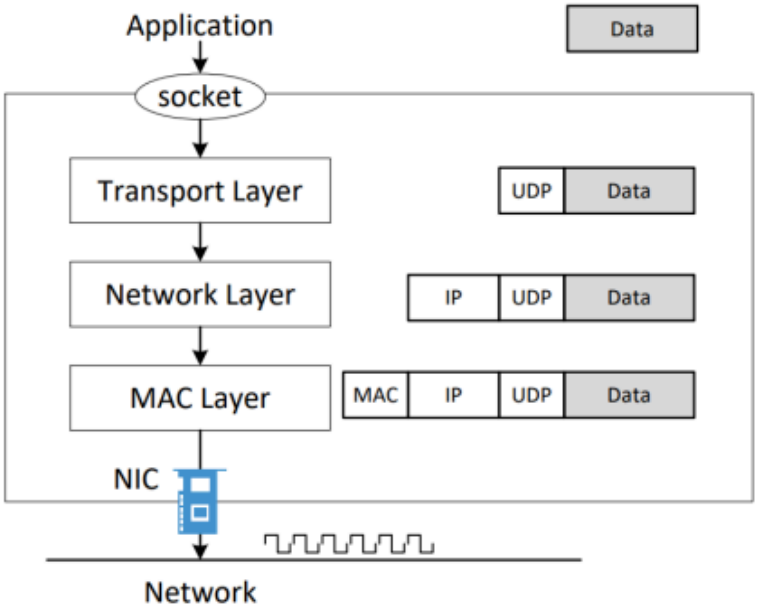
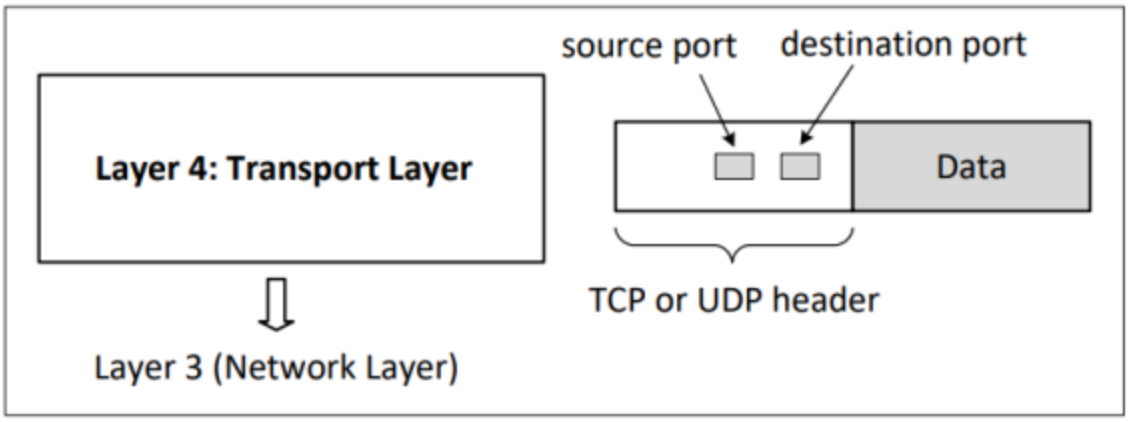
Network Security Basics

