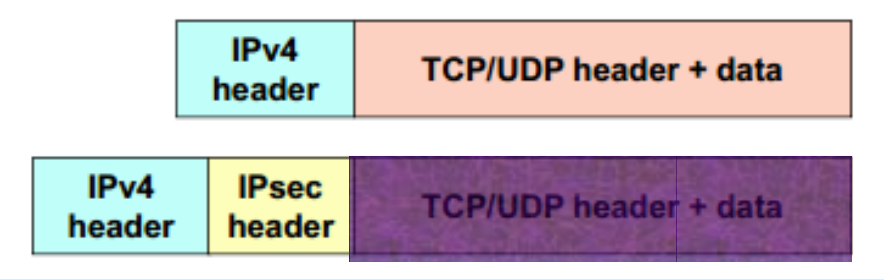
IPsec operating modes



Transport mode

- used for end-to-end security, that is used by hosts, not gateways (exception: traffic for the gateway itself e.g. SNMP, ICMP)

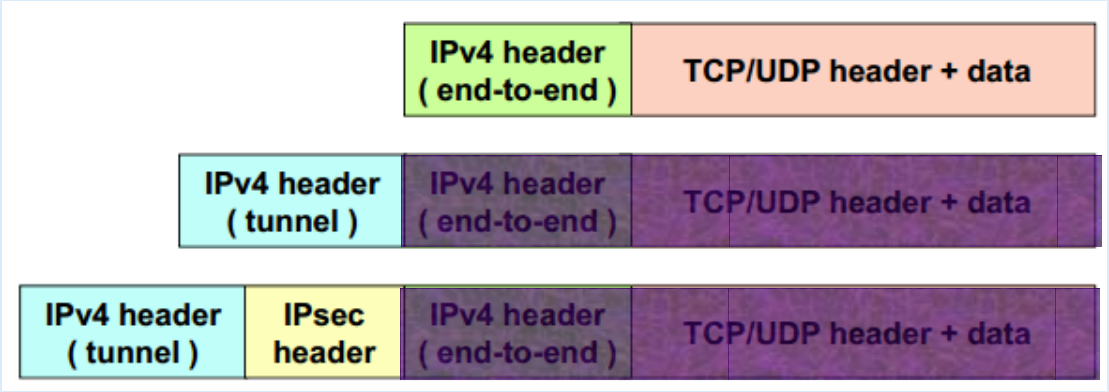
- pro: computationally light
- con: no protection of header fields



Tunnel Mode

- used to create a VPN, usually by gateways
  - gateway-to-gateway mode

- pro: protection of header variable fields
- con: computationally heavy



IPsec protocol suite

IPsec standard defines three distinct protocols:

- definition of two specific packet types:
  - AH (Authentication Header)
    - for integrity, authentication and anti-replay
  - ESP (Encapsulating Security Payload)
    - for confidentiality, integrity, authentication & anti-replay
- protocol for key exchange:
  - IKE (Internet Key Exchange)
  - manages secure key exchange using public key algorithms and digital certificates

Security Policies

- Rules to decide if an IP packet needs to be processed and how.

A policy defines

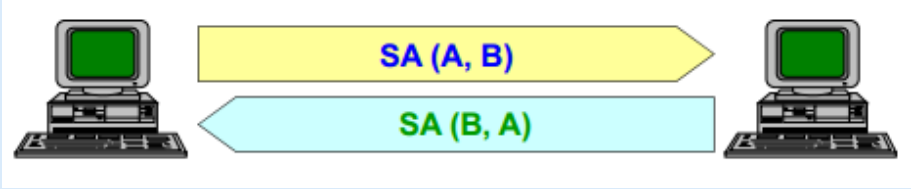
a) packet selection rules like source, destination

b) action to take (encrypt, authenticate, block)

c) IPsec protocol (ESP/AH) and mode (transport/tunnel)
   - e.g. all outbound packets from database server (say 192.168.5.1) to clients subnet (e.g. 192.168.12.0/24) should be authenticated by AH in transport mode
   - all traffic going to remote gateway (say 170.34.12.6) should be encrypted with ESP in tunnel mode

