Unit IV

IP Security

Secure Network Protocols for the OSI Stack

Communication layers	Security protocols
Application layer	ssh, S/MIME, PGP, Kerberos, WSS
Transport layer	TLS, [SSL]
Network layer	IPsec
Data Link layer	[PPTP, L2TP], IEEE 802.1X, IEEE 802.1AE, IEEE 802.11i (WPA2)
Physical layer	Quantum Cryptography

What Security Problem?

- Internet was not created with security in mind
- Today's Internet is primarily comprised of :
  - Public
  - Un-trusted
  - Unreliable IP networks
- Because of this inherent lack of security , the Internet is subject to various types of threats…

Internet Threats

- Data integrity

  The contents of a packet can be accidentally or deliberately modified.

- Identity spoofing

  The origin of an IP packet can be forged.

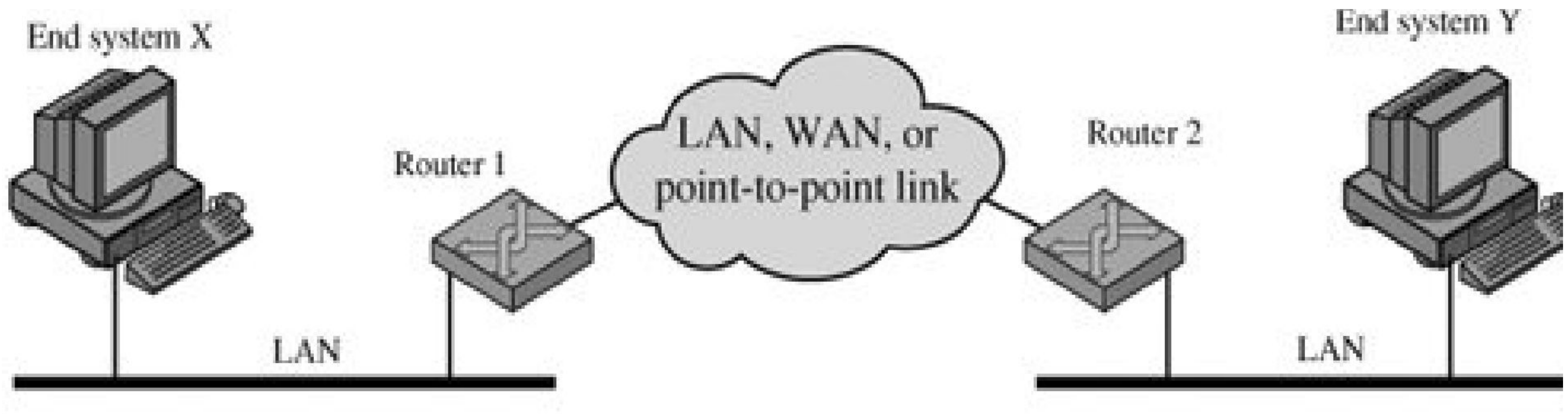
- Anti-reply attacks

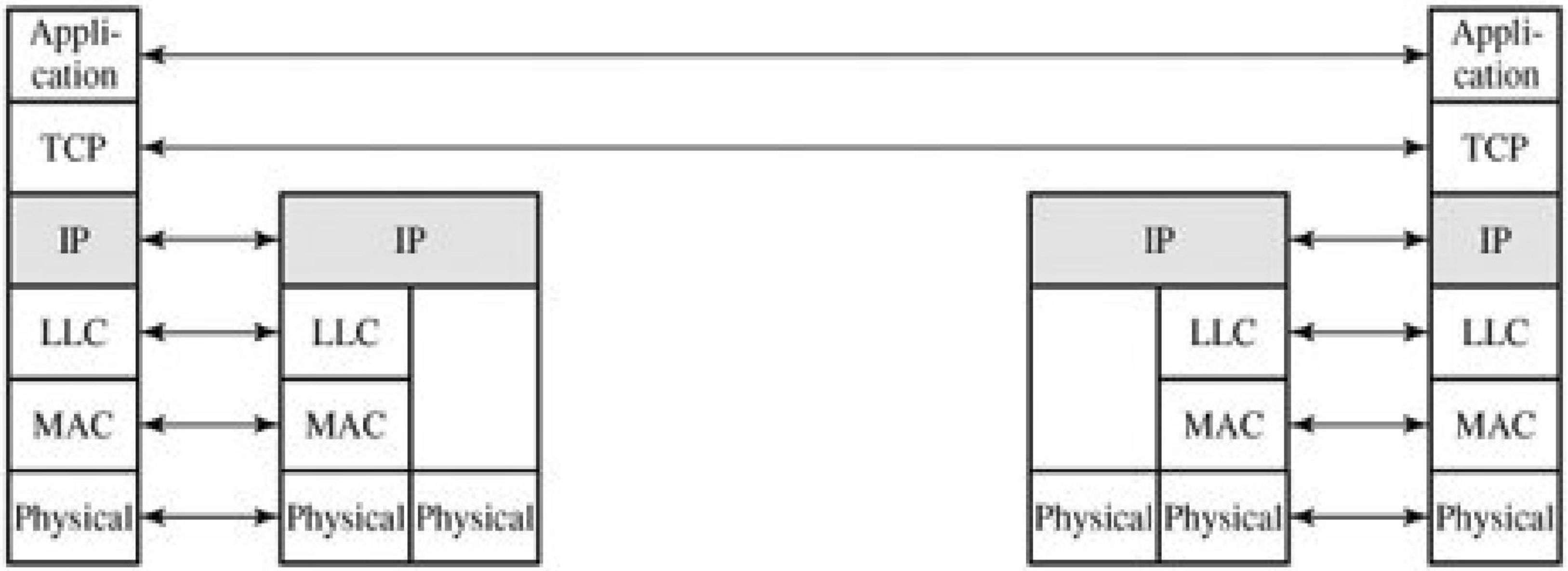
  Unauthorized data can be retransmitted.

- Loss of privacy

  The contents of a packet can be examined in transit.

How Internet Protocol Works

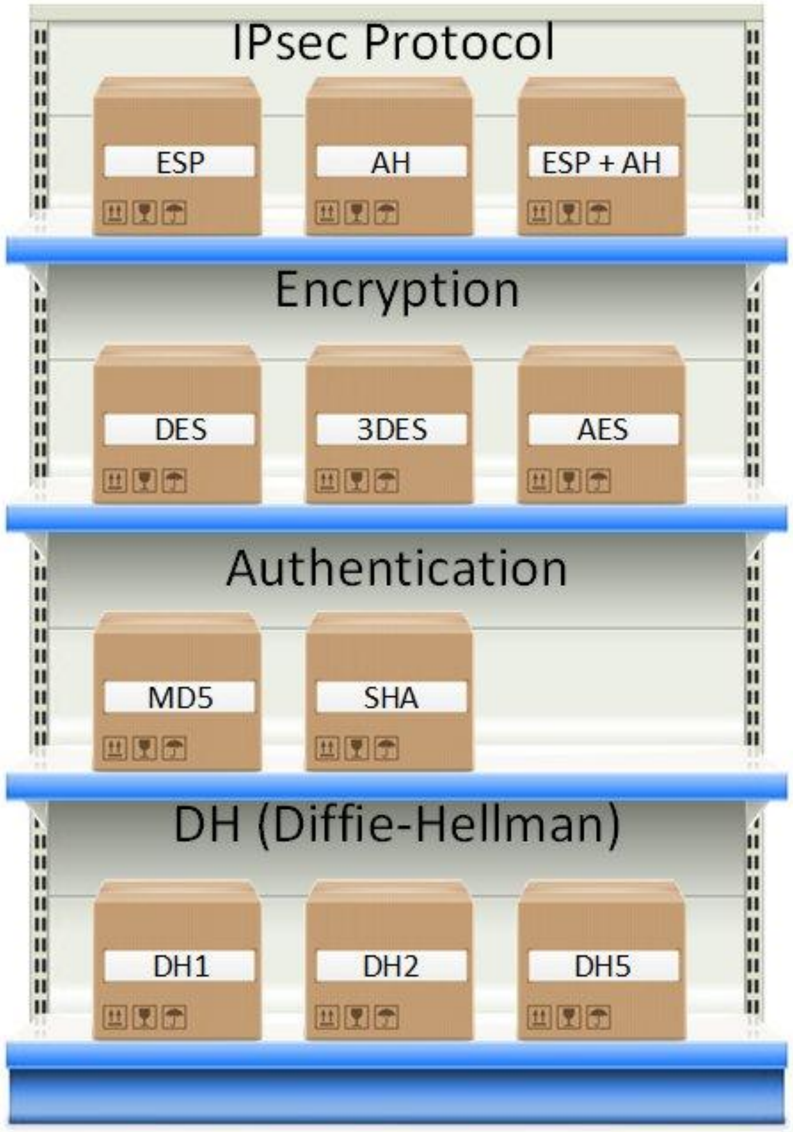




Applications of IP Security

- IPsec can be used on many different devices, it's used on routers, firewalls, hosts and servers. Here are some examples how you can use it:
  - Between two routers to create a site-to-site VPN that "bridges" two LANs together.
  - Between a firewall and windows host for remote access VPN.
  - Between two linux servers to protect an insecure protocol like telnet.
  - Secure branch office connectivity over the Internet
  - Secure remote access over the Internet
  - Establishing extranet and intranet connectivity with partners
  - Enhancing electronic commercial security

IP Sec


IPsec Protocol
ESP
AH
ESP + AH
Encryption
DES
3DES
AES
Authentication
MD5
SHA
DH (Diffie-Hellman)
DH1
DH2
DH5

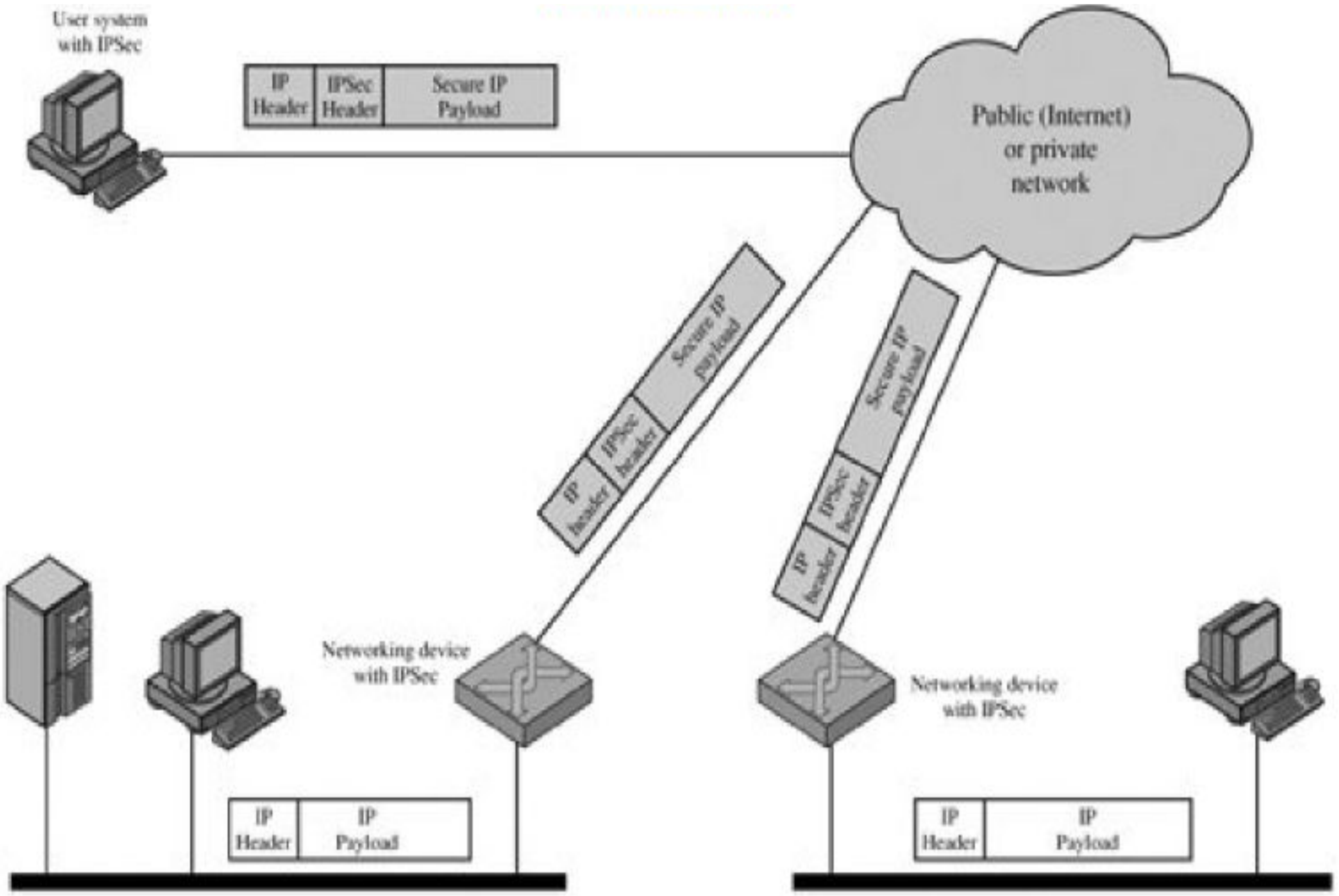
IP Security

- IP security (IPSec) is a capability that can be added to either current version of the Internet Protocol (IPv4-32bits or IPv6-128 bits), by means of additional headers.
- IPSec encompasses three functional areas: **authentication, confidentiality, and key management.**
- Authentication **makes use of the HMAC message authentication code.** Authentication can be applied to the entire original IP packet **(tunnel mode)** or to all of the packet except for the IP header **(transport mode).**
- **Confidentiality is provided by an encryption format known as encapsulating security payload.** Both tunnel and transport modes can be accommodated.

IP Security

- Users have security concerns that cut across protocol layers.
- For example, an enterprise can run a secure, private IP network by disallowing links to untrusted sites encrypting packets that leave the premises, and authenticating packets that enter the premises.
- IP Security is provided at the firewall or routers
- IP-level security encompasses three functional areas: authentication, confidentiality, and key management.
- The authentication mechanism assures that a received packet was, in fact, transmitted by the party identified as the source in the packet header.
- The confidentiality facility enables communicating nodes to encrypt messages to prevent eavesdropping by third parties.
- The key management facility is concerned with the secure exchange of keys.