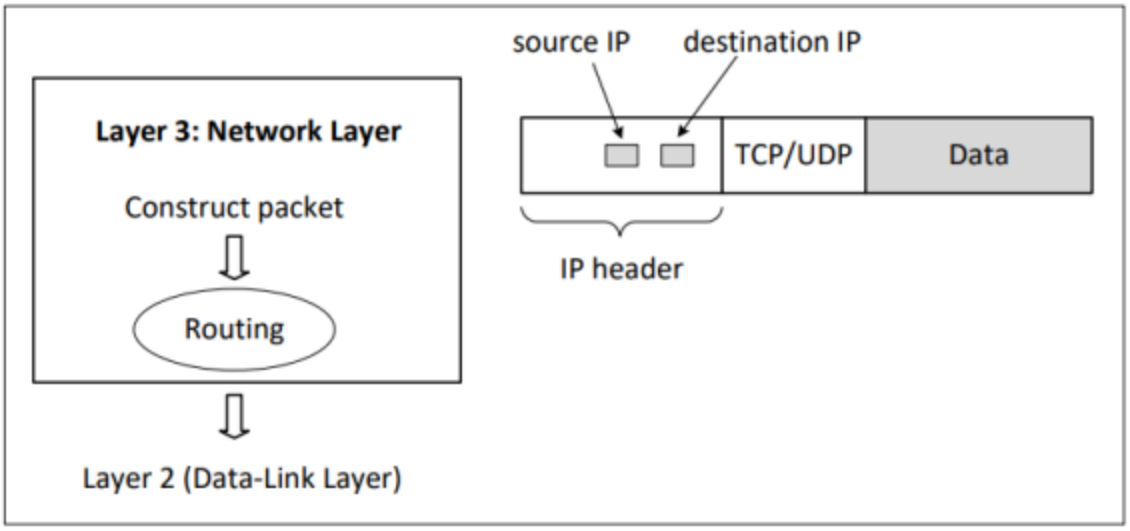
#week2 #security

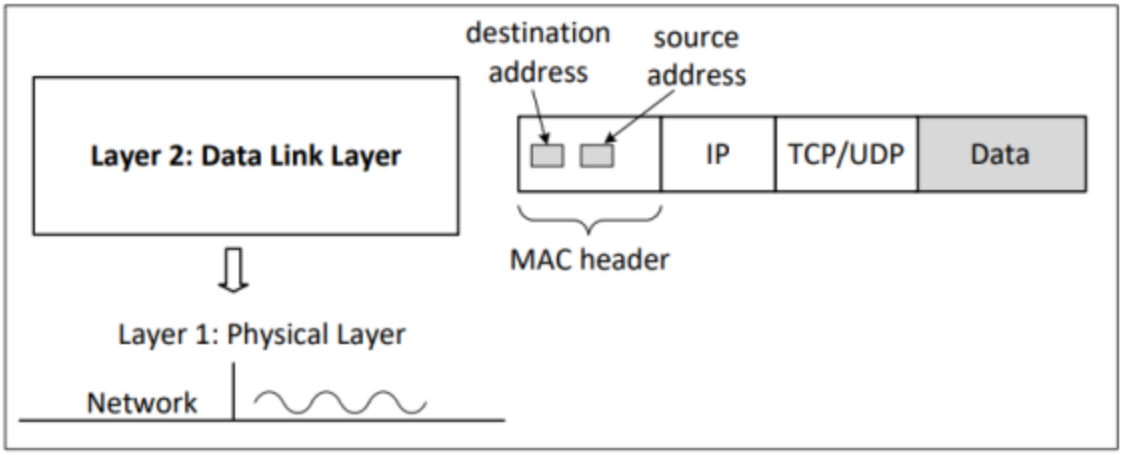
Network Recap

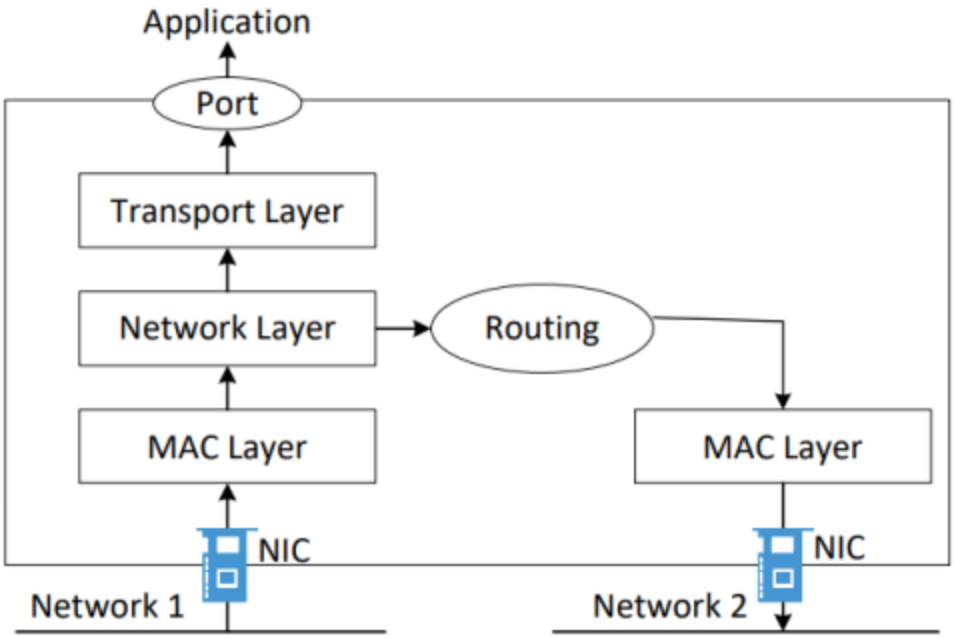
Networks consist of several layers, each layer also adds a section to a packet

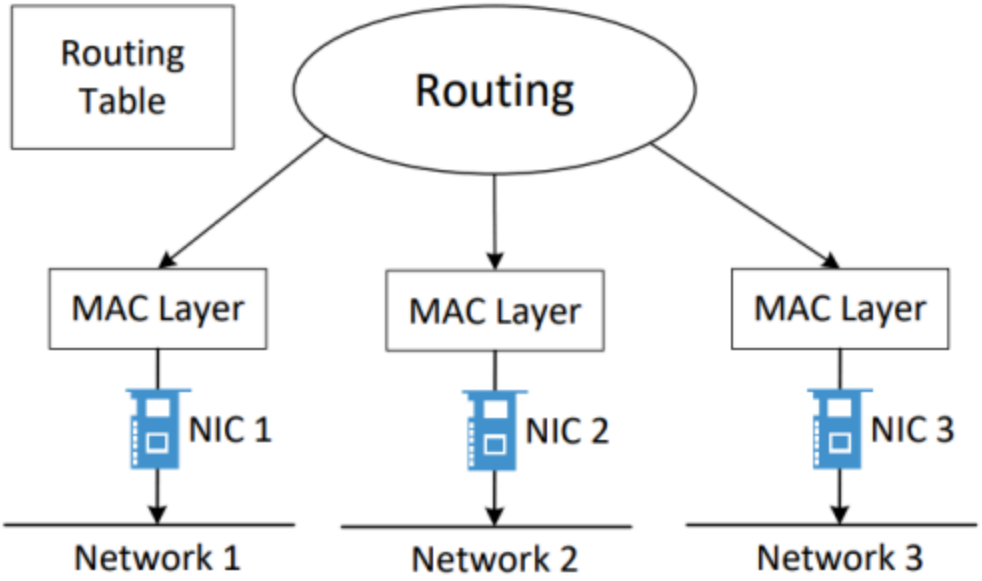





Layer 3: Network Layer
Construct packet
Routing
Layer 2 (Data-Link Layer)
source IP
destination IP
TCP/UDP
Data
IP header