Security Associations

- A contract between two peers on security parameters (like keys, cipher algorithms, key expiry)
- Separate associations in each direction
  - two SA are needed to get complete protection of a bidirectional packet flow in IPsec

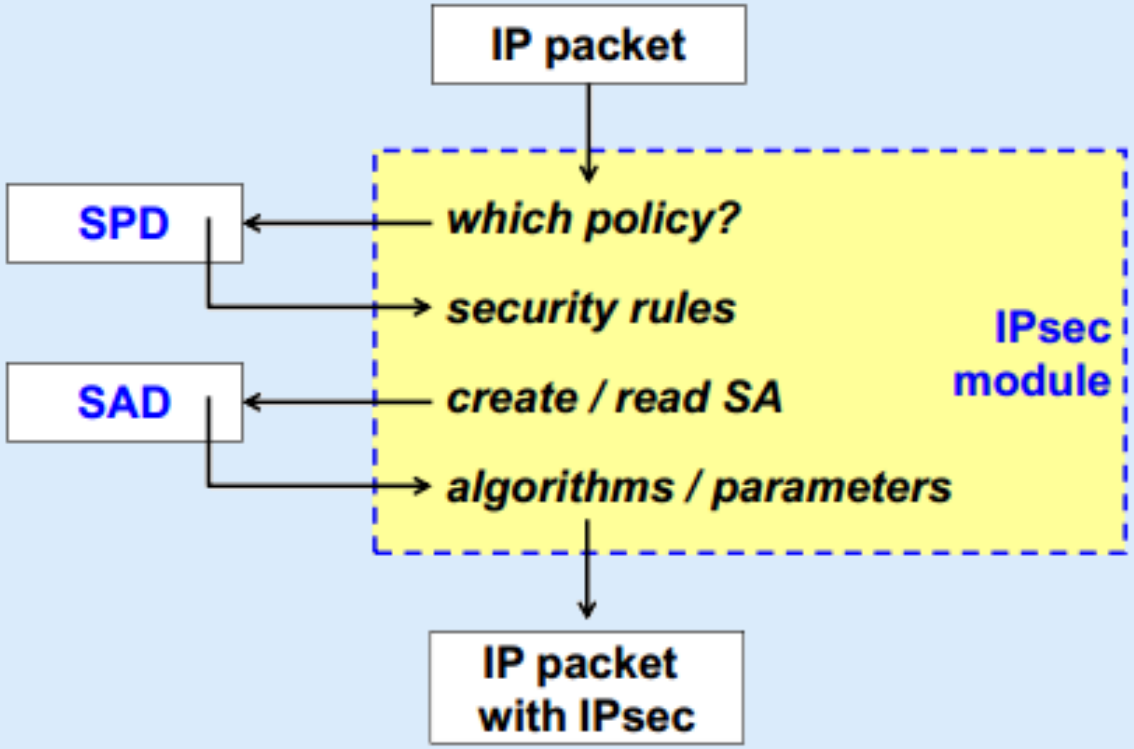


- Three SA identification parameters
  - Security parameter index (SPI)
  - IP destination address
  - IPsec protocol (AH or ESP)

IPsec local databases

- SAD (SA Database)
  - list of active SA and their characteristics
  - maintained by user-processes
- SPD (Security Policy Database)
  - list of security policies to apply to the different packet flows
  - a-priori configured (e.g. manually) or connected to an automatic system (e.g. ISPS: Internet Security Policy System)