IP Security Scenario



Benefits of IP Security

- When IPSec is implemented in a firewall or router, it **provides strong security that can be applied to all traffic crossing the perimeter**
- IPSec in a firewall is resistant to bypass if all traffic from the outside must use IP, and the firewall is the only means of entrance from the Internet into the organization.
- IPSec is below the transport layer (TCP, UDP) and so is transparent to applications.
- IPSec can be transparent to end users. There is no need to train users on security mechanisms, issue keying material on a per-user basis
- IPSec can provide security for individual users if needed

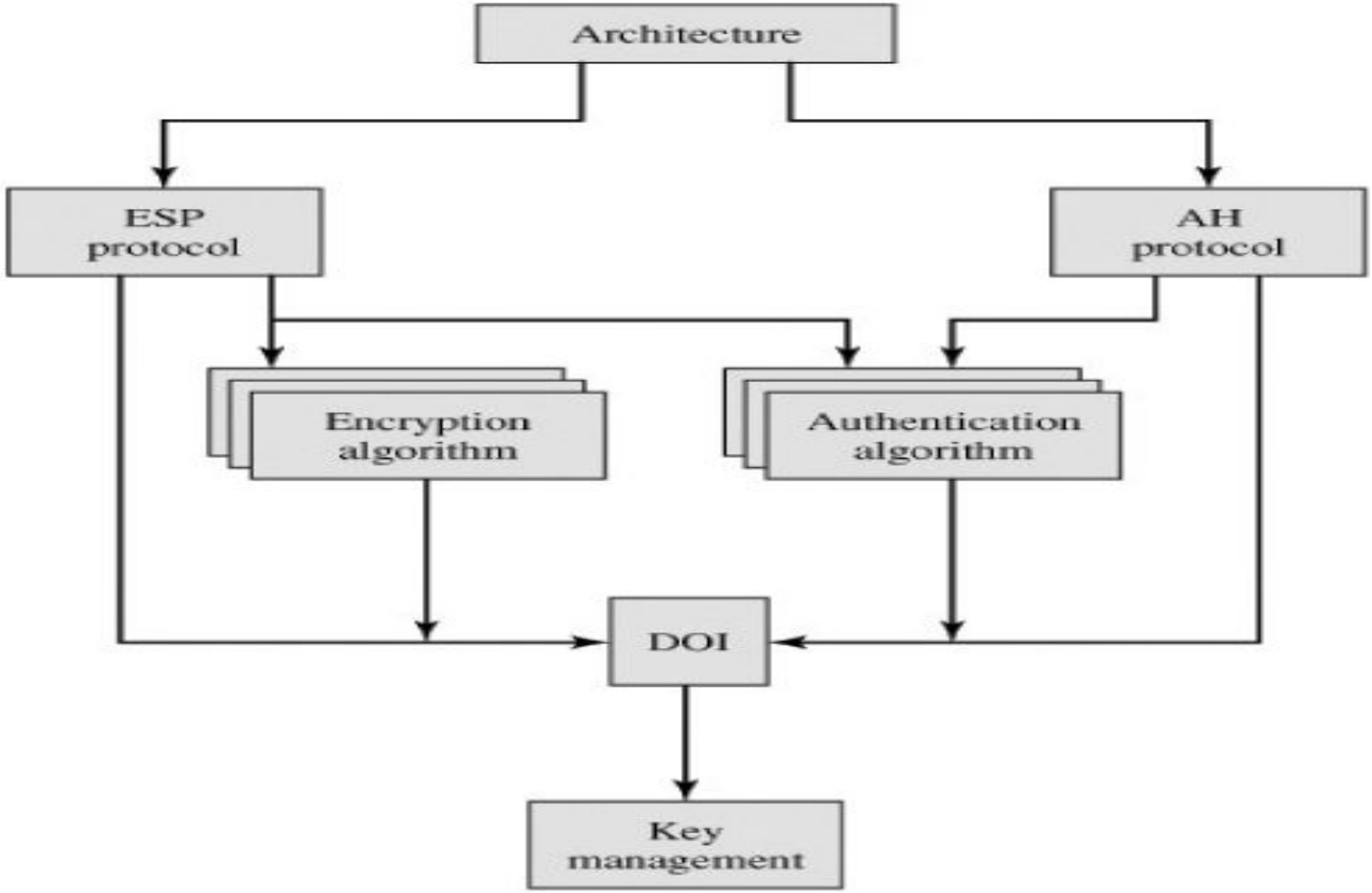
IP Security

- Internet protocol security (IPsec) is a set of protocols that provides security for Internet Protocol.
- It can use cryptography to provide security.
- IPsec can be used for the setting up of virtual private networks (VPNs) in a secure manner.
- IPsec involves two security services:
  - Authentication Header (AH): This authenticates the sender and it discovers any changes in data during transmission.
  - Encapsulating Security Payload (ESP): This not only performs authentication for the sender but also encrypts the data being sent.
- There are two modes of IPsec:
  - Tunnel Mode: This will take the whole IP packet to form secure communication between two places, or gateways.
  - Transport Mode: This only encapsulates the IP payload (not the entire IP packet as in tunnel mode) to ensure a secure channel of communication

IPsec Protocol Suite

- Internet Key Exchange (IKE) protocol
  - For negotiating security parameters and establishing authenticated keys
  - Uses UDP port 500 for ISAKMP
- Encapsulating Security Payload (ESP) protocol
  - For encrypting, authenticating, and securing data
  - IP protocol 50
- Authentication Header (AH) protocol
  - For authenticating and securing data
  - IP protocol 51

IP Security Document Overview



IP Security Document Overview

- ESP: Encrypting Security Payload: ESP for packet Encryption
- AH: Authentication Header: AH for packet Authentication
- Encryption Algorithm
- Authentication Algorithm
- Key Management Schemes
- DOI: Domain of Interpretation: This is required to approve authentication and Encryption algorithm

IPSec Services

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets
  - a form of partial sequence integrity
- Confidentiality (encryption)
- Limited traffic flow confidentiality

IPSec

- IKE builds the tunnels for us but it doesn't authenticate or encrypt user data. We use two other protocols for this:
- **AH (Authentication Header)**
- **ESP (Encapsulating Security Payload)**
- AH and ESP both offer authentication and integrity but only **ESP supports encryption**.
- Because of this, ESP is the most popular choice nowadays.
- Both protocols support two different modes:
  - Transport mode
  - Tunnel mode

IPSec

•Before we can protect any IP packets, we need two IPsec peers that build the IPsec tunnel.

•To establish an IPsec tunnel, we use a protocol called **IKE (Internet Key Exchange)**.

•There are **two phases** to build an IPsec tunnel:

•IKE phase 1

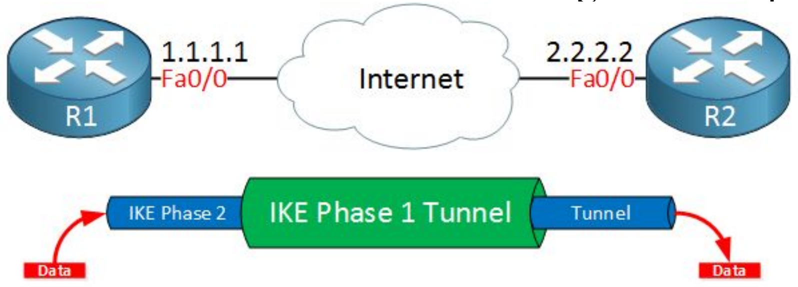
•IKE phase 2

•In IKE phase 1, two peers will negotiate about the encryption, authentication, hashing and other protocols that they want to use and some other parameters that are required. In this phase, an **ISAKMP (Internet Security Association and Key Management Protocol)** session is established. This is also called the **ISAKMP tunnel** or **IKE phase 1** tunnel.

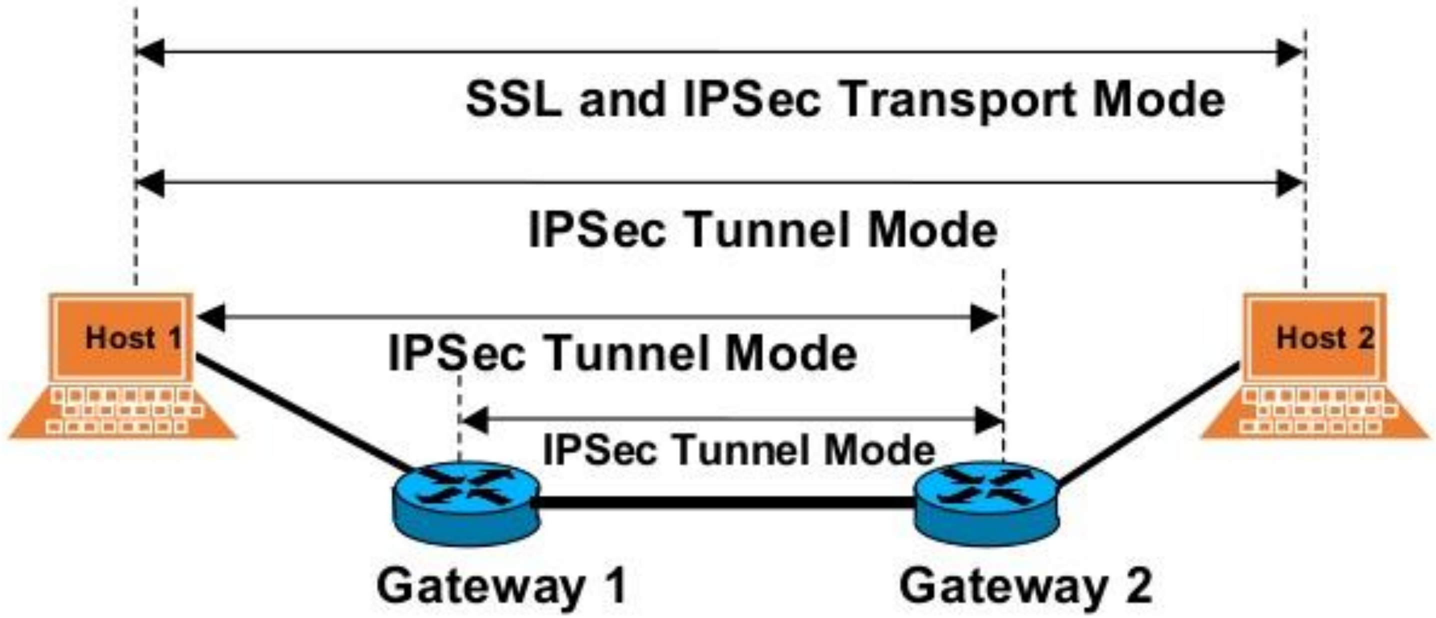
•The collection of parameters that the two devices will use is called a **SA (Security Association)**. Here's an example of two routers that have established the IKE phase 1 tunnel:

•The IKE phase 1 tunnel is only used for **management traffic**. We use this tunnel as a *secure method to establish the second tunnel* called the **IKE phase 2 tunnel** or **IPsec tunnel** and for management traffic like keepalives.

•Once IKE phase 2 is completed, we have an IKE phase 2 tunnel (or IPsec tunnel) that we can use to protect our user data. This user data will be sent through the IKE phase 2 tunnel:



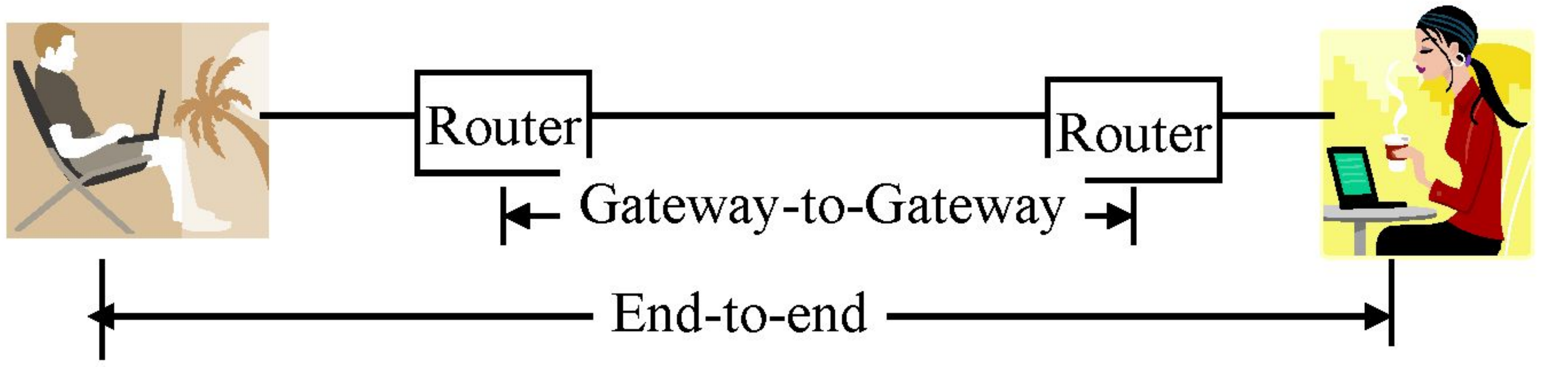
Transport and Tunnel Mode

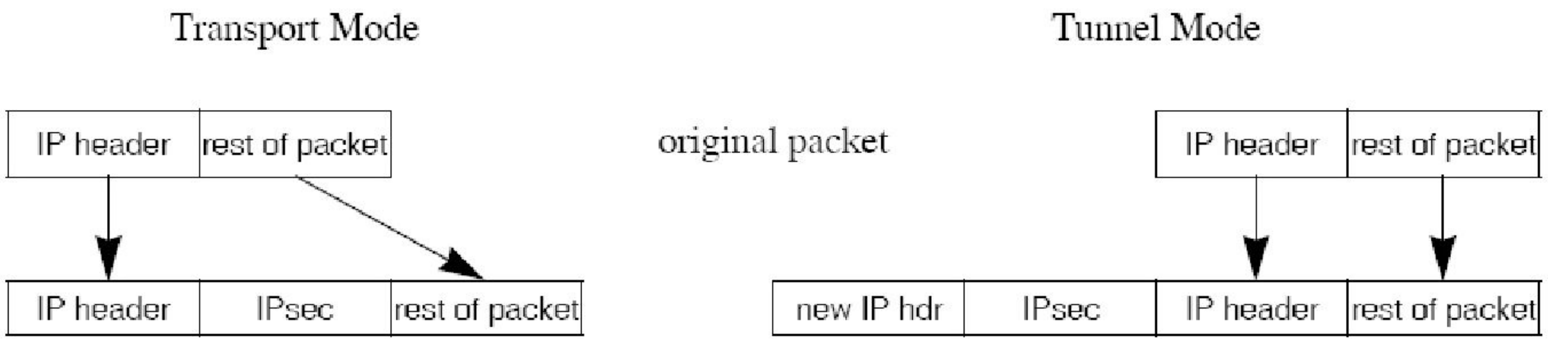


IPSec Tunnel Mode Most popular and appropriate security solution in the data plane