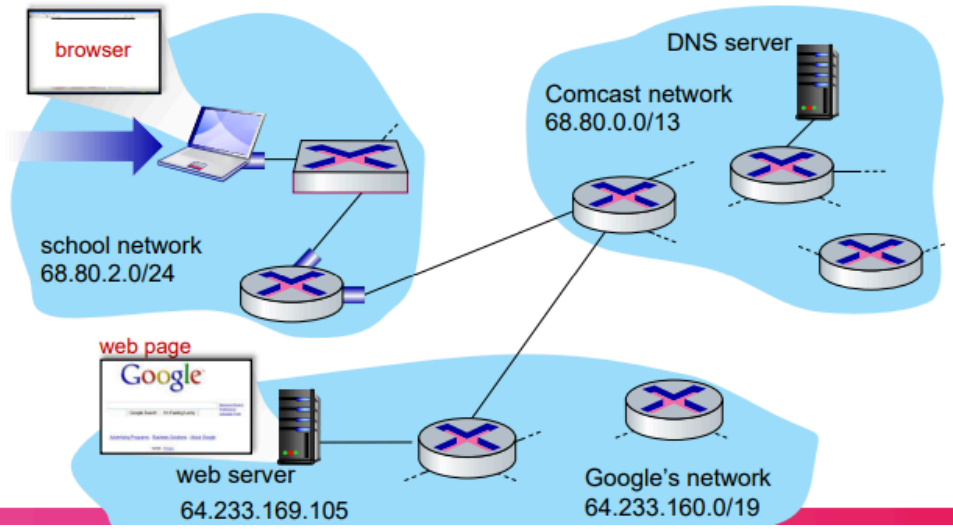

Layer 2: Data Link Layer
Layer 1: Physical Layer
Network
destination address
source address
IP
TCP/UDP
Data
MAC header





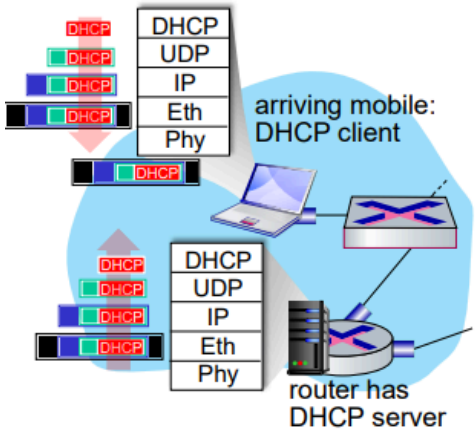
At a high level the internet consists of multiple networks and routers that a packet travels through

scenario:

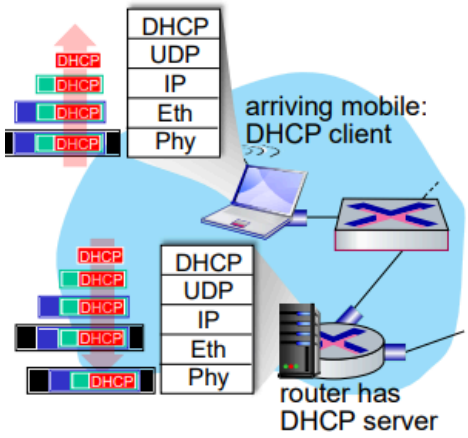


- arriving mobile client attaches to network …
- requests web page: www.google.com

Sounds simple!



- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP



- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

IP Addresses

Has both numbers and binary representation

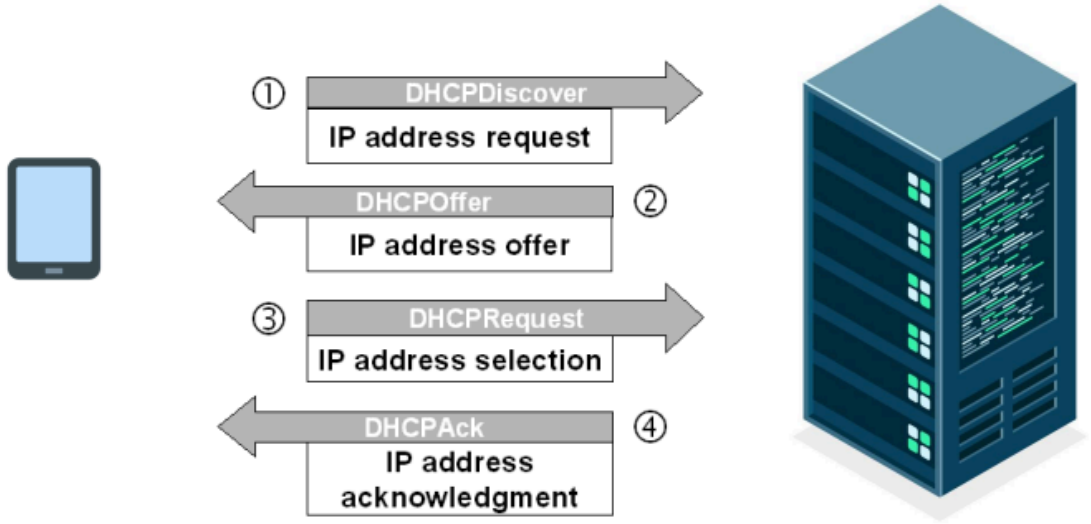
```
Class A                     |<--      Host ID     -->|
  0.  0.  0.  0 = 00000000.00000000.00000000.00000000
127.255.255.255 = 01111111.11111111.11111111.11111111

Class B                               |<-- Host ID -->|
128.  0.  0.  0 = 10000000.00000000.00000000.00000000
191.255.255.255 = 10111111.11111111.11111111.11111111

Class C                                        |HostID|
192.  0.  0.  0 = 11000000.00000000.00000000.00000000
223.255.255.255 = 11011111.11111111.11111111.11111111

Class D                 |<--   Address Range       -->|
224.  0.  0.  0 = 11100000.00000000.00000000.00000000
239.255.255.255 = 11101111.11111111.11111111.11111111

Class E                 |<--   Address Range       -->|
240.  0.  0.  0 = 11110000.00000000.00000000.00000000
255.255.255.255 = 11111111.11111111.11111111.11111111
```