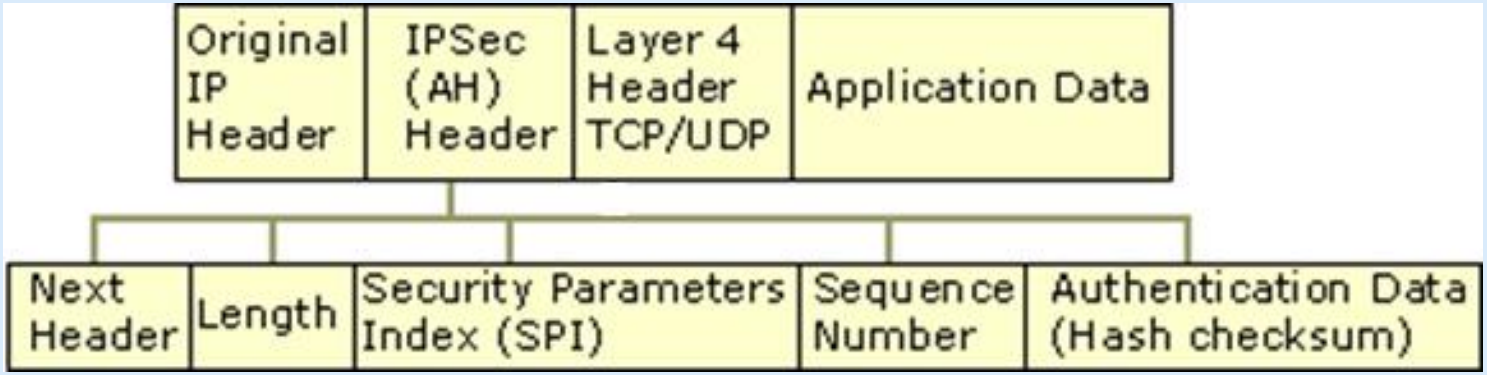
How IPsec works (sending)



Authentication Header (AH)

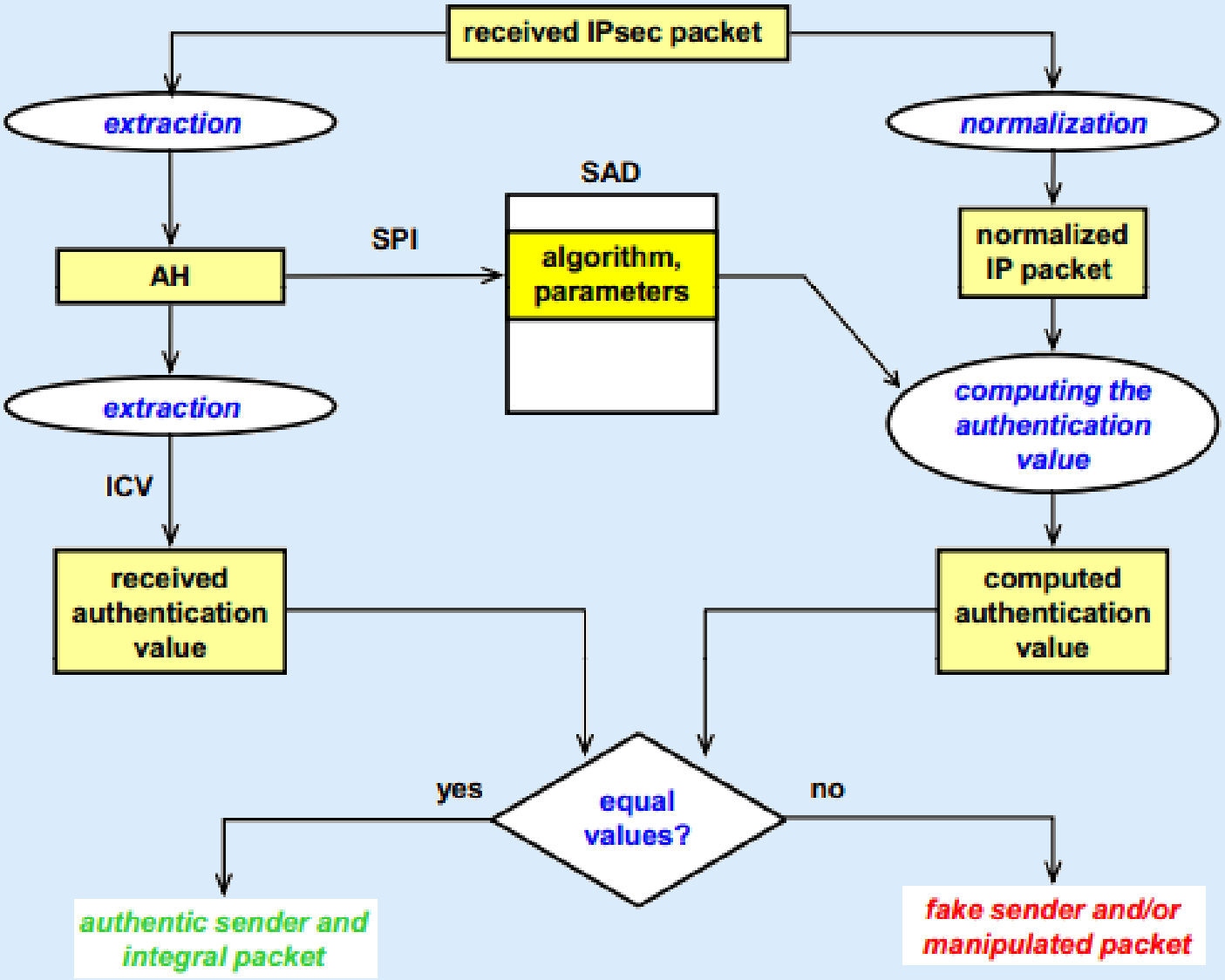
- first version mechanism (RFC-1826):
  - data integrity and sender authentication
  - compulsory support of keyed-MD5 (RFC-1828)
  - optional support of keyed-SHA1 (RFC-1852)
- second version mechanism (RFC-2402):
  - data integrity, sender authentication and protection from replay attack
  - HMAC-MD5
  - HMAC-SHA-1