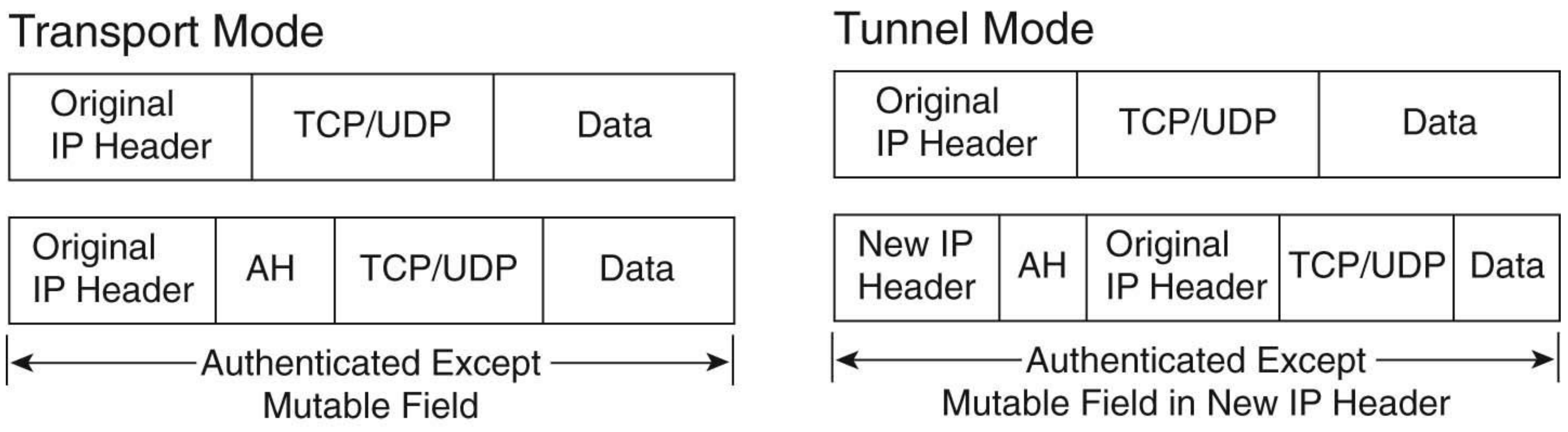
Tunnel vs. Transport Mode

- Gateway-to-gateway vs. end-to-end

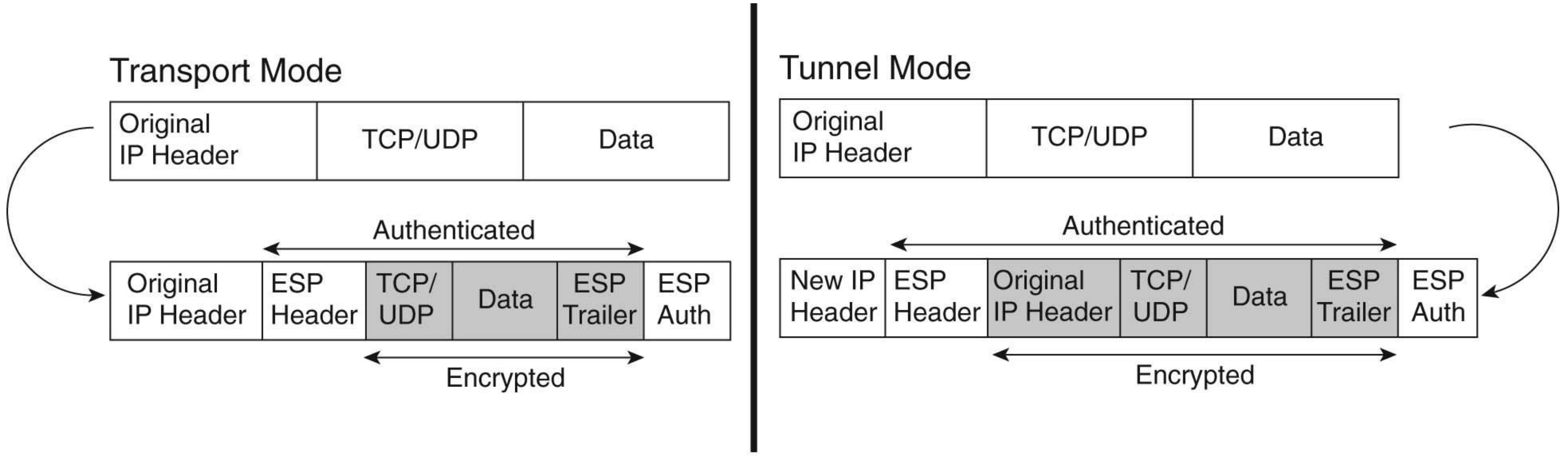




Packet Format Using AH in Tunnel and Transport Modes



Packet Format Using ESP in Tunnel and Transport Modes



Transport Mode

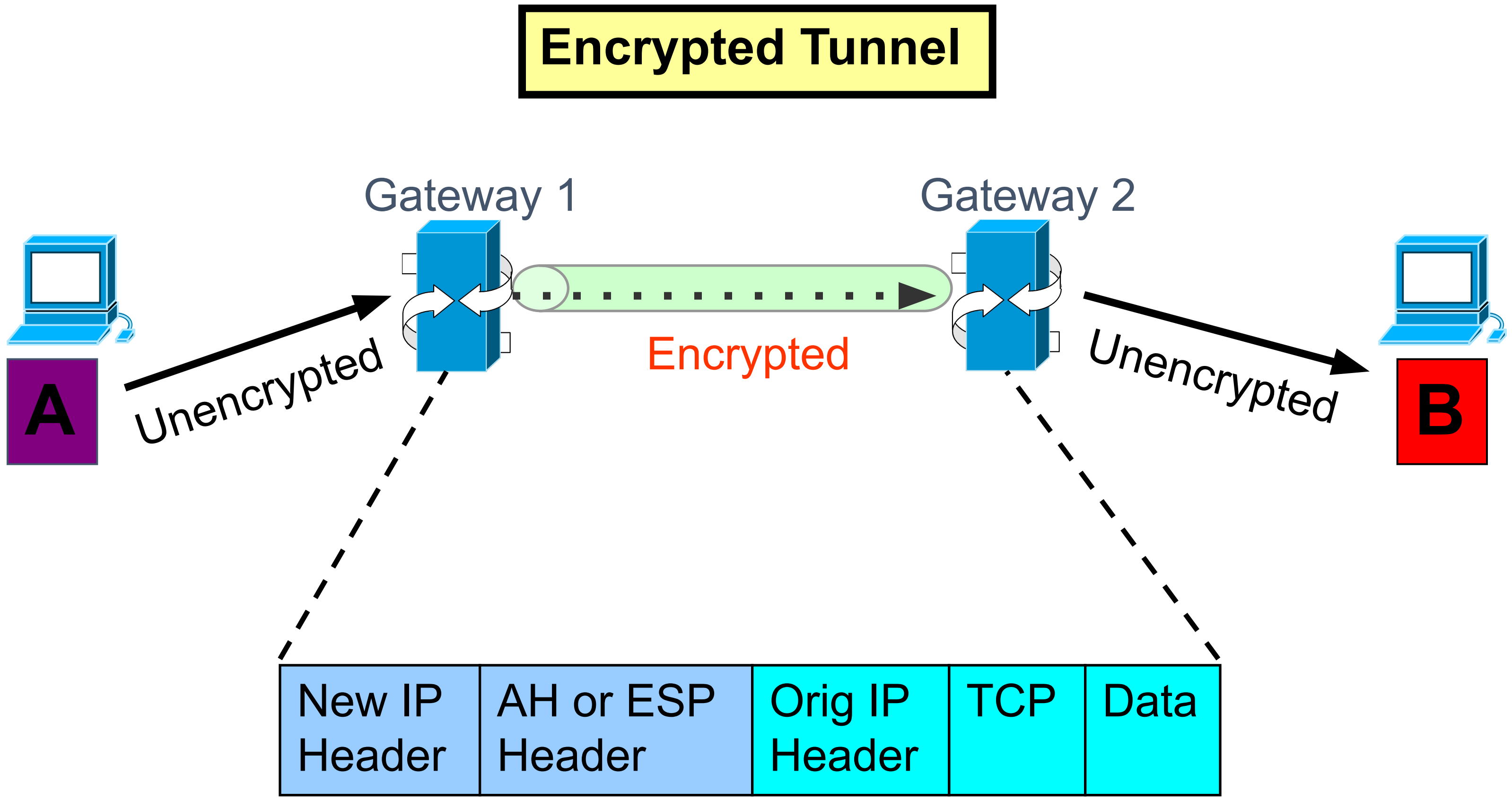
- Transport mode provides protection primarily for upper-layer protocols eg. TCP or UDP segment or an ICMP packet, directly above IP in a host protocol stack.
- for end-to-end communication between two hosts
- When a host runs AH or ESP over IPv4, the payload is the data that normally follow the IP header.
- **ESP in transport mode encrypts and optionally authenticates the IP payload, not the IP header.**
- **AH in transport mode authenticates the IP payload and selected portions of the IP header.**

Tunnel mode

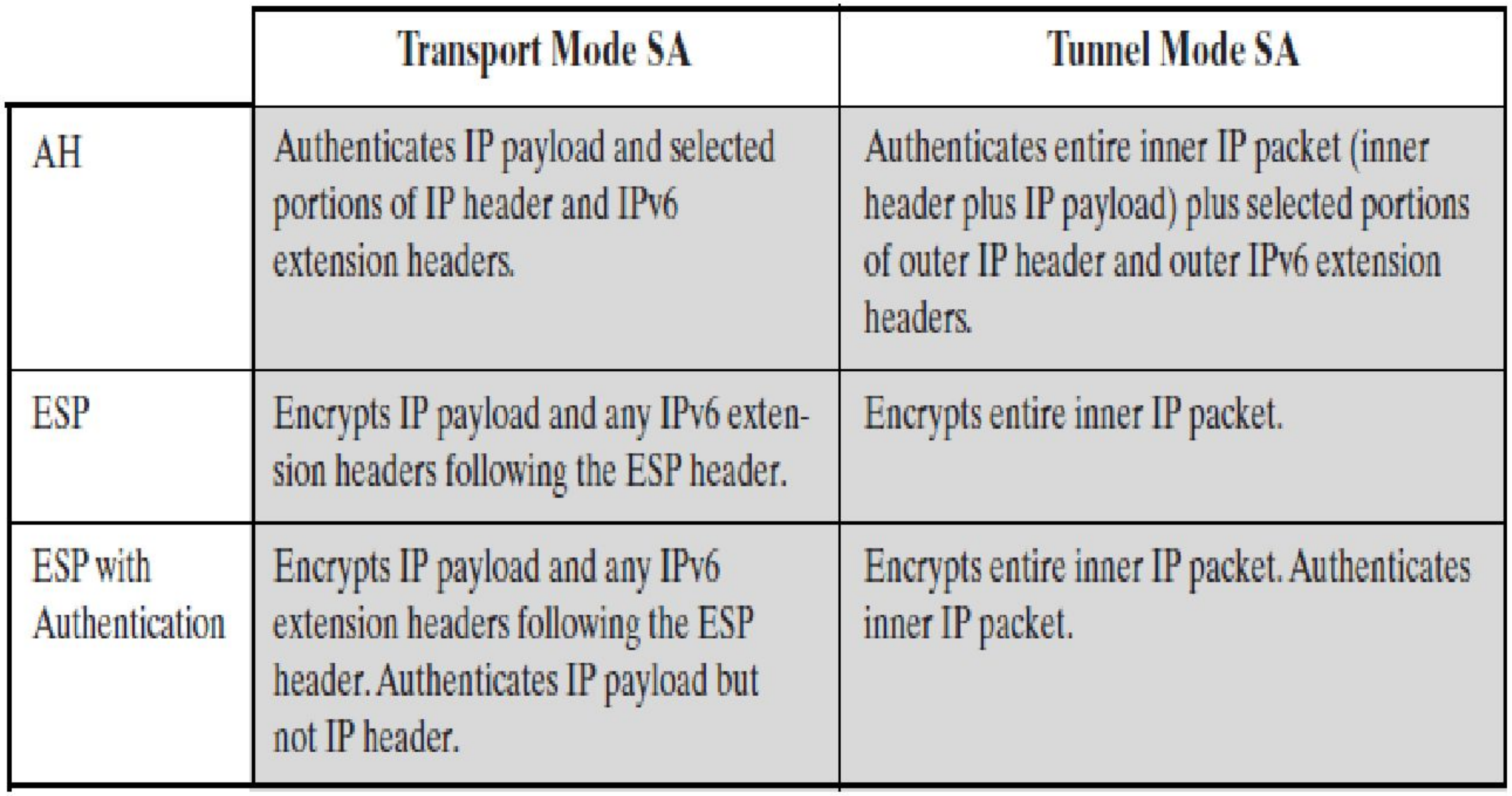
- Tunnel mode provides protection to the entire IP packet.
- After the AH or ESP fields are added to the IP packet, the entire packet plus security fields is treated as the payload of new outer IP packet with a new outer IP header.
- The entire original, inner, packet travels through a tunnel from one point of an IP network to another;
- No routers along the way are able to examine the inner IP header.
- Because the original packet is encapsulated, the new, larger packet may have totally different source and destination addresses, adding to the security.
- **ESP in tunnel mode encrypts and optionally authenticates the entire inner IP packet, including the inner IP header. AH in tunnel mode authenticates the entire inner IP packet and selected portions of the outer IP header.**
- Tunnel mode is used when one or both ends of a security association (SA) are a security gateway, such as a firewall or router that implements IPsec.

Transport Mode vs. Tunnel Mode

- Transport mode: host -> host
- Tunnel mode: host->gateway or gateway->gateway

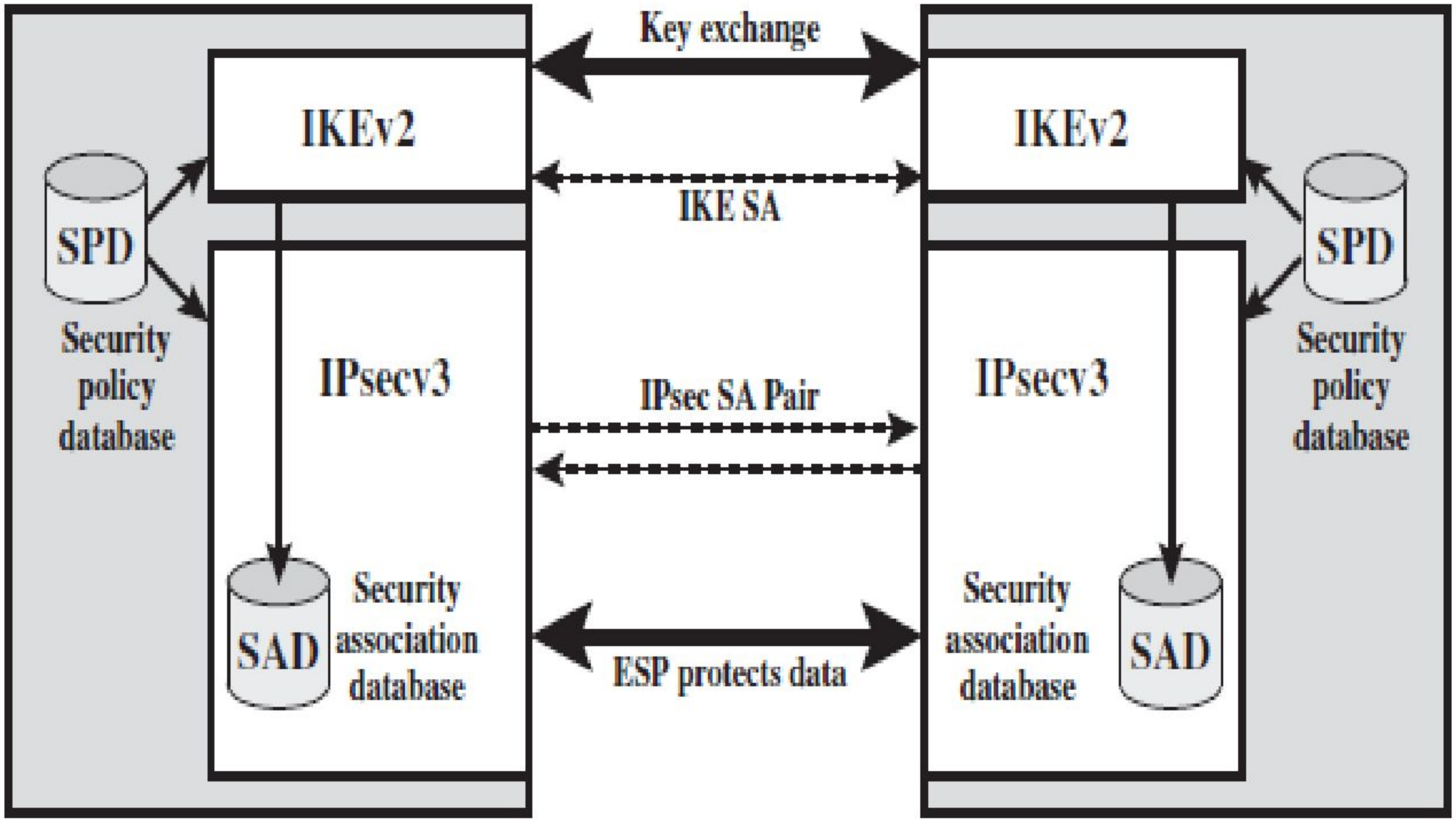


Transport and Tunnel mode



	Transport Mode SA	Tunnel Mode SA
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers.	Authenticates entire inner IP packet (inner header plus IP payload) plus selected portions of outer IP header and outer IPv6 extension headers.
ESP	Encrypts IP payload and any IPv6 extension headers following the ESP header.	Encrypts entire inner IP packet.
ESP with Authentication	Encrypts IP payload and any IPv6 extension headers following the ESP header. Authenticates IP payload but not IP header.	Encrypts entire inner IP packet. Authenticates inner IP packet.

IP Sec Architecture



IP Security Policy

- Fundamental to the operation of IPsec is the concept of a security policy applied to each IP packet that transits from a source to a destination.
- IPsec policy is determined primarily by the interaction of two databases,
- the **security association database (SAD)**
- the **security policy database (SPD)**.

Security policy and association

- **Security Policies:** A *security policy* is a rule that is programmed into the IPSec implementation that tells it how to process different datagrams received by the device. For example, security policies are used to decide if a particular packet needs to be processed by IPSec or not.
- Security policies for a device are stored in the device's *Security Policy Database (SPD)*.
- **Security Associations:** A *Security Association (SA)* is a set of security information that describes a particular kind of secure connection between one device and the other
- A device's security associations are contained in its *Security Association Database (SAD)*.
- The main difference between them is that security policies are general while security associations are more specific. To determine what to do with a particular datagram, a device first checks the SPD. The security policies in the SPD may reference a particular security association in the SAD. If so, the device will look up that security association and use it for processing the datagram.

Security Association

- A **Security Association** (SA) is the establishment of shared security attributes between two network entities to support secure communication.
- An SA may include attributes such as: cryptographic algorithm and mode; traffic encryption key; and parameters for the network data to be passed over the connection.
- The framework for establishing security associations is provided by the Internet Security Association and Key Management Protocol (ISAKMP). Protocols such as Internet Key Exchange
- For example, a mobile subscriber and a base station . The subscriber may subscribe itself to more than one service. Therefore, each service may have different service primitives, such as a data encryption algorithm, public key, or initialization vector.
- To make things easier, all of this security information is grouped logically, and the logical group itself is a Security Association. Each SA has its own ID called SAID. So both the base station and mobile subscriber will share the SAID, and they will derive all the security parameters.
- In other words, an SA is a logical group of security parameters that enable the sharing of information to another entity.

