Application Layer Protocols

→ HTTP → communication btwn web browsers & web servers
retrieval & delivery of web pages, images & other resources

→ SMTP (Simple Mail Transfer Protocol)
→ send & receive email msgs btwn mail servers

→ FTP → file upload, download & management operations

→ DNS → Resolves domain names to IP addresses

→ SNMP (Simple N/w Mgmt Protocol)
→ mgmt & monitoring of n/w devices

→ POP (Post office Protocol) & IMAP (Internet Msg Access Protocol)
→ retrieving email msgs from mail servers

→ RTP (Real Time Transport Protocol) → real time audio & video

HTTP

→ language for web browsers & web servers communicate

→ set of rules for requesting & delivering web page

→ HTTP/0.9 (1991) → simple → only GET method
→ no headers / other content
→ response in plain text format
→ no status code

→ HTTP/1.0 (1996) → support for multiple types of content like images & sound files
→ headers in req & response
→ status codes
→ limitations in terms of performance & scalability particularly for handling multiple reqs over single connect^{n}

→ HTTP/1-1 (1999) → persistent connects → multiple reqs over single connect

→ reducing overhead of establishing new connect for each req.

→ chunked transfer encoding
→ cache control
→ Range requests
→ content negotiat
→ host header (virtual hosting) - multiple sites on same IP addr

→ HTTP/2 (2015) → Multiplexing

→ multiple reqs & responses to be interleaved over single connect

→ Header compression - efficient binary framing layer reducing size of headers

→ HTTP/3 → underlying protocol - QUIC transport protocol (Quick UDP Internet Connections)

(2020)

(Developed by Google)

→ QUIC built on top of UDP

→ adv over TCP → reduced latency, improved congestion control, better handling of unreliable n/ws

HTTP Methods

→ Get → Retrieves representation of a resource / collection of resources
→ only retrieve data & should not have any side effect on server

→ Post → Submits data to be processed to specific resource
→ commonly used to create new resources on server
→ can be used for submitting form data or uploading files

→ Put → updates an existing resource with provided data
→ reqs are idempotent → multiple put reqs should have same effect as a single req.

→ Delete → Deletes specified resource from server

→ Patch, Head, Options, Trace, Connect

HTTPS

→ secure version of HTTP

→ encrypted communicat → ensures confidentiality & integrity of data

① Client Hello

→ info like supported SSL/TLS versions, cipher suits etc] Hand Shake

② Server Hello

→ selects opt TLS version & cipher suite

→ sends its digital certificate containing public key issued by certificate authority (CA)

③ Certificate Validat"

→ client verifies authenticity by checking its validity, issuer & signature

④ Key Exchange → Client key Exchange
Server key Exchange

⑤ Session key Generate → both generate session keys (encrypt keys)

⑥ Change Cipher Spec → indicate that subsequent msgs will be encrypted - both agree to switch to encrypted comm.

⑦ Encrypted Data Transfer - Symmetric encrypt algos like AES (Advanced Encrypt std)

→ key exchange uses asymmetric encrypt while data transfer uses symmetric encrypt

Symmetric Encrypn - Computationally more efficient for large amount of data. Requires secure method for sharing & managing secret key b/w communicating parties

eg - AES (Advanced Encrypn Std), DES (Data Encrypn Std)

Asymmetric Encrypn → Pair of mathematically related keys → public & private

→ Public key → widely distributed
→ used for encrypn

→ Private key → kept secret for decrypn

→ Computationally infeasible to derive private key from public key

→ Typically slower & more expensive

→ often used for key exchange, digital signs etc.

→ eg - RSA (Rivest Shamir