AH packet: header structure



- Next header: identifies the nature of the payload (TCP/UDP)
- Length: Indicates the length of the AH header
- SPI: Identifies the correct security association for the communication
- Sequence Number: Provides anti-replay protection for the SA
- Auth. Data: contains the Integrity Check Value (ICV) that is used to verify the integrity of the message. The receiver calculates the hash value and checks it against this value (calculated by the sender) to verify integrity.

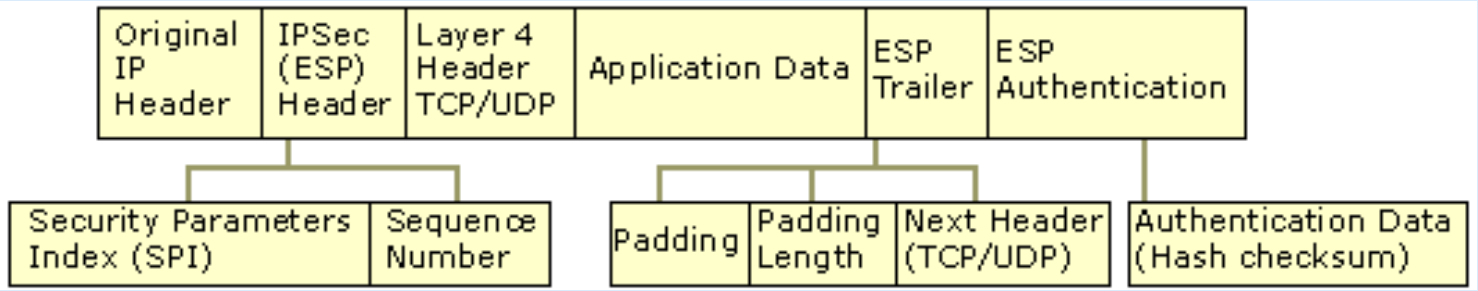
AH verification (receiving)



Encapsulating Security Payload (ESP)

- first version (RFC-1827) included confidentiality only
  - base mechanism: DES-CBC (RFC-1829)
- second version (RFC-2406)
  - provides confidentiality & authentication (but not the IP header, so the coverage is not equivalent to that of AH)