192.168.60.5/**24**

Indicate the first 24 bits are network ID

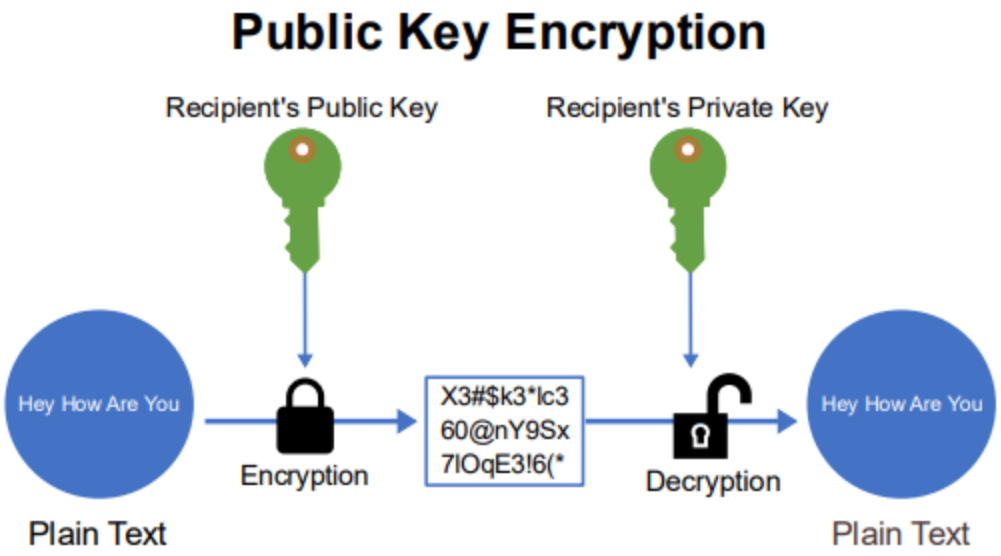
DNS automatically assign IP addresses

- DHCP: Dynamic Host Configuration Protocol



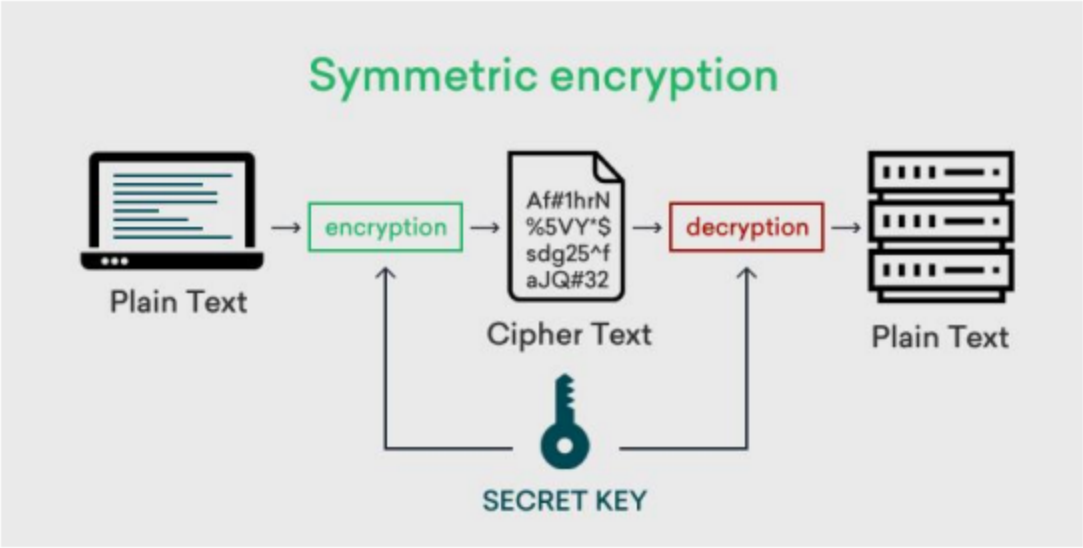
Application Layer Security

- **Authentication** and **Integrity** are often provided by **public-key cryptography** and secure transmission of message digests (e.g., digitally signed hash values)
- **Public-key cryptography**, also known as asymmetric cryptography, uses pairs of mathematically linked keys – a public key and a private key – for secure data operations, such as encrypting messages or digitally signing them
- **Confidentiality** is provided by **symmetric key cryptography**
- **Symmetric key cryptography** is a system of encrypting data where the same secret key is used for both encryption and decryption.



!

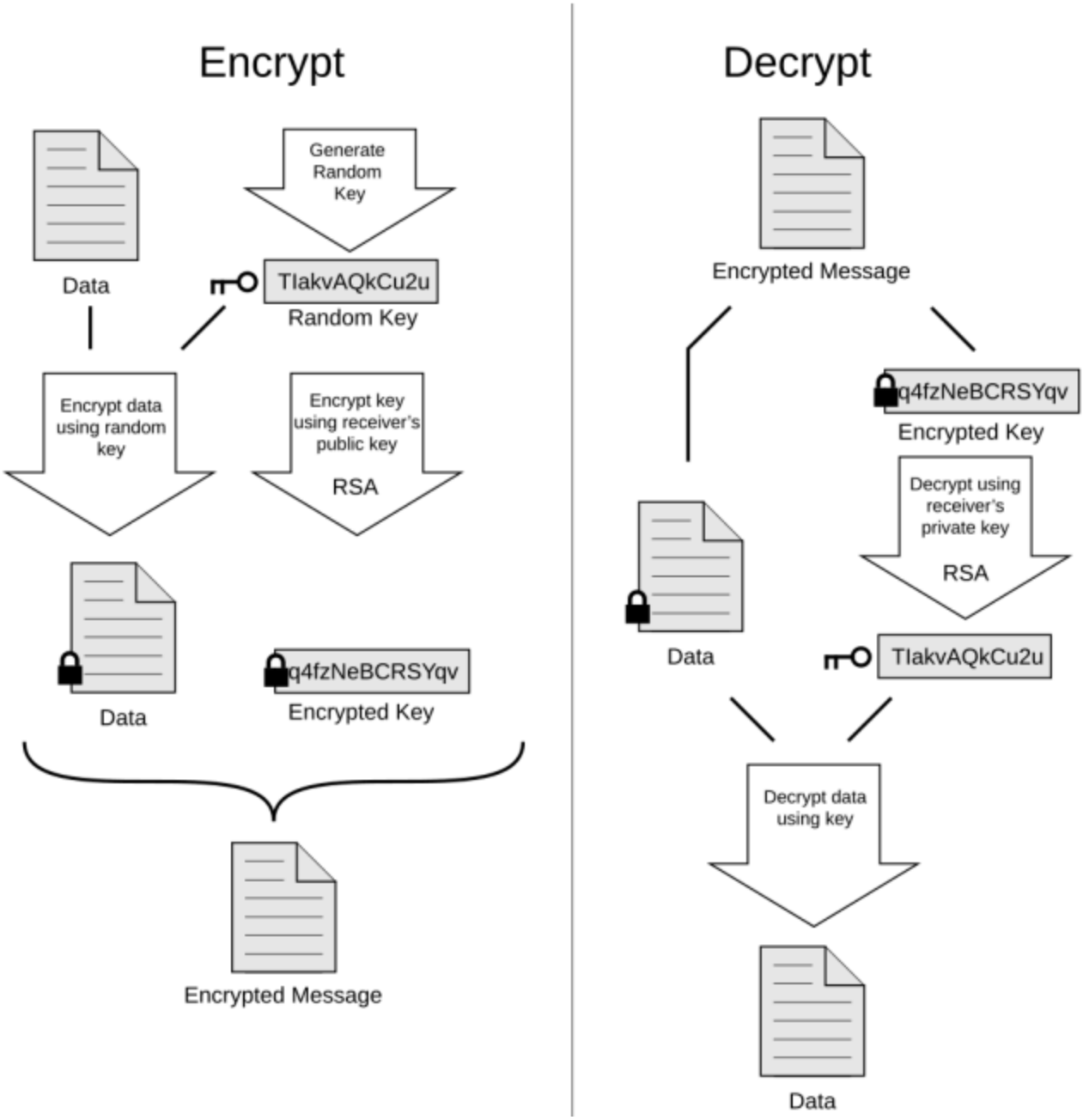
For instance, when contacting a bank you have your password and the bank would have their password