Security Associations

- One way relationship between sender and receiver that affords security for traffic flow
- For two way, two associations are required---defined by 3 parameters:
  - **Security Parameters Index (SPI): A bit string assigned to this SA for local significance only.** The SPI is carried in AH and ESP headers to enable the receiving system to select the SA under which a received packet will be processed.
  - **IP Destination Address: Currently, only unicast addresses are allowed; this is the address of the destination endpoint of the** SA, which may be an end user system or a network system such as a firewall or router.
  - **Security Protocol Identifier:** This indicates whether the association is an AH or ESP security association has a number of other parameters
- have a database of Security Associations

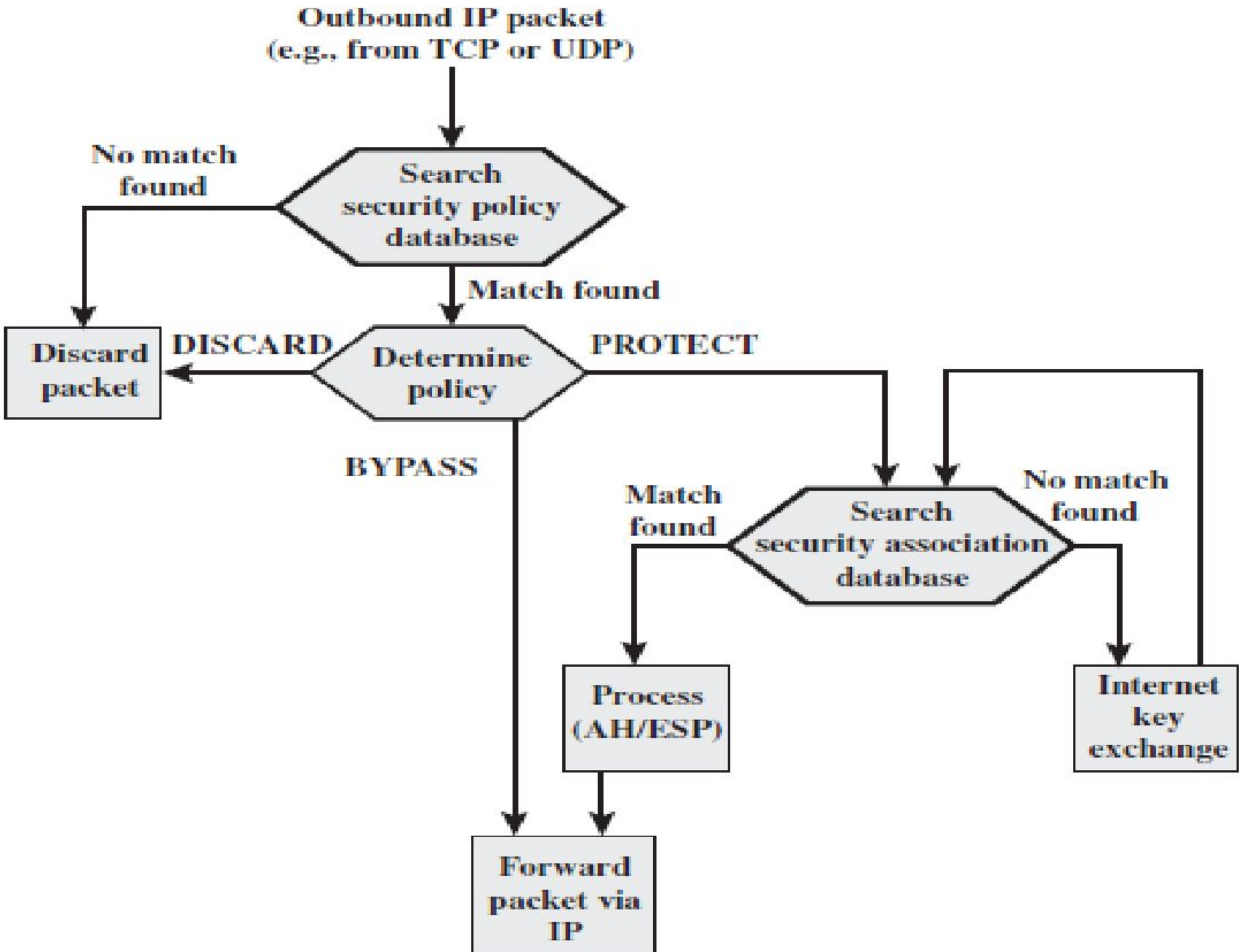
SAD Parameters

- **Security Parameter Index:** A 32-bit value selected by the receiving end of an SA to uniquely identify the SA.
- **Sequence Number Counter:** A 32-bit value used to generate the Sequence Number field in AH or ESP Headers
- **Sequence Counter Overflow:** A flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent further transmission of packets on this SA.
- **Anti-Replay Window:** Used to determine whether an inbound AH or ESP packet is a replay.
- **AH Information:** Authentication algorithm, keys, key lifetimes, and related parameters being used with AH.
- **ESP Information:** Encryption and authentication algorithm, keys, initialization values, key lifetimes, and related parameters being used with ESP
- **Lifetime of this Security Association:** A time interval or byte count after which an SA must be replaced with a new SA .

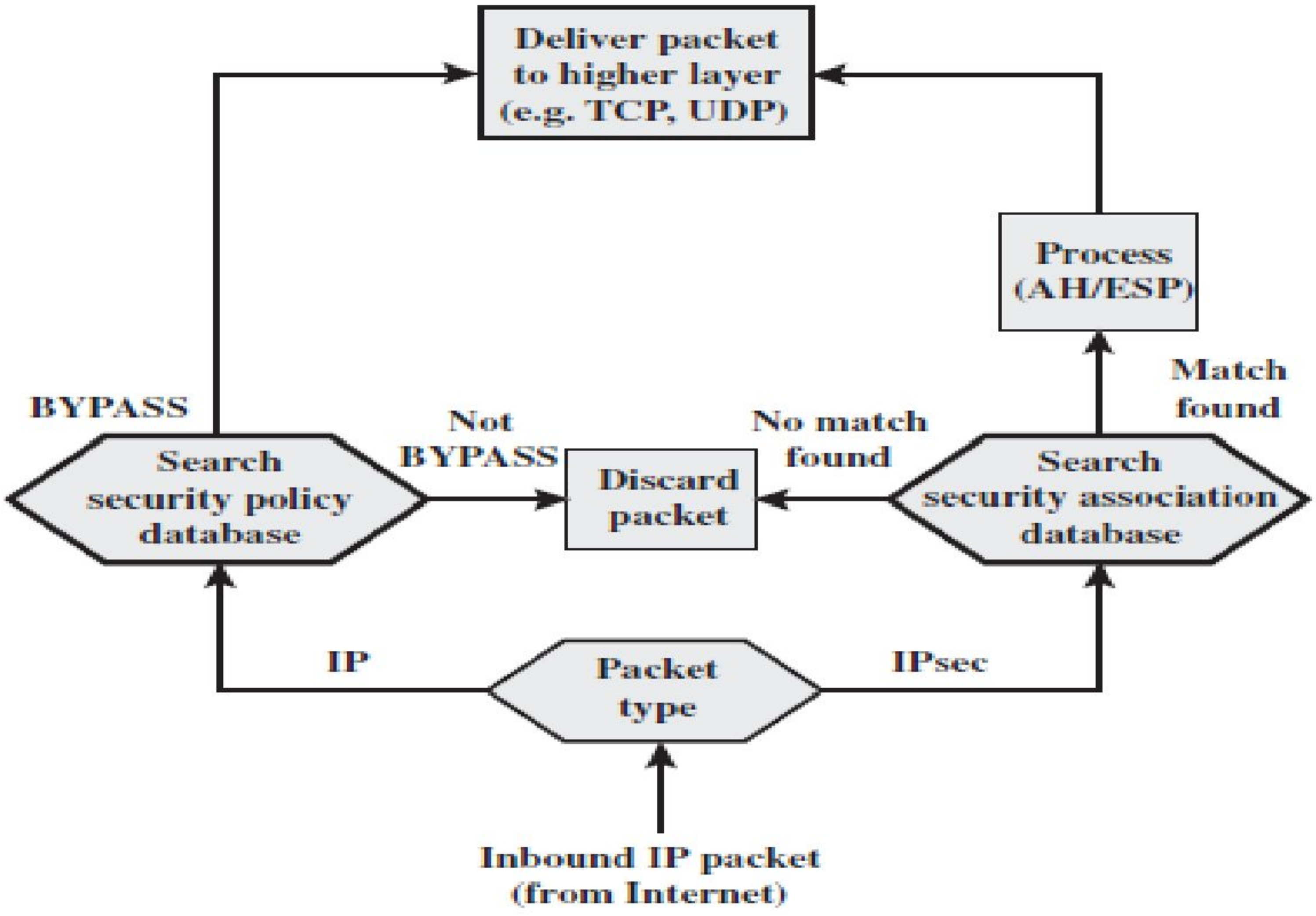
SPD Parameters

- **Remote IP Address:** This may be a single IP address, an enumerated list or range of addresses, or a wildcard (mask) address. The latter two are required to support more than one destination system sharing the same SA .
- **Local IP Address:** This may be a single IP address, an enumerated list or range of addresses, or a wildcard (mask) address. The latter two are required to support more than one source system sharing the same SA.
- **Next Layer Protocol:** The IP protocol header (IPv4, IPv6, or IPv6 Extension)
- **Name:** A user identifier from the operating system. This is not a field in the IP or upper-layer headers but is available if IPsec is running on the same operating system as the user.
- **Local and Remote Ports:** These may be individual TCP or UDP port values, an enumerated list of ports, or a wildcard port

Processing for outbound packets



Processing model of inbound traffic


Deliver packet to higher layer (e.g. TCP, UDP)
Process (AH/ESP)
Match found
BYPASS
Not BYPASS
No match found
Search security policy database
Discard packet
Search security association database
IP
Packet type
IPsec
Inbound IP packet (from Internet)

Different ESP MODES

- ESP : Encapsulating Security Payload
- ESP supports two modes:
  - Tunnel Mode
  - Transport Mode

Transport Mode: ESP header is inserted after the IP header and before the next layer protocol header

Tunnel Mode: ESP header is inserted after the IP header and before the encapsulated IP Header

ESP may be applied with the AH or alone or combination of both.

Authentication Header (AH): **AH provides the authentication on entire packet** but ESP provides the authentication on IP datagram only

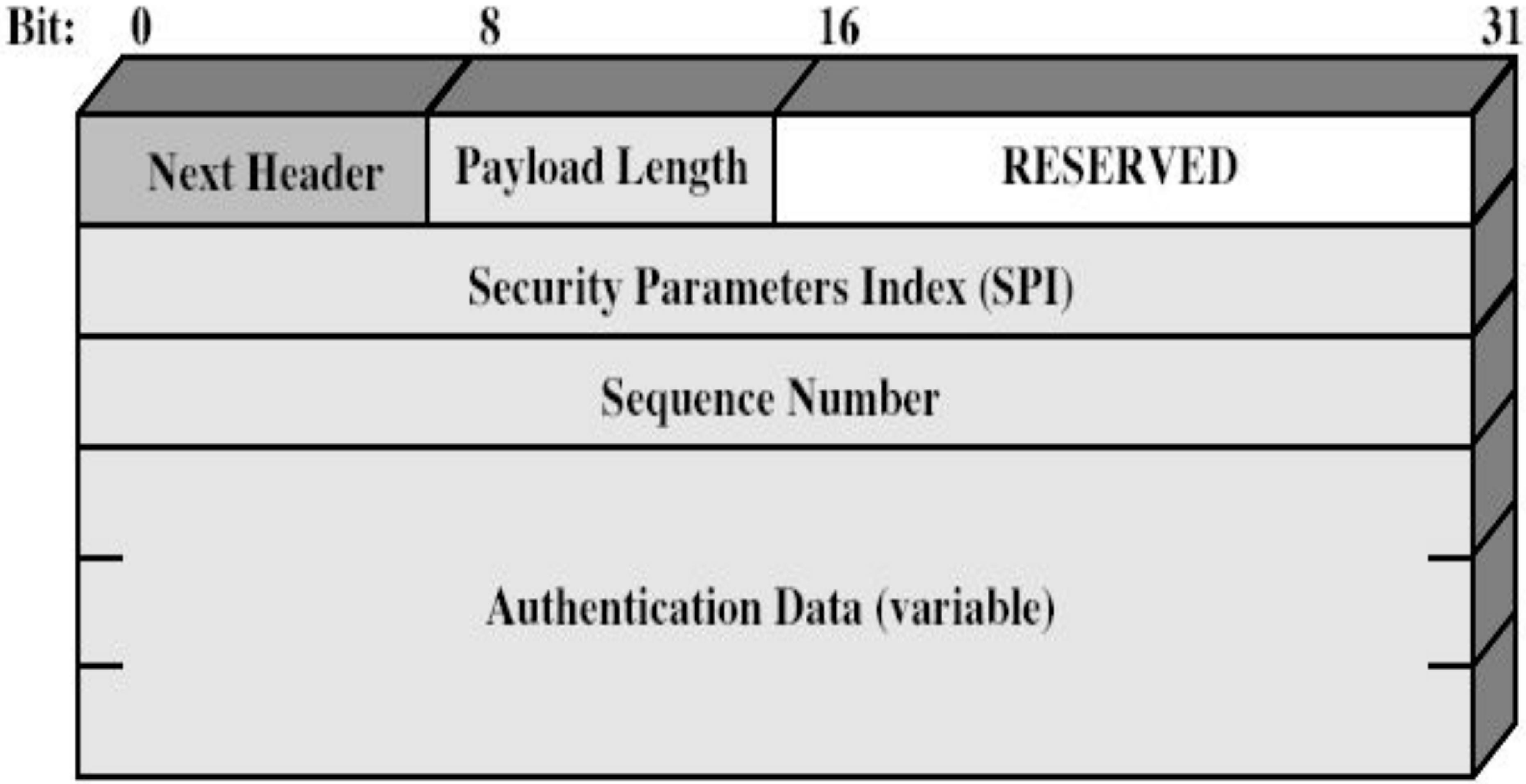
Authentication Header (AH)

- Provides source authentication
  - Protects against source spoofing
- Provides data integrity
- Protects against replay attacks
  - Use monotonically increasing sequence numbers
  - Protects against denial of service attacks
- NO protection for confidentiality!

AH Details

- Use 32-bit monotonically increasing sequence number to avoid replay attacks
- Use cryptographically strong hash algorithms to protect data integrity (96-bit)
  - Use symmetric key cryptography
  - HMAC-SHA-96, HMAC-MD5-96

Authentication Header


Bit:
0
8
16
31
Next Header
Payload Length
RESERVED
Security Parameters Index (SPI)
Sequence Number
Authentication Data (variable)

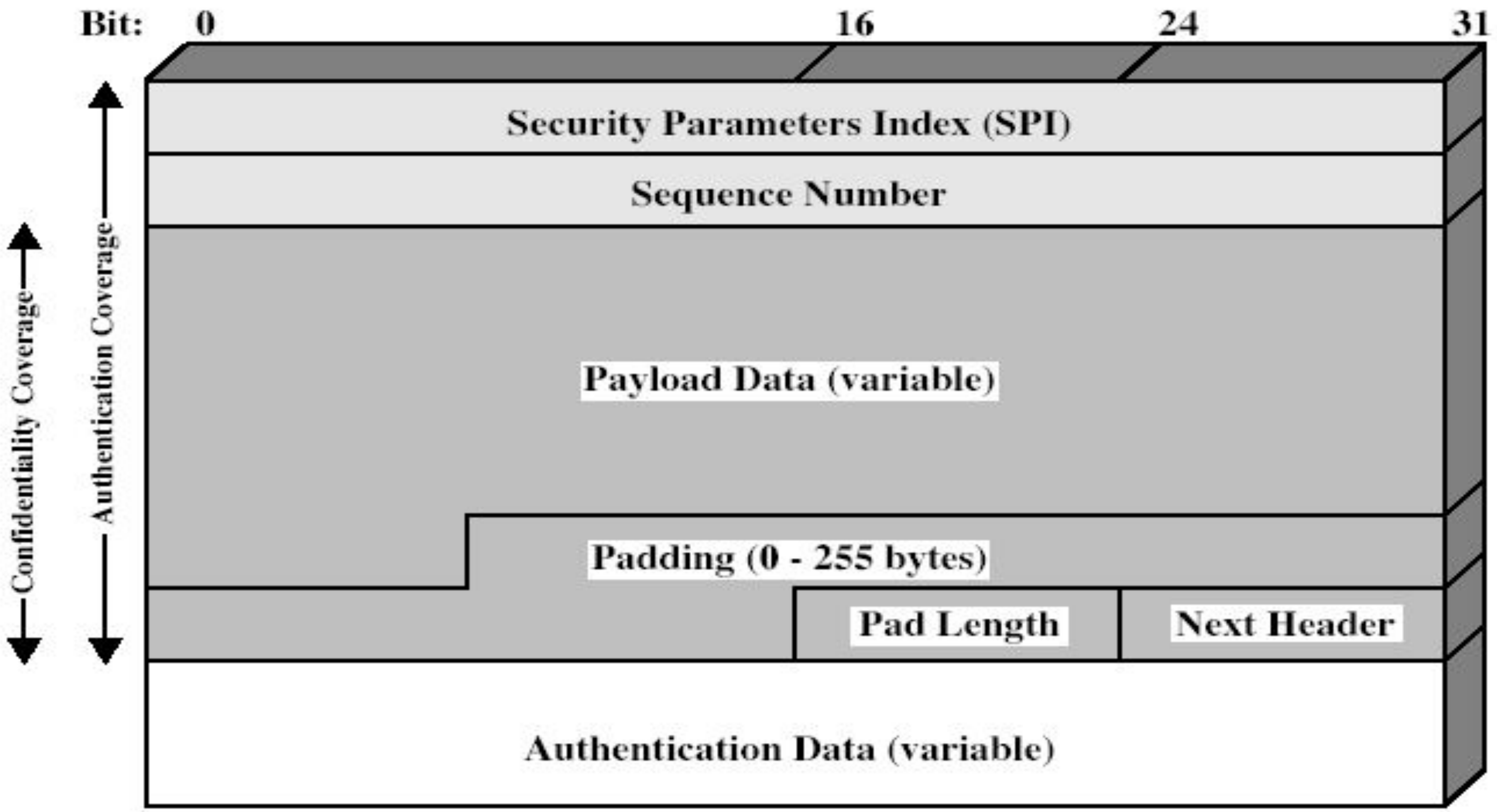
Authentication Header

- **Next Header** Identifies the next header that uses the IP protocol ID. For example, the value might be "6" to indicate TCP.
- **Length** Indicates the length of the AH header.
- **Security Parameters Index (SPI)** Used in combination with the destination address and the security protocol (AH or ESP) to identify the correct security association for the communication.
- **Sequence Number** Provides anti-replay protection for the SA. It is 32-bit, incrementally increasing number (starting from 1) that is never allowed to cycle and that indicates the packet number sent over the security association for the communication. The receiver checks this field to verify that a packet for a security association with this number has not been received already. If one has been received, the packet is rejected.
- **Authentication 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.