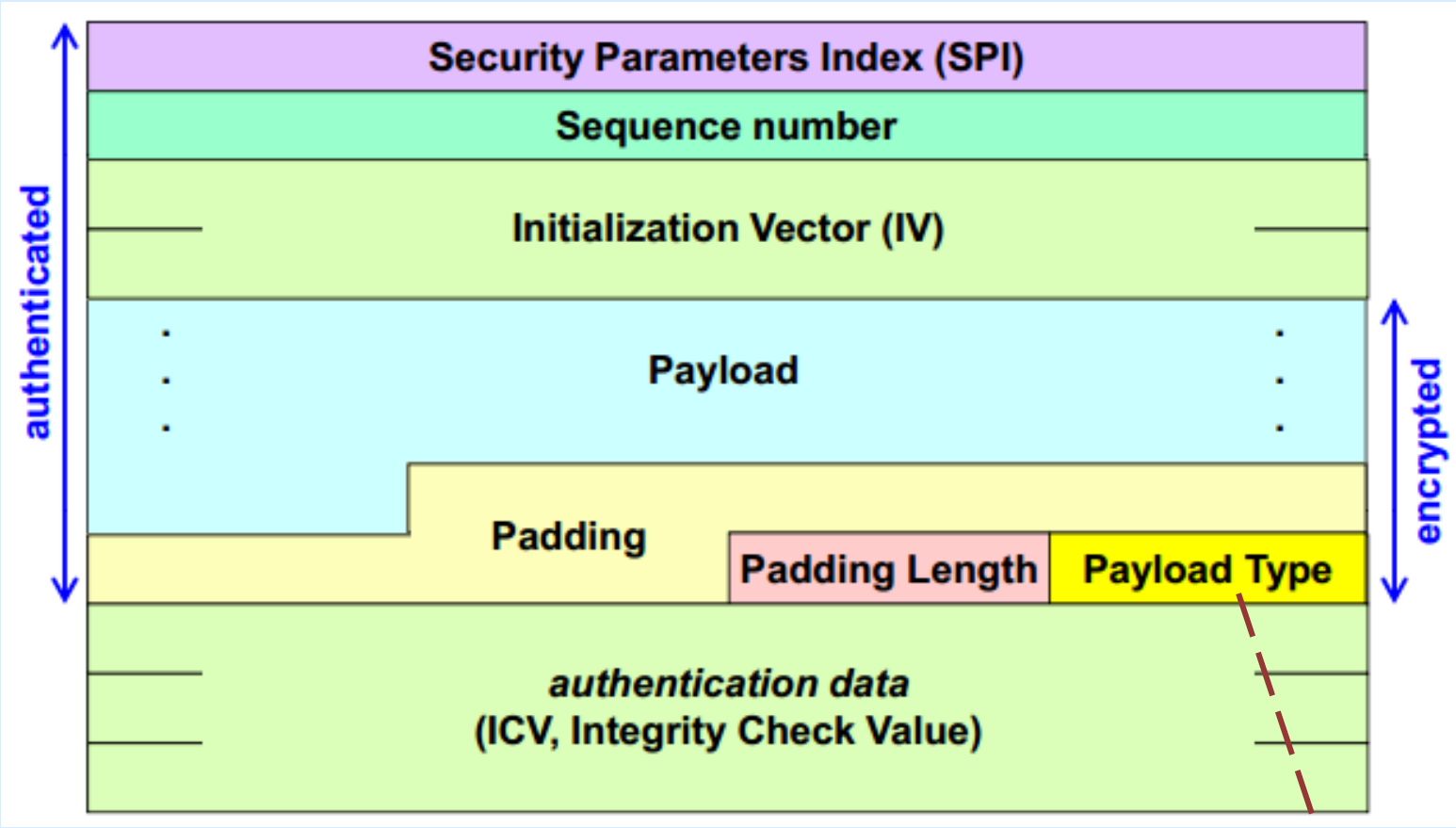
ESP header structure



- SPI, Sequence number, Next header: same fields as in AH

- Auth. Data: Contains the Integrity Check Value (ICV), which is a message authentication code used to verify the sender's identity and message integrity. The ICV is calculated over the ESP header, the payload data and the ESP trailer
- Initialization Vector (IV): optional. After the sequence number