The difference here is that they key is kept secret

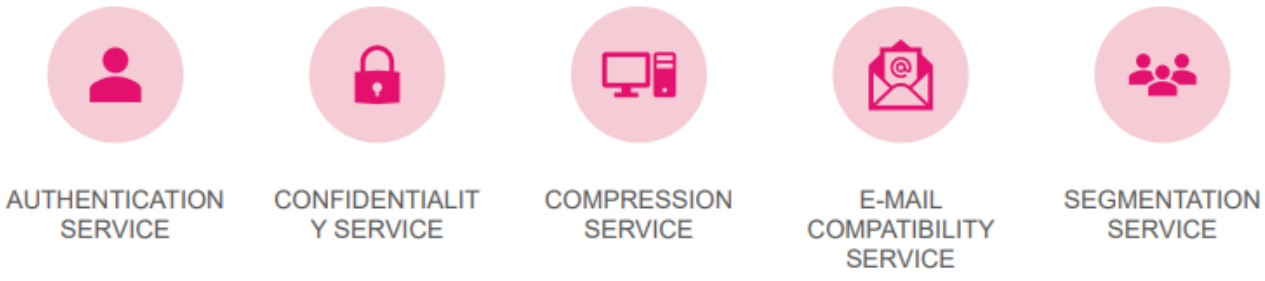
Both of these are used at the same time to keep a network secure

PGP: Pretty Good Privacy



Originally a way to connect universities and allowing them to share research., now it's widely used for email and data storage security.