Encapsulating Security Payload (ESP)

- provides message content confidentiality & limited traffic flow confidentiality
- can optionally provide the same authentication services as AH
- supports range of ciphers, modes, padding
    - incl. DES, Triple-DES, RC5, IDEA, CAST etc
    - CBC most common
    - pad to meet blocksize, for traffic flow

Encapsulating Security Payload



ESP Header

- **Security Parameters Index** Identifies, when used in combination with the destination address and the security protocol (AH or ESP), the correct security association for the communication. The receiver uses this value to determine the security association with which this packet should be identified.
- **Sequence Number** Provides anti-replay protection for the SA. It is 32-bit, incrementally increasing number (starting from 1) that indicates the packet number sent over the security association for the communication. The sequence number is never allowed to cycle. The receiver checks this field to verify that a packet for a security association with this number has not been received already. If one has been received, the packet is rejected.
- The ESP trailer contains the following fields:
- **Padding** 0 to 255 bytes is used for 32-bit alignment and with the block size of the block cipher.
- **Padding Length** Indicates the length of the Padding field in bytes. This field is used by the receiver to discard the Padding field.
- **Next Header** Identifies the nature of the payload, such as TCP or UDP.
- The ESP Authentication Trailer contains the following field:
- **Authentication Data** Contains the Integrity Check Value (ICV), and a message authentication code that is 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.

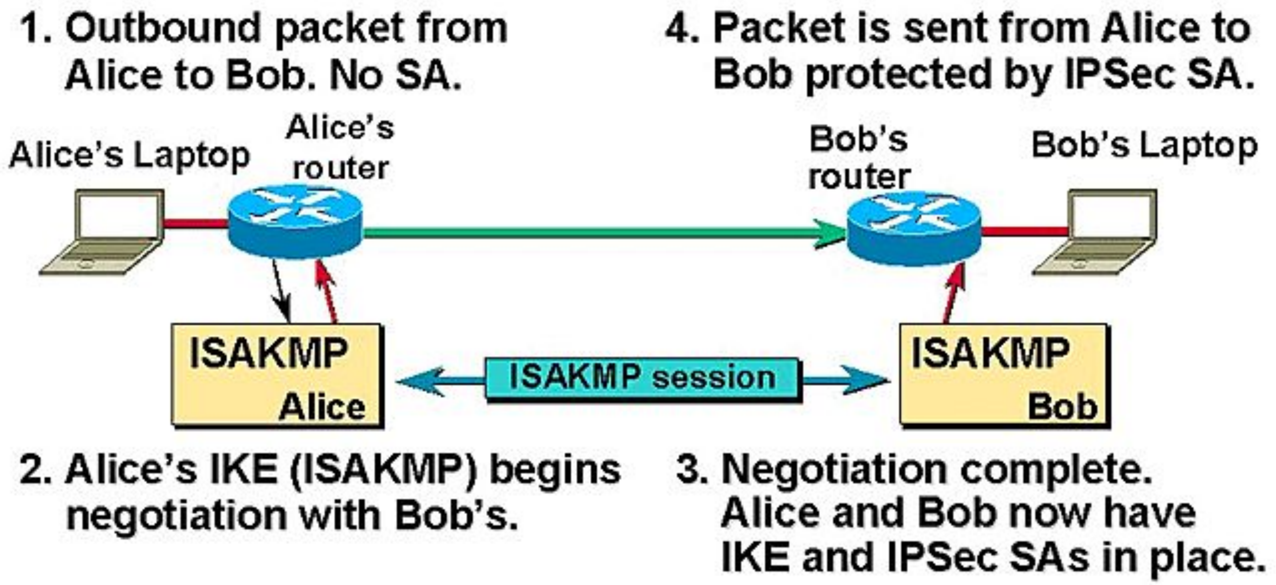
Internet Key Exchange

- Before IPSec sends authenticated or encrypted IP data,
- The sender and receiver must agree
  - The protocols,
  - Encryption algorithms
  - Keys to use for message integrity, authentication and encryption.
- IKE is used to negotiate these and provides primary authentication.
- Key lifetimes can be set and rekeying can be done Automatically
- Internet Key Exchange (IKE) is a network security Protocol designed to allow two devices to dynamically exchange Encryption Keys and negotiate Security Associations (SA).
- Internet Key Exchange (IKE) Security Associations (SA) can be established dynamically and removed at a negotiated time period.
- Internet Key Exchange is a hybrid protocol made from the combination of Oakley, SKEME (A Versatile Secure Key Exchange Mechanism for Internet) and ISAKMP (Internet Security Association and Key Management Protocol) protocols.

Internet Key Exchange

- Internet Security Association and Key Management Protocol (ISAKMP) provide a framework for authentication and key exchange.
- The Oakley Protocol is a Key Agreement protocol that allows the authenticated devices to exchange keys using the Diffie-Hellman key exchange algorithm.
- SKEME is a key exchange mechanism suggested by Hugo Krawczyk (IBM T.J.Watson Research Center). SKEME provides anonymity, and allows repudiation of communication by avoiding the use of digital signatures and quick Key refreshment. SKEME uses cookie against Denial-of-Service (DoS) attacks

IKE phases



1. Outbound packet from Alice to Bob. No SA.

2. Alice's IKE (ISAKMP) begins negotiation with Bob's.

3. Negotiation complete. Alice and Bob now have IKE and IPSec SAs in place.

4. Packet is sent from Alice to Bob protected by IPSec SA.

- IKE sets up a secure channel to negotiate the IPSec security associations.

IKE phases

- In phase 1 of this process, IKE creates an authenticated, secure channel between the two IKE peers, called the *IKE security association*.
- The Diffie-Hellman key agreement is always performed in this phase.
- In phase 2, IKE negotiates the IPSec security associations and generates the required key material for IPSec.
- The sender offers one or more transform sets that are used to specify an allowed combination of transforms with their respective settings. The sender also indicates the data flow to which the transform set is to be applied. The sender must offer at least one transform set. The receiver then sends back a single transform set, which indicates the mutually agreed-upon transforms and algorithms for this particular IPSec session.
- A new Diffie-Hellman agreement may be done in phase 2, or the keys may be derived from the phase 1 shared secret.

Key Management

- handles key generation & distribution
- typically need 2 pairs of keys
  - 2 per direction for AH & ESP
- manual key management
  - sysadmin manually configures every system
- automated key management
  - automated system for on demand creation of keys for SA’s in large systems
  - has Oakley & ISAKMP elements

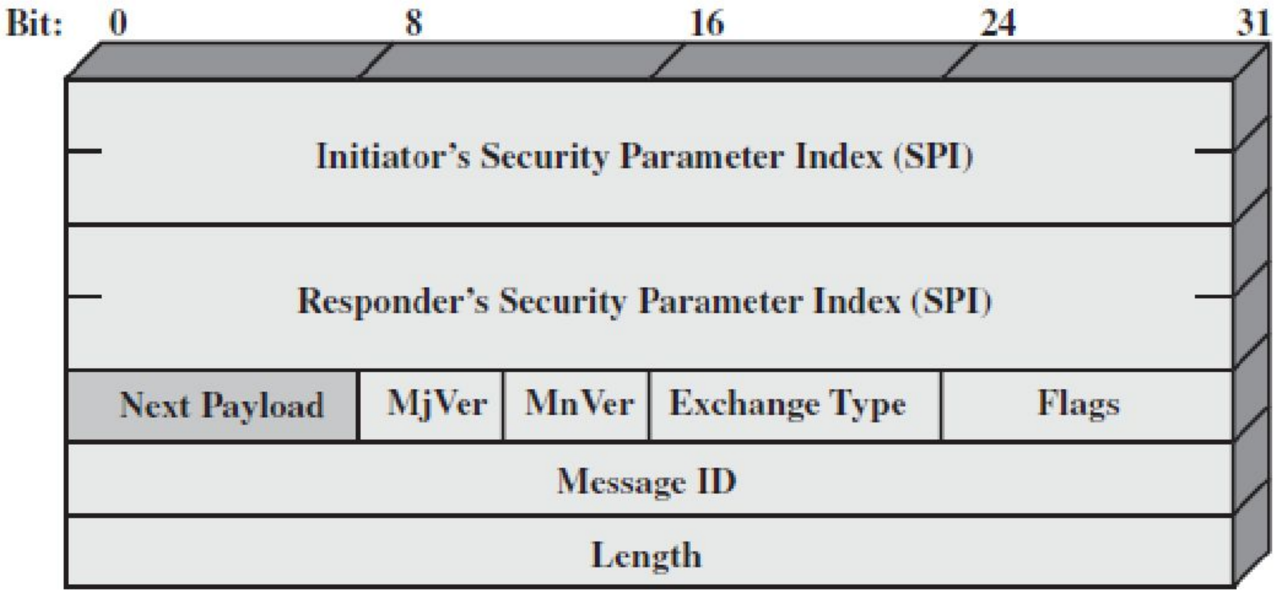
Oakley

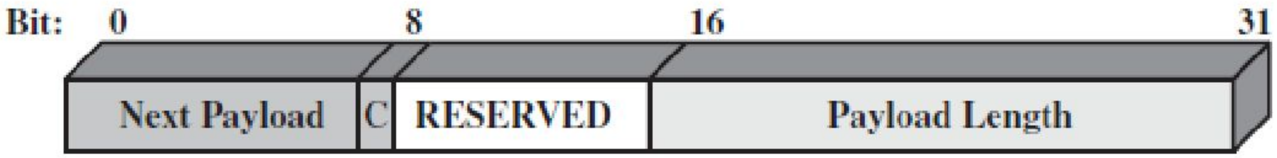
- a key exchange protocol
- based on Diffie-Hellman key exchange
- adds features to address weaknesses
  - cookies, groups (global params), nonces, DH key exchange with authentication
- can use arithmetic in prime fields or elliptic curve fields

ISAKMP

- Framework developed by the NSA mainly concerned with the details of Security Association management
- Consists of procedures and fields for
  - Authentication of peers
  - Negotiation, modification, deletion of Security Associations
  - key generation techniques
  - threat mitigation (e.g. DoS, replay attacks)
- Is distinct from key exchange protocols so details of managing security associations are separated from details of managing key exchange
- An implementation requires a key exchange protocol like IKE
- Common implementation is OAKLEY, a key agreement protocol using DH

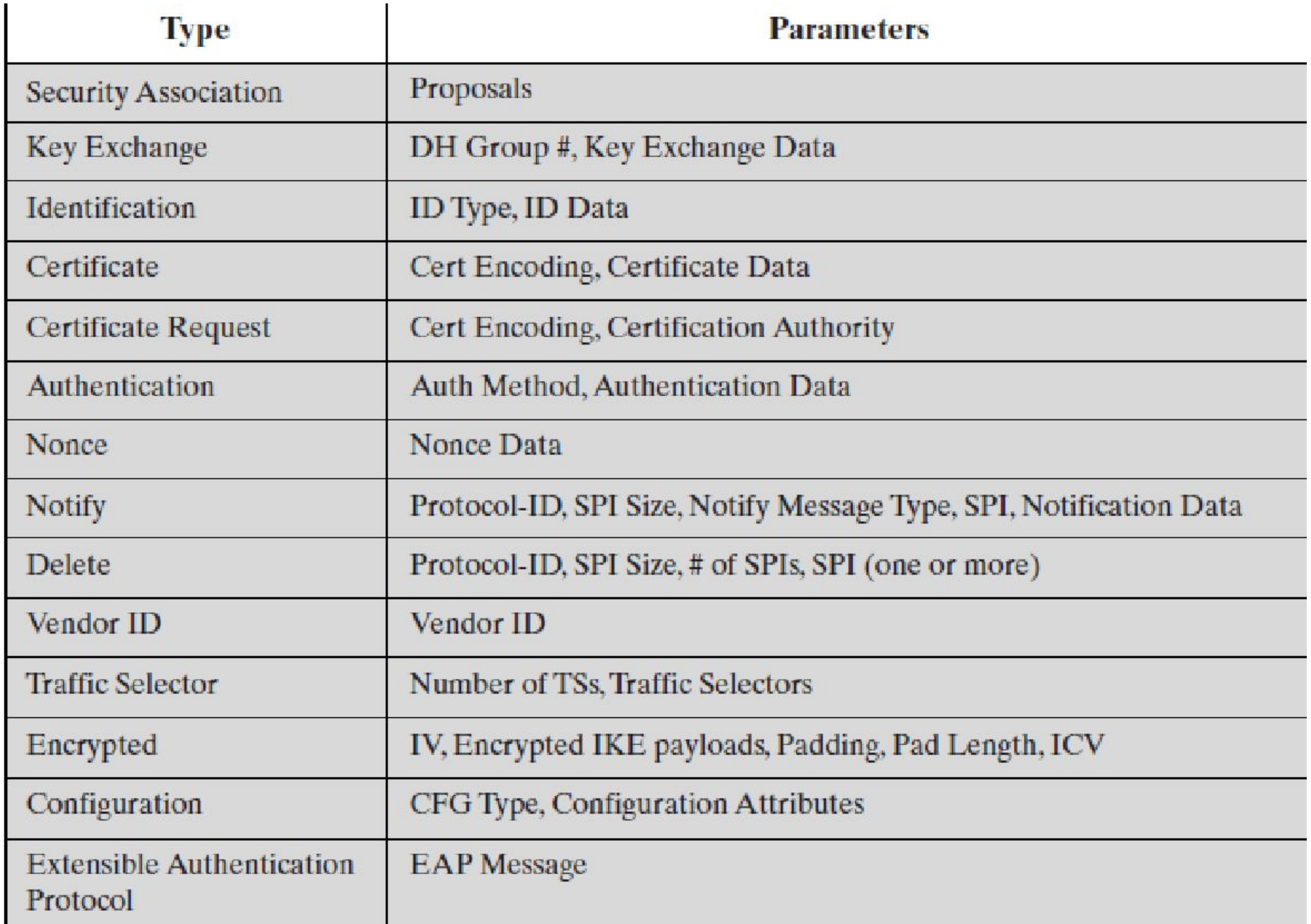
ISAKMP


Bit: 0 8 16 24 31
Initiator's Security Parameter Index (SPI)
Responder's Security Parameter Index (SPI)
Next Payload
MjVer
MnVer
Exchange Type
Flags
Message ID
Length
(a) IKE header