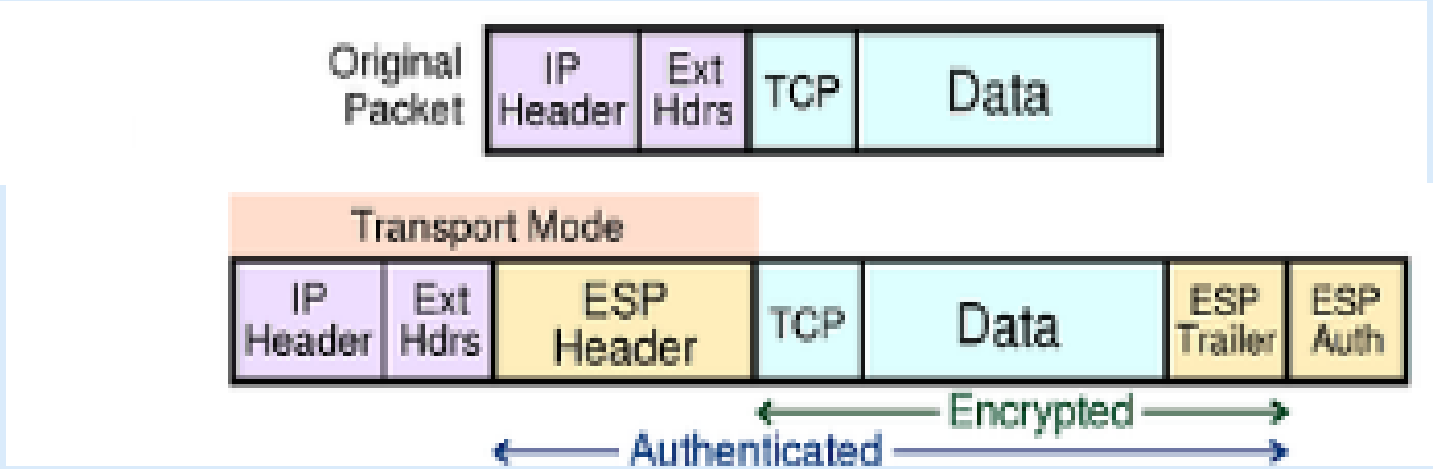
ESP Packet: Encryption & Authentication



aka Next header

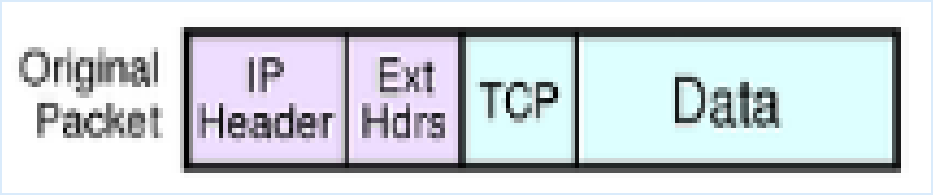
ESP in transport mode

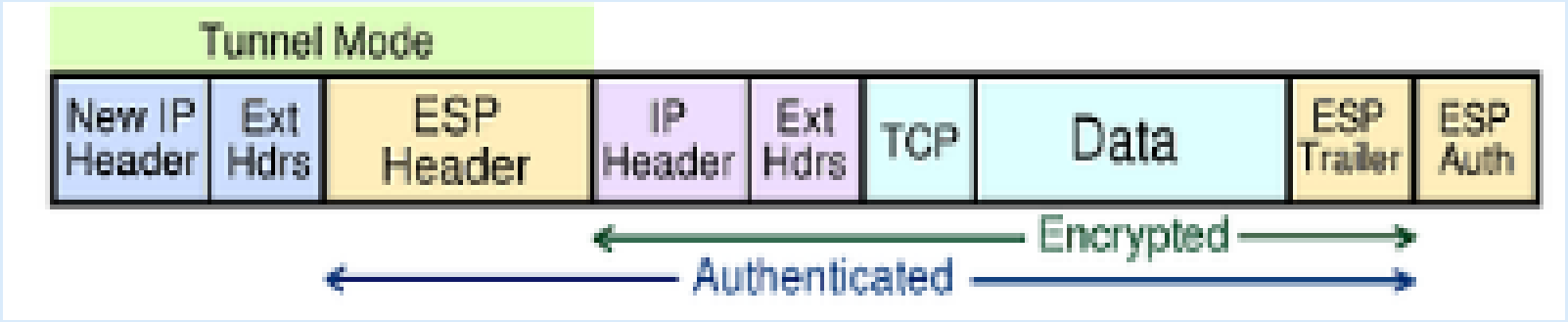
- the payload is hidden (including info needed for QoS or intrusion detection)
- the header remains in clear



ESP in tunnel mode

- hides both the payload and (original) header
- con: larger packet size





Comparing AH & ESP coverage