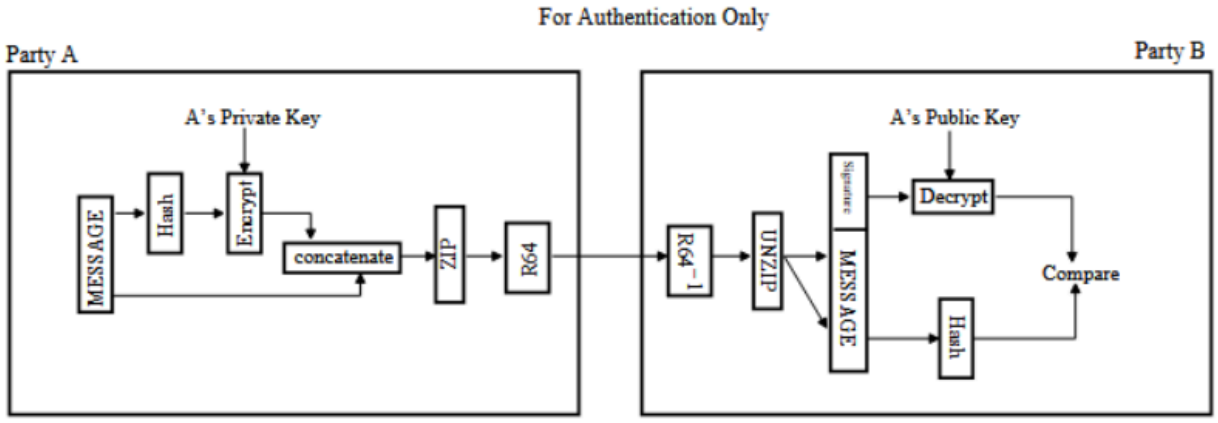
PGP offers fiver services



Nowadays an open source version of PGP is more widely used, known as OpenPGP

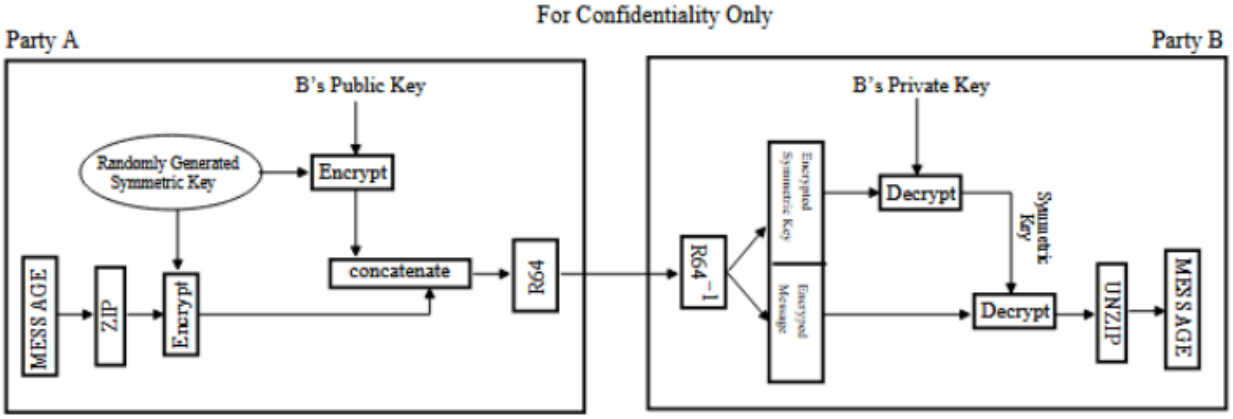
PGP uses public-key encryption for authentication

- The sender creates a 160 bit SHA message which is encrypted with the sender's private key which is prepdended to the message
- The recieved usesers the senders public key to decrypt the recieved message

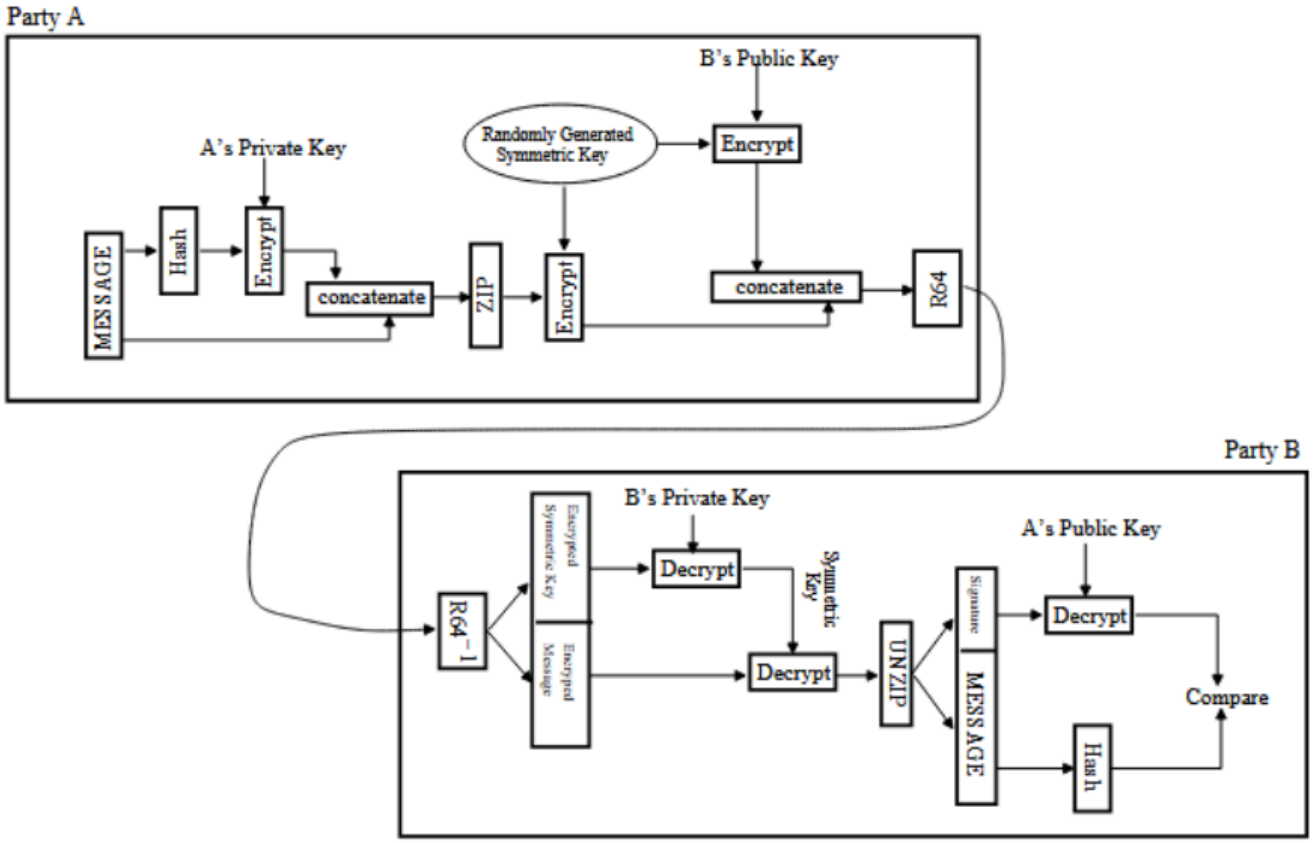


Symmetric key encryption is used for confidentiality, users can choose from several block-cihper algorithms which are used in Cipher Feedback Mode (CFB).

- A randomly generated 128-bit session key is used to encrypt the email message content
- The session key itself is encrypted using the receiver's public key (RSA or **ElGamal** algorithm)
- The encrypted message (using the session key) is concatenated with the encrypted session key (using the receiver's public key) and transmitted



PGP does both of these at the same time



PGP compresses the email message after appending he signature but before encryption, this makes long-term storage of messages more efficient.

Compression Steps

- For messages that exceed a maximum size limit (which can be as low as 50,000 octets on some systems), the PGP process breaks down the message after several steps have already occurred.
- **Compression**: The original message is compressed to reduce its size and remove redundancy. This can strengthen the encryption.
- **Encryption**: PGP encrypts the message using a one-time session key, which is itself encrypted with the recipient's public key.
- **ASCII Conversion**: For email compatibility, the encrypted text is converted into standard ASCII characters using a process called radix-64 conversion.
- **Segmentation**: If the message is still too large, the PGP software splits it into multiple parts. Because this occurs after all encryption and formatting, the digital signature and session key information only appear once, typically in the first segment.