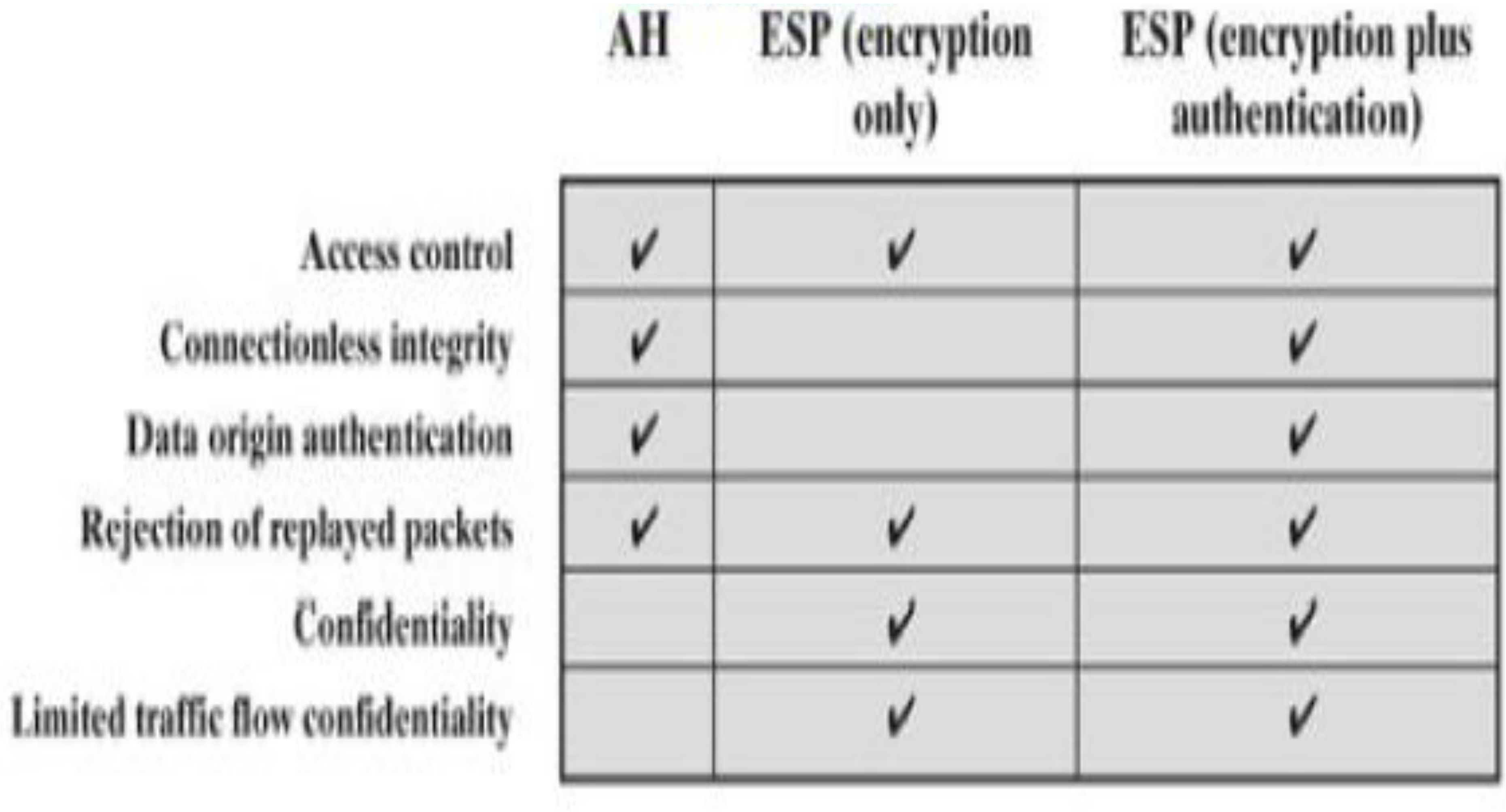
(b) Generic Payload header

IKE Payload



Type	Parameters
Security Association	Proposals
Key Exchange	DH Group #, Key Exchange Data
Identification	ID Type, ID Data
Certificate	Cert Encoding, Certificate Data
Certificate Request	Cert Encoding, Certification Authority
Authentication	Auth Method, Authentication Data
Nonce	Nonce Data
Notify	Protocol-ID, SPI Size, Notify Message Type, SPI, Notification Data
Delete	Protocol-ID, SPI Size, # of SPIs, SPI (one or more)
Vendor ID	Vendor ID
Traffic Selector	Number of TSs, Traffic Selectors
Encrypted	IV, Encrypted IKE payloads, Padding, Pad Length, ICV
Configuration	CFG Type, Configuration Attributes
Extensible Authentication Protocol	EAP Message

IP Security Services

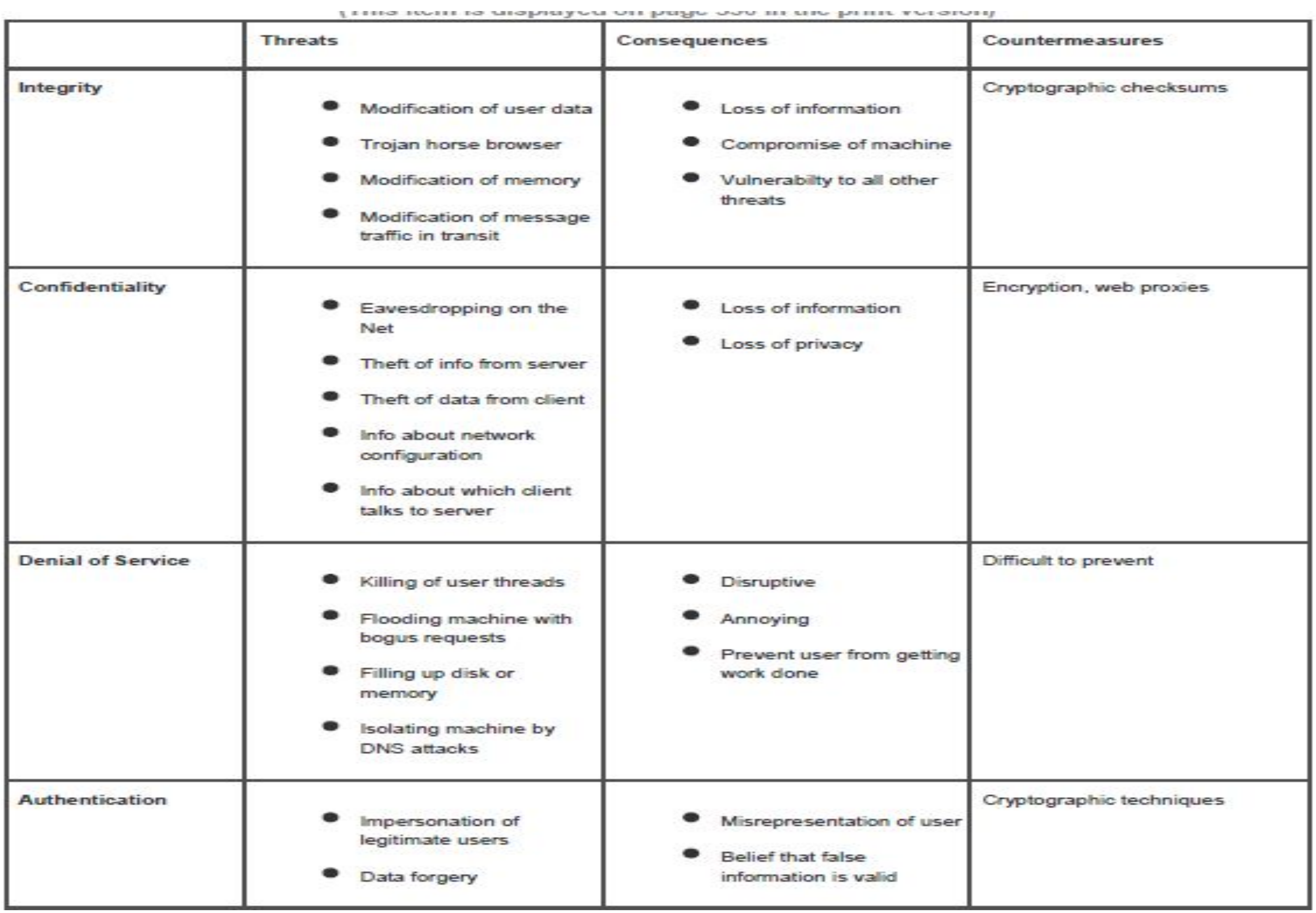


	AH	ESP (encryption only)	ESP (encryption plus authentication)
Access control	✓	✓	✓
Connectionless integrity	✓		✓
Data origin authentication	✓		✓
Rejection of replayed packets	✓	✓	✓
Confidentiality		✓	✓
Limited traffic flow confidentiality		✓	✓

Why Web Security?

- The Web is vulnerable to attacks on the Web servers over the Internet.
- The Web is increasingly serving as a highly visible outlet for corporate and product information and as the platform for business transactions. Reputations can be damaged and money can be lost if the Web servers are subverted.
- Although Web browsers are very easy to use, Web servers are relatively easy to configure and manage, and Web content is increasingly easy to develop, the underlying software is extraordinarily complex. This complex software may hide many potential security flaws.
- Casual and untrained (in security matters) users are common clients for Web-based services. Such users are not necessarily aware of the security risks that exist and do not have the tools or knowledge to take effective countermeasures.

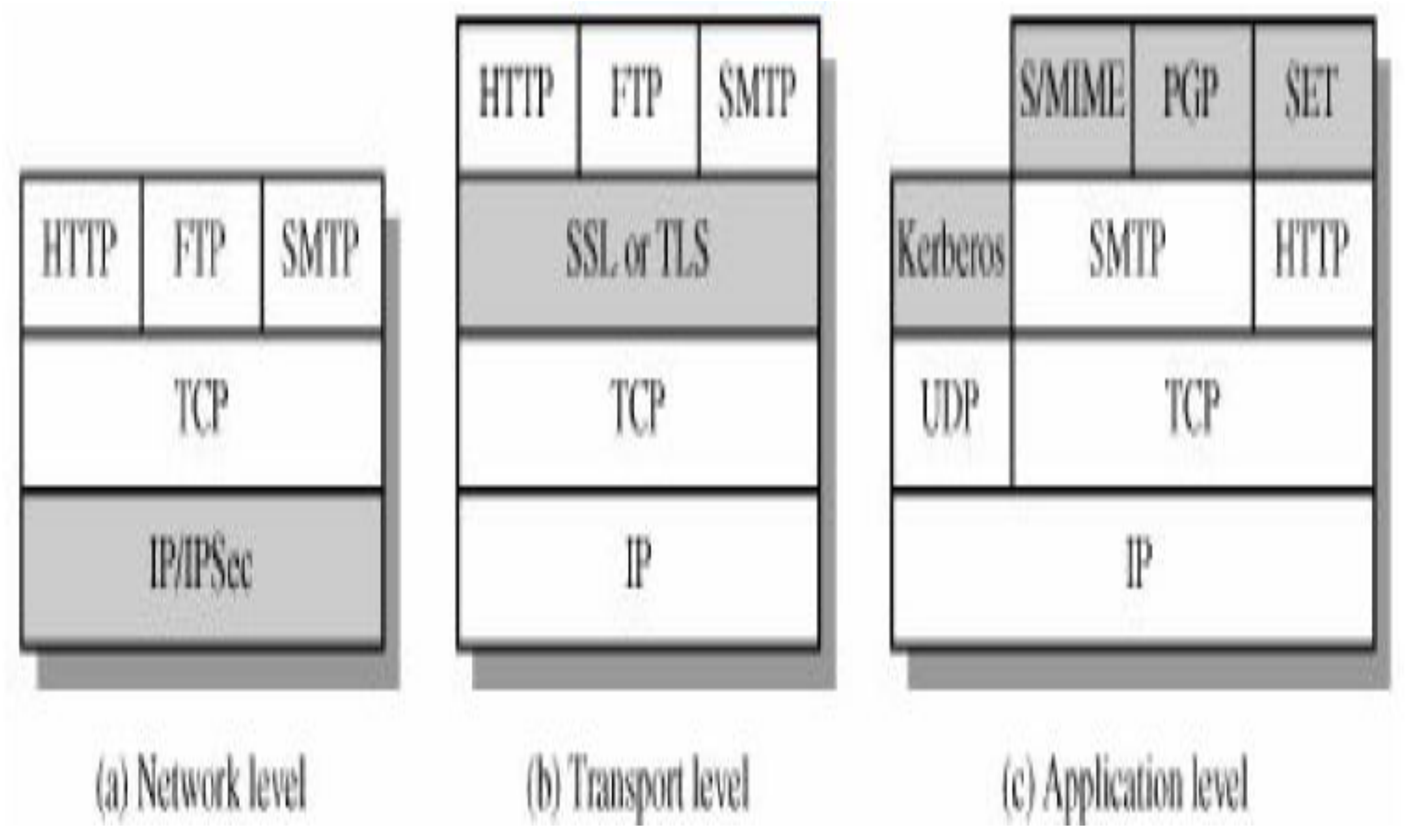
Web Security Threats



(This item is displayed on page 556 in the print version)

	Threats	Consequences	Countermeasures
Integrity	• Modification of user data • Trojan horse browser • Modification of memory • Modification of message traffic in transit	• Loss of information • Compromise of machine • Vulnerabilty to all other threats	Cryptographic checksums
Confidentiality	• Eavesdropping on the Net • Theft of info from server • Theft of data from client • Info about network configuration • Info about which client talks to server	• Loss of information • Loss of privacy	Encryption, web proxies
Denial of Service	• Killing of user threads • Flooding machine with bogus requests • Filling up disk or memory • Isolating machine by DNS attacks	• Disruptive • Annoying • Prevent user from getting work done	Difficult to prevent
Authentication	• Impersonation of legitimate users • Data forgery	• Misrepresentation of user • Belief that false information is valid	Cryptographic techniques

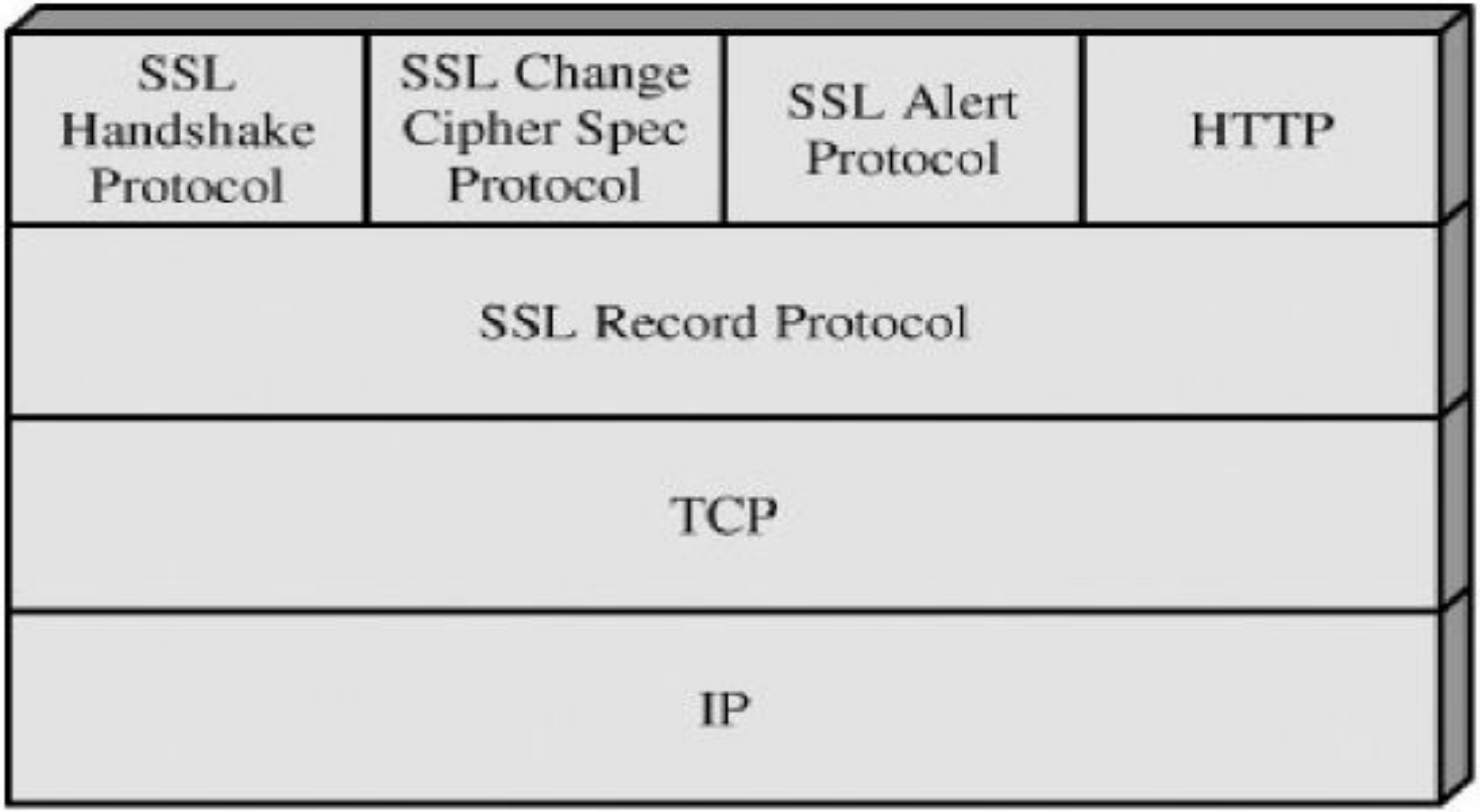
Relative Locations of Security in TCP/IP Protocol Stack



SSL

- Netscape originated SSL Version 3 of the protocol was designed with public review and input from industry and was published as an Internet draft document.
- Subsequently, when a consensus was reached to submit the protocol for Internet standardization, the TLS working group was formed within IETF to develop a common standard.
- **Transport Layer Security** (**TLS**) is the successor to **SSL**.
- This first published version of TLS can be viewed as essentially an SSLv3.1 and is very close to and backward compatible with SSLv3.

SSL Architecture



SSL Architecture

- Two important SSL concepts are the **SSL session and the SSL connection**, which are defined in the specification as follows:
- **Connection: A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For SSL,** such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.
- **Session: An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol.**

  Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

Connection and session difference

- A session between two systems is an association that can last for a long time; a connection can be established and broken several times during a session.
- Some of the security parameters are created during the session establishment and are in effect until the session is terminated. Some of the security parameters must be recreated (or occasionally resumed) for each connection.“
- Connection is really a living communication channel, and session is usually a set of negotiated cryptanalysis guidelines.
- You can close connection, but keep session, even store it to disk, and subsequently resume it

SSL Architecture Cont..

A session state is defined by the following parameters

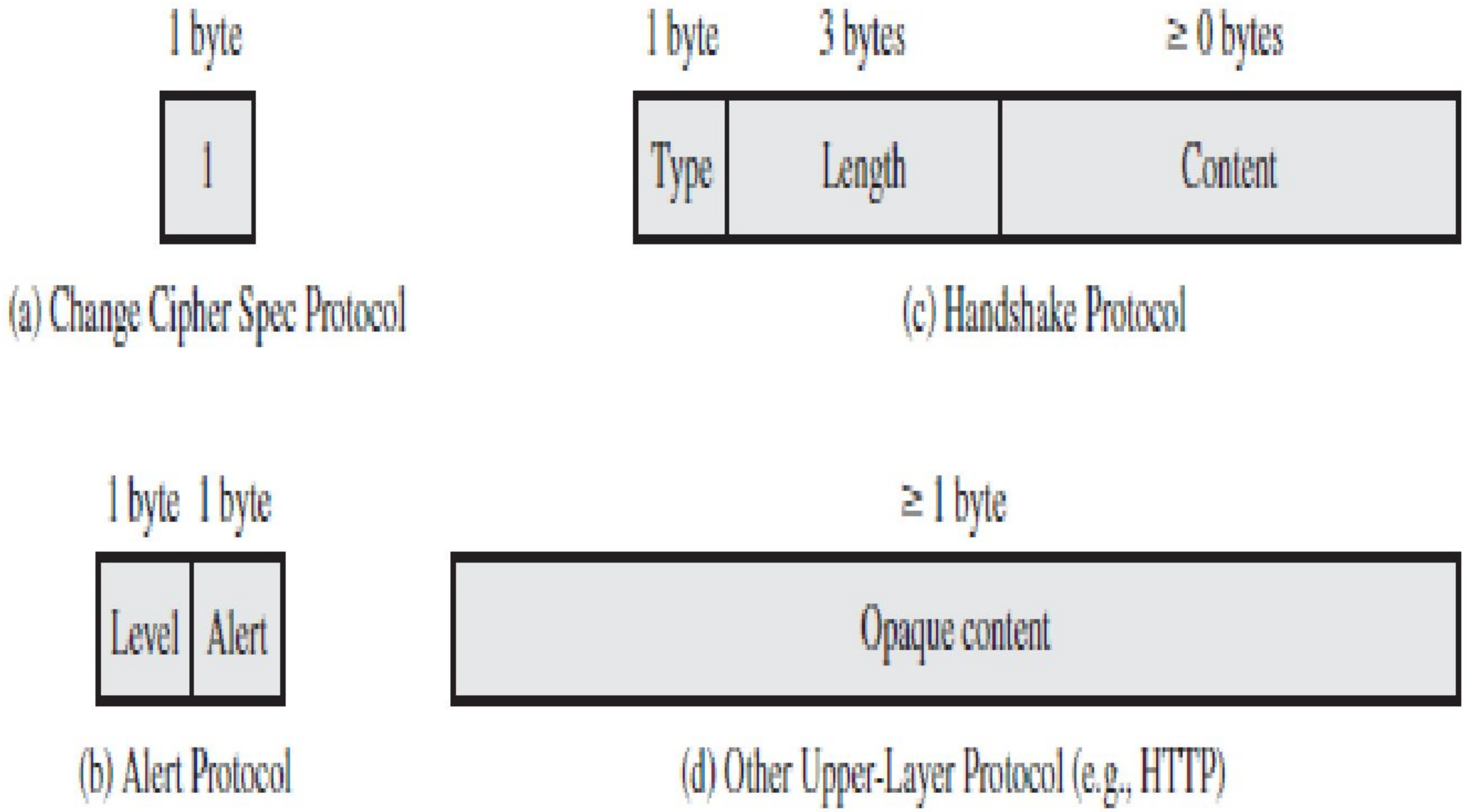
- **Session identifier: An arbitrary byte sequence** chosen by the server to identify an active or resumable session state.
- **Peer certificate:** An X509.v3 certificate of the peer. This element of the state may be null.
- **Compression method:** The algorithm used to compress data prior to encryption.
- **Cipher spec:** Specifies the bulk data encryption algorithm (such as null, AES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash_size.
- **Master secret:** 48-byte secret shared between the client and server.

SSL Architecture Cont..

A connection state is defined by the following parameters

- **Server and client random:** Byte sequences that are chosen by the server and client for each connection.
- **Server write MAC secret:** The secret key used in MAC operations on data sent by the server.
- **Client write MAC secret:** The secret key used in MAC operations on data sent by the client.
- **Server write key:** The conventional encryption key for data encrypted by the server and decrypted by the client.
- **Client write key:** The conventional encryption key for data encrypted by the client and decrypted by the server.
- **Initialization vectors: When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This** field is first initialized by the SSL Handshake Protocol. Thereafter the final ciphertext block from each record is preserved for use as the IV with the following record.

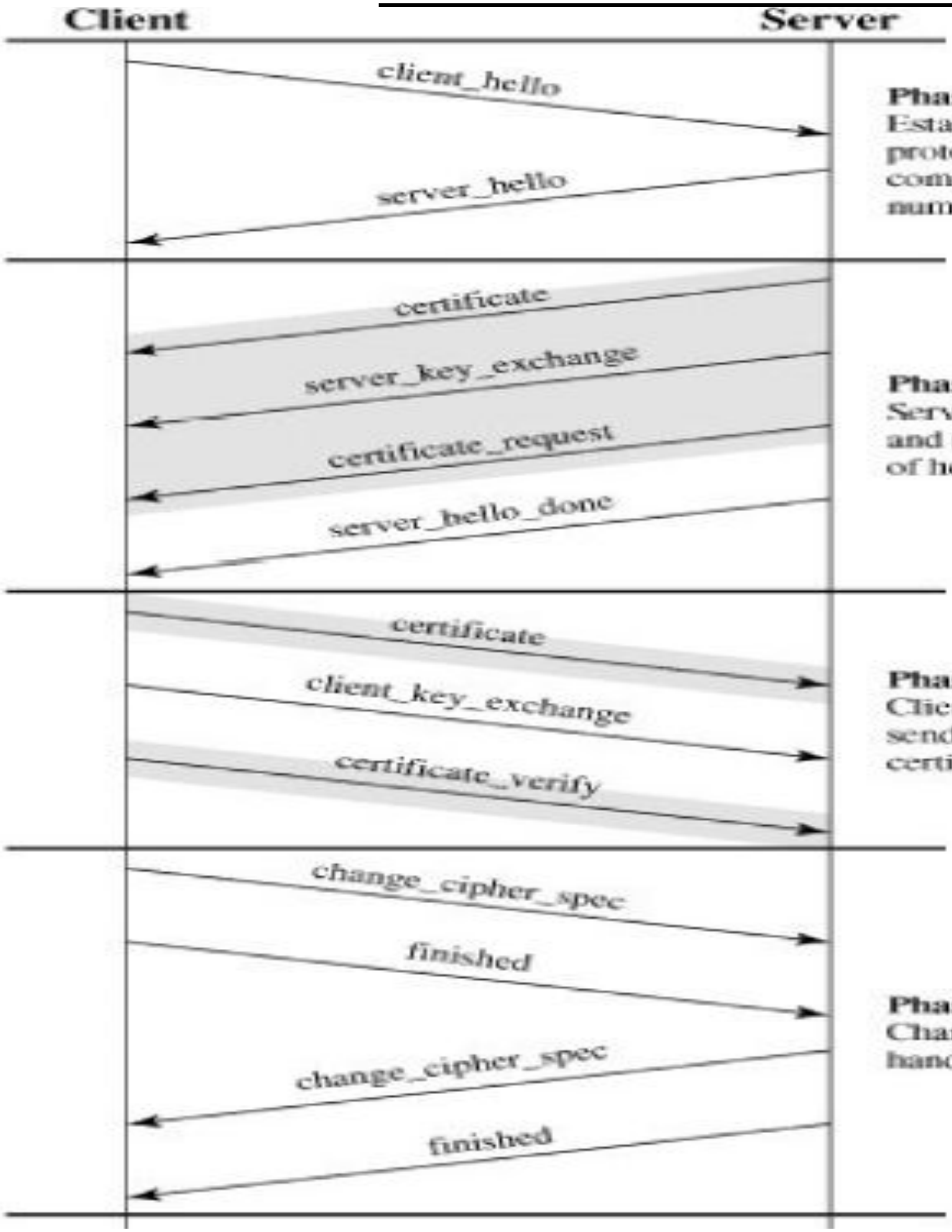
SSL Record Protocol format



SSL Handshake Protocol

- allows server & client to:
  - authenticate each other
  - to negotiate encryption & MAC algorithms
  - to negotiate cryptographic keys to be used
- comprises a series of messages in phases
  - Establish Security Capabilities
  - Server Authentication and Key Exchange
  - Client Authentication and Key Exchange
  - Finish

Handshake Protocol

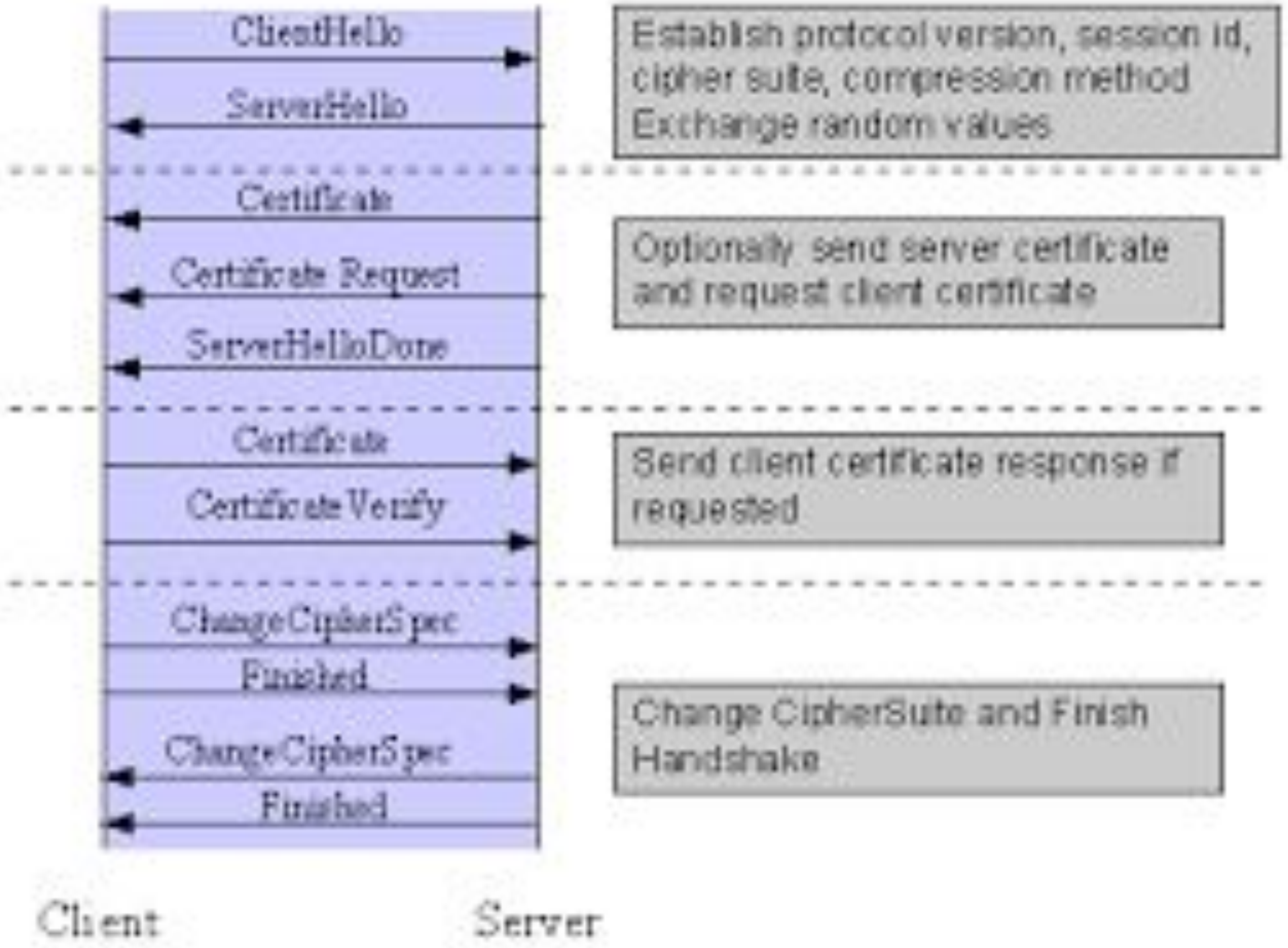


Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.

Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.

Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.

Phase 4
Change cipher suite and finish handshake protocol.


ClientHello
ServerHello
Establish protocol version, session id, cipher suite, compression method Exchange random values
Certificate
Certificate Request
ServerHelloDone
Optionally send server certificate and request client certificate
Certificate
CertificateVerify
Send client certificate response if requested
ChangeCipherSpec
Finished
ChangeCipherSpec
Finished
Change CipherSuite and Finish Handshake
Client
Server

SSL-Handshake Protocol

- **Handshake Protocol divided into 4 phases:**
  1. Establish Security Capabilities
  2. Server Authentication and key Exchange
  3. Client Authentication and key Exchange
  4. Change *CipherSpec* and Finish

SSL Handshake

Phase 1
Security capabilities

1. C □ S: **C**LIENT**H**ELLO
2. S □ C: **S**ERVER**H**ELLO

Phase 2
Optional server messages

[**C**ERTIFICATE]
[**S**ERVER**K**EY**E**XCHANGE]
[**C**ERTIFICATE**R**EQUEST]
SERVER**H**ELLO**D**ONE

Phase 3
Client key exchange

3. C □ S: [**C**ERTIFICATE]
CLIENT**K**EY**E**XCHANGE
[**C**ERTIFICATE**V**ERIFY]

Phase 4
Change cipher suite

CHANGE**C**IPHER**S**PEC
FINISH

4. S □ C: **C**HANGE**C**IPHER**S**PEC
FINISH

SSL Handshake

- CLIENTHELLO message is sent by the client
  - When the **client wants to establish a TCP connection** to the server,
  - When a **HELLOREQUEST message is re**ceived, or
  - When client wants to **renegotiate security parameters** of an existing connection
- Message content:
  - Number of highest SSL understood by the client
  - Client’s random structure (32-bit timestamp and 28-byte pseudorandom number)
  - Session ID client wishes to use (ID is empty for existing sessions)
  - List of cipher suits the client supports
  - List of compression methods the client supports

SSL Handshake

- Server processes CLIENTHELLO message
- Server Respond to client with SERVERHELLO message:
    - Server version number: lower version of that suggested by the client and the highest supported by the server
    - Server's random structure: 32-bit timestamp and 28-byte pseudorandom number
    - Session ID: corresponding to this connection
    - Cipher suite: selected by the server for client's list
    - Compression method: selected by the server from client's list

SSL Handshake

Optional messages: Phase 2– server authentication

- CERTIFICATE:
  - If the server is using certificate-based authentication
  - May contain RSA public key □ good for key exchange
- SERVERKEYEXCHANGE:
  - If the client does not have certificate, has certificate that can only be used to verify digital signatures, or uses FORTEZZA token-based key exchange (not recommended)
- CERTIFICATEREQUEST:
  - Server may request personal certificate to authenticate a client

SSL Handshake

- Client processing:
  - Verifies site certification
    - Valid site certification if the server’s name matches the host part of the URL the client wants to access
  - Checks security parameters supplied by the SERVERHELLO

SSL Handshake

- Client messages: Phase 3 – client authentication and key exchange
  - CERTIFICATE
    - If server requested a client authentication, client sends
  - CLIENTKEYEXCHANGE
    - Format depends on the key exchange algorithm selected by the server
      - RSA: 48-byte premaster secret encrypted by the server’s public key
      - Diffie-Hellman: public parameters between server and client in SERVERKEYEXCHANGE and CLIENTKEYEXCHANGE msgs.
      - FORTEZZA: token-based key exchange based on public and private parameters
    - Premaster key is transformed into a 48-byte master secret, stored in the session state

SSL Handshake

- Client messages:
  - CERTIFICATEVERIFY
    - If client authentication is required
    - Provides explicit verification of the use’s identity (personal certificate)
  - CHANGECIPHERSPEC
    - Completes key exchange and cipher specification
  - FINISH
    - Encrypted by the newly negotiated session key
    - Verifies that the keys are properly installed in both sites

SSL Handshake

- Phase 4: finish
- Server finishes handshake by sending CHANGECIPHERSPEC and FINISH messages

- After SSL handshake <u>completed</u> a secure connection is established to send <u>application data encapsulated in SSL Record Protocol</u>

Change Cipher Spec protocol

- ChangeCipherSpec messages are used in SSL to indicate, that the communication is shifted from unencrypted to encrypted.
- This message informs that, the data following will be encrypted with the shared secret.
- The change cipher spec protocol is used to change the encryption being used by the client and server. It is normally used as part of the handshake process to switch to symmetric key encryption.
- The CCS protocol is a single message that tells the peer that the sender wants to change to a new set of keys, which are then created from information exchanged by the handshake protocol.

Change Cipher Spec.Protocol

- **The Change Cipher Spec Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest.** This protocol consists of a single message which consists of a single byte with the value 1.

- **The sole purpose of this message is to cause the pending state to be copied into the current state,** which updates the cipher suite to be used on this connection.

Alert Protocol

- **The Alert Protocol is used to convey SSL-related alerts to the peer entity.** As with other applications that use SSL, alert messages are compressed and encrypted, as specified by the current state.
- **Each message in this protocol consists of two bytes**. The first byte takes the value warning(1) or fatal(2) to convey the severity of the message. **If the level is fatal, SSL immediately terminates the connection.** Other connections on the same session may continue, but no new connections on this session may be established.

Fatal Alerts

- **unexpected_message:** An inappropriate message was received.
- **bad_record_mac:** An incorrect MAC was received.
- **decompression_failure:** The decompression function received improper input (e.g., unable to decompress or decompress to greater than maximum allowable length).
- **handshake_failure:** Sender was unable to negotiate an acceptable set of security parameters given the options available.
- **illegal_parameter:** A field in a handshake message was out of range or inconsistent with other fields.

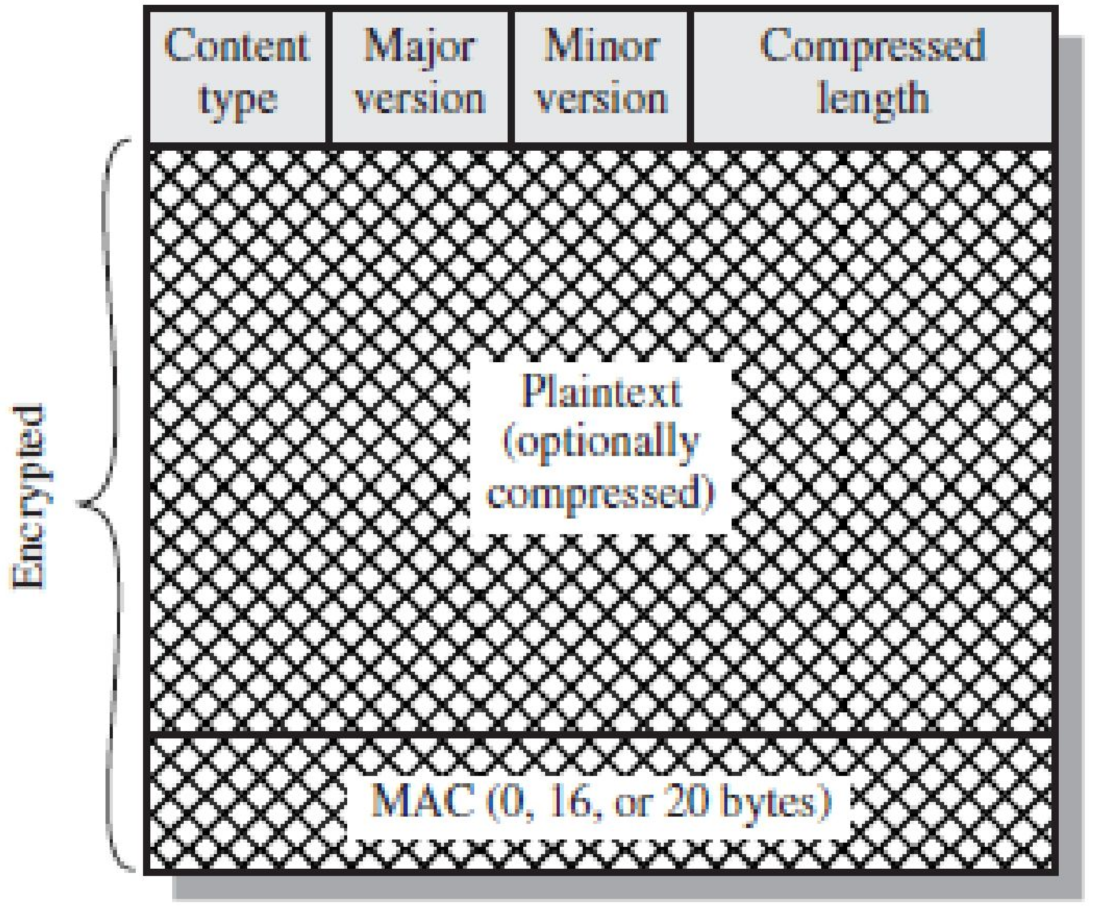
Other alerts

- **close_notify** alert before closing the write side of a connection.
- **no_certificate:** May be sent in response to a certificate request if no appropriate certificate is available.
- **bad_certificate:** A received certificate was corrupt (e.g., contained a signature that did not verify).
- **unsupported_certificate:** The type of the received certificate is not supported.
- **certificate_revoked:** A certificate has been revoked by its signer.
- **certificate_expired:** A certificate has expired

SSL Record Protocol

- The SSL Record Protocol provides two services for SSL connections:
- **Confidentiality:** The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.
- **Message Integrity:** The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

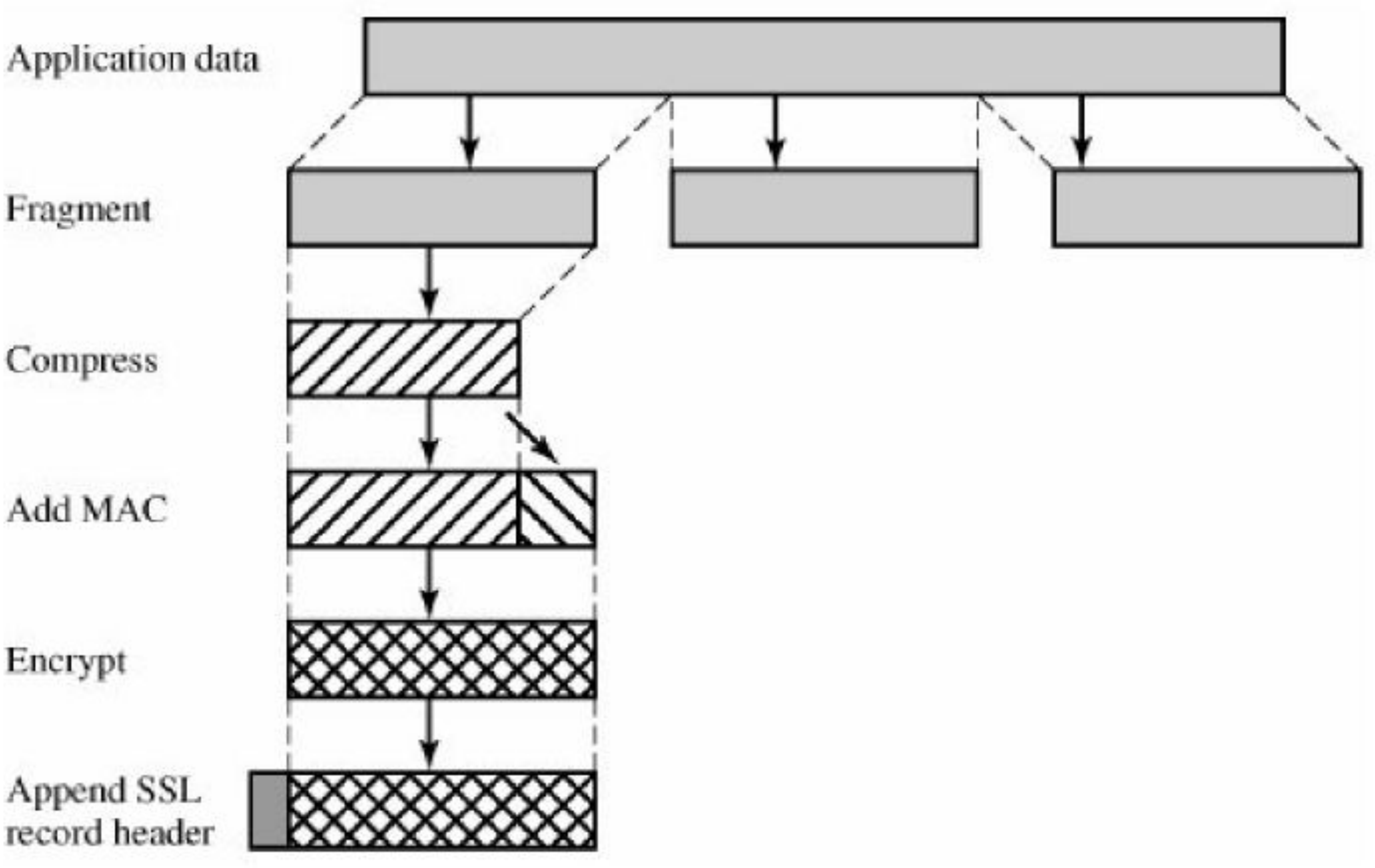
SSL Record format



SSL Record format

- **Content Type (8 bits):** The higher-layer protocol used to process the enclosed fragment.
- **Major Version (8 bits):** Indicates major version of SSL in use. For SSLv3, the value is 3.
- **Minor Version (8 bits):** Indicates minor version in use. For SSLv3, the value is 0.
- **Compressed Length (16 bits):** The length in bytes of the plaintext fragment (or compressed fragment if compression is used).

SSL Record Protocol Operation



Introduction

- Transport Layer Security (TLS) [RFC2246]
- TLS provides transport layer security for Internet applications
- It provides for confidentiality and data integrity over a connection between two end points
- TLS operates on a reliable transport, such as TCP, and is itself layered into
  - TLS Record Protocol
  - TLS Handshake Protocol

TLS

- Advantage of TLS
  - applications can use it transparently to securely communicate with each other
  - TLS is visible to applications, making them aware of the cipher suites and authentication certificates negotiated during the set-up phases of a TLS session

TLS Record Protocol

- TLS Record Protocol layers on top of a reliable connection-oriented transport, such as TCP
- TLS Record Protocol
  - provides data confidentiality using symmetric key cryptography
  - provides data integrity using a keyed message authentication checksum (MAC)
- The keys are generated uniquely for each session based on the security parameters agreed during the TLS handshake

TLS Record Protocol

- Basic operation of the TLS Record Protocol
  1. read messages for transmit
  2. fragment messages into manageable chunks of data
  3. compress the data, if compression is required and enabled
  4. calculate a MAC
  5. encrypt the data
  6. transmit the resulting data to the peer

TLS Record Protocol

- At the opposite end of the TLS connection, the basic operation of the sender is replicated, but in the reverse order
  1. read received data from the peer
  2. decrypt the data
  3. verify the MAC
  4. decompress the data, if compression is required and enabled
  5. reassemble the message fragments
  6. deliver the message to upper protocol layers

TLS Handshake Protocol

- TLS Handshake Protocol is layered on top of the TLS Record Protocol
- TLS Handshake Protocol is used to
  - authenticate the client and the server
  - exchange cryptographic keys
  - negotiate the used encryption and data integrity algorithms before the applications start to communicate with each other

TLS Handshake protocol

- Similar to SSL Handshake protocol

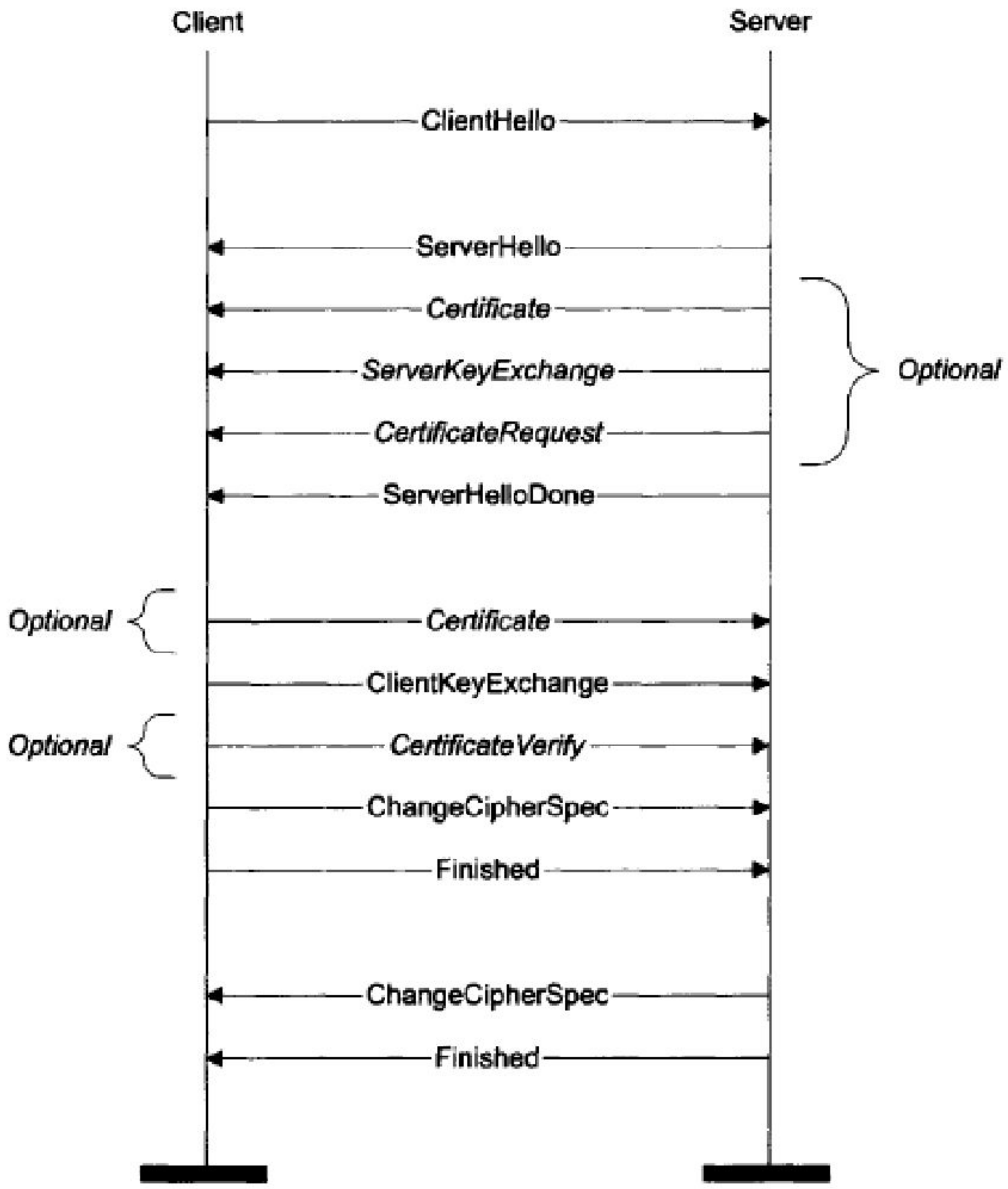


Figure 14.1 The TLS handshake.

TLS v/s SSL

- Find Difference between TLS and SSL

SSL

- Netscape originally developed the SSL protocol to transmit information privately, ensure message integrity, and guarantee the server identity.
- SSL works mainly through using public/private key encryption on data.
- It is commonly used on web browsers, but SSL can also be used with email servers also.

TLS

- The Internet Engineering Task Force (IETF) created TLS (Transport Layer Security) as the successor to SSL.
- It is most often used as a setting in email programs

Key Differences

- The TLS protocol does not support Fortezza/DMS cipher suites while SSL supports Fortezza. Also, the TLS standardization process makes it much easier to define new cipher suites.
- In SSL to create a master secret, the message digest of the pre-master secret is used. In contrast, TLS uses a pseudorandom function to create master secret.
- The SSL record protocol adds MAC (Message Authentication Code) after compressing each block and encrypts it. As against, TLS record protocol uses HMAC (Hash-based Message Authentication Code).
- The "No certificate" alert message is included in SSL. On the other hand, TLS eliminates alert description (No certificate) and adds a dozen other values.
- In the TLS certificate verify the message, the MD5 and SHA-1 hashes are calculated only over handshake messages. On the contrary, in SSL the hash calculation also include the master secret and pad.
- As with the finished message in TLS, created by applying the PRF to the master key and handshake messages. Whereas in SSL, it's created by applying message digest to the master key and handshake messages.