One problem with PGP is that individuals may possess multiple public and private keys, this is addressed by using a relatively short key identifiers.

These keys ids consist of the least significant 64 bits of a public key

Every PGP agent has two lists:

- Private key ring: stores paired private/public keys. Private keys are stored in an encrypted form
- Public Key Ring: Stores public keys

Private Key Ring Table:

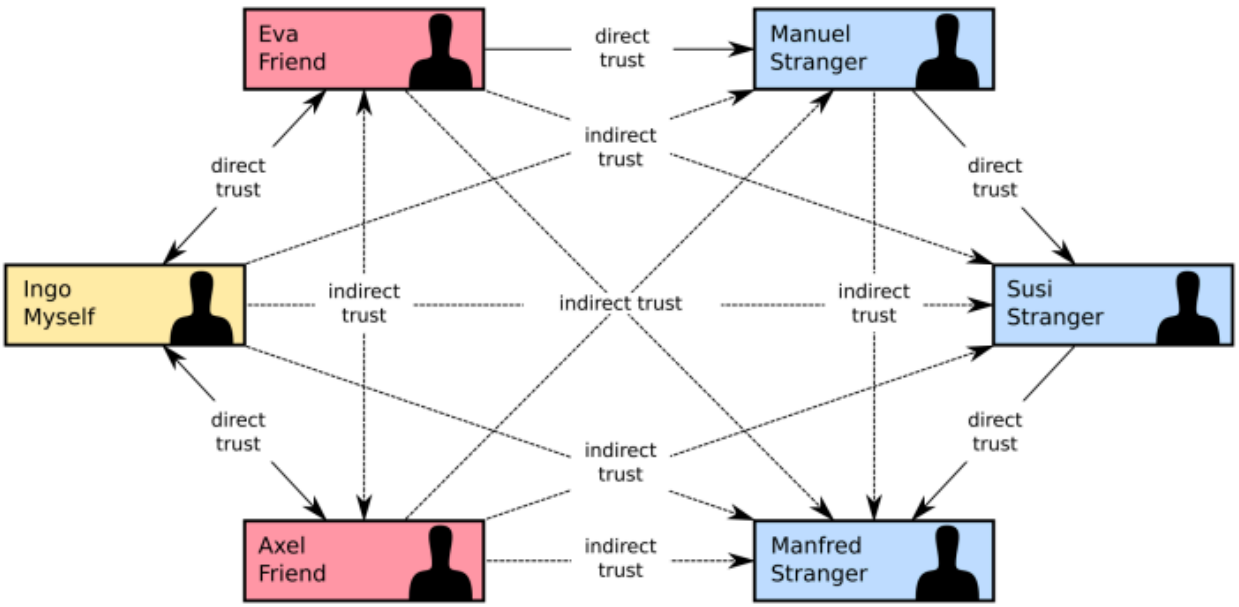
User ID	Key ID	Public Key	Encrypted Private Key	Timestamp
kak@abc.com	EA132....43	EA132....43....A21	34ABF23.....A9	041908−11:30
⋮	⋮	⋮	⋮	⋮

Public Key Ring Table:

User ID	Key ID	Public Key	Producer Trust	Certificate	Certificate Trust	Key Legitimacy	Timestamp
kak@abc.com	EA132....43	EA132....43....A21	Full	——	——	Full	041908−11:30
zaza@foo.com	132AB....02	132AB....02....23A	Full	——	——	Full	——
toto@bar.com	231DA....02	231DE....02....33E	Full	Zaza's	Full	Full	——
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

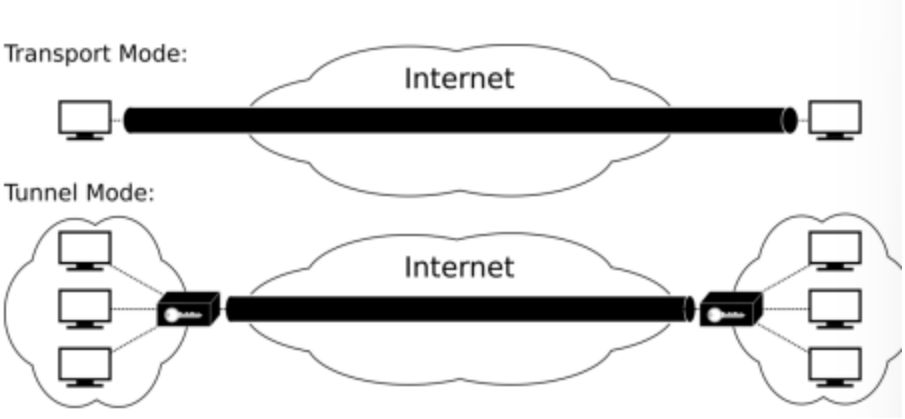
We use a web of trust for authentication for large groups of peopke, meaning not everyone needs to be authenticated with other people.

- A unique, **bottom-up approach** for authenticating the binding between a public key and its owner, contrasting with the hierarchical PKI/CA approaches
- In the web of trust, **a user's public key can be signed by any other user**
- If User A fully trusts User Zaza, and Zaza signs Toto's public key, A may subsequently trust Toto fully
- The **Key Legitimacy** field in the Public Key Ring is automatically derived by PGP from the trust values stored in the **Certificate Trust** field(s)



IP Security

IPsec is a group of protocols for securing connections between devices, this helps keep data sent over public networks secure. Works by providing private tunnels over insecure networks



IPsec works at the network layer, the largest applications of IPsec are VPNs. These features are built into IPv6 and can be used with IPv4

Components

Authentication header provides IP authentication (confirming source and integrity, protecting again man in the middle attacks for example)

- The AH stores a hash value of invariant packet portions. (Protocol number 51 in IPv4)
  - Contains a 32-bit Security Parameter Index (SPI), which establishes the Security Association (SA) for the packet
  - Contains a Sequence Number to prevent replay attacks
  - Contains Authentication Data (the MAC calculated using SHA-1 or HMAC)
- Encapsulating Security Payload (ESP): Provides IP-level confidentiality through encryption
- ESP can also provide authentication. (Protocol number 50 in IPv4)

IPSec Modes of Operation

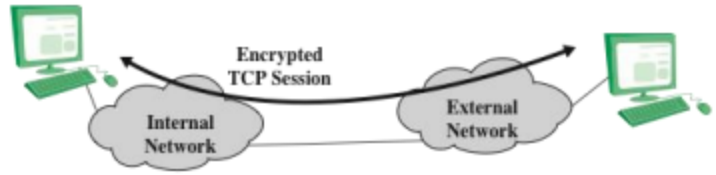
- 1. Transport Mode
  - The regular mode for packets traveling from source to destination
  - The endpoints must carry out their own security checks
  - In IPv4, the AH/ESP header is inserted right after the original IP header
  - When ESP is used, the Encrypted Payload Data field is the encrypted version of the TCP segment (TCP header + data payload)
- 2. Tunnel Mode
  - Used when the source and destination endpoints cannot carry out the security checks
  - Packets are routed to designated locations (P and Q) for security insertion/verification
  - The original IP packet is encapsulated inside a new IP header (IP-in-IP protocol)
  - In Tunnel Mode, the ESP payload contains an encryption of the entire IP packet

Transport Mode

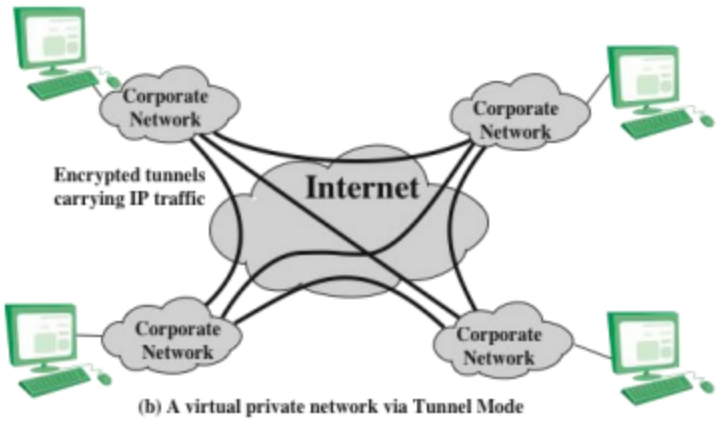
- Provides protection primarily for upper-layer protocols
- Examples include a TCP or UDP segment or an ICMP packet
- Typically used for end-to-end communication between two hosts
- ESP in transport mode encrypts and optionally authenticates the IP payload but not the IP header
- AH in transport mode authenticates the IP payload and selected portions of the IP header

Tunnel Mode

- Provides protection to the entire IP packet
- Used when one or both ends of a security association (SA) are a security gateway
- A number of hosts on networks behind firewalls may engage in secure communications without implementing IPsec
- 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



(a) Transport-level security



(b) A virtual private network via Tunnel Mode

SSL/TLS for Transport Layer

- **SSL (Secure Socket Layer)** was originally developed by Netscape in 95
- Provides **Transport Layer Security** (sitting immediately above TCP in the protocol stack)
- IETF standardized SSL Version 3 as **TLS (Transport Layer Security) Version 1** (RFC 2246)
- The combined acronym SSL/TLS is common due to the popularity of the **OpenSSL** library
- Critical to secure web commerce (e.g., **HTTPS**) and secures various application traffic (email, chat, SSH)
- Security relies fundamentally on **certificates issued by Certificate Authorities (CA)**

- **Server-Only Authentication:** Client verifies the server's certificate, encrypts a client-generated secret key with the server's public key, and sends it to the server

- **Server-Client Authentication:** The client also sends its certificate to the server for authentication

SSL/TLS is composed of **four protocols in two layers**

Protocol	Layer	Role
SSL Handshake Protocol	Upper Layer (above Record)	Authenticates clients and servers to each other
SSL Record Protocol	Lower Layer (above TCP)	Transmits data confidentially
SSL Cipher Change Protocol	Upper Layer (above Record)	Plays a relatively minor role
SSL Alert Protocol	Upper Layer (above Record)	Used to convey SSL-related alerts

- SSL Handshake Protocol - Phases 1 & 2 (20.4.3)

- The Handshake Protocol determines the algorithms, authenticates peers, and generates cryptographic keys

Phase	Action	Key Messages / Details
Phase 1: Establish Capabilities	Client and server agree on security capabilities	**Phase 1: Establish Capabilities**
Phase 2: Server Authentication/Key Exchange	Server validates its identity to the client	**Phase 2: Server Authentication/Key Exchange**

Phase	Action	Key Messages / Details
Phase 3: Client Key Exchange/Authentication	Client sends required keys and potentially its certificate	**Phase 3: Client Key Exchange/Authentication**
Phase 4: Finish Setup	Client and server finalize the secure connection state	**Phase 4: Finish Setup**

- **Purpose**: To keep an SSL/TLS session alive during temporary lulls in data exchange, avoiding the significant overhead of renegotiating security parameters
- The protocol sits on top of the SSL/TLS Record Protocol
- **Key messages** are HeartbeatRequest and HeartbeatResponse
- An endpoint uses a retransmit timer; if a HeartbeatResponse is not received within that interval, the session is terminated
- A request packet includes an arbitrary payload and its length, which the receiver must return without change in its response, serving as protection against a replay attack

- The Heartbleed bug (discovered April 7, 2014) exploited this extension

- The receiver of a HeartbeatRequest did not check that the declared payload_length actually matched the content size

- This flaw allowed an attacker to request a large payload length (up to 2^{16} bytes) while sending minimal content, causing the receiver to allocate memory based on the requested length and return unverified memory contents (potentially containing private keys, passwords, etc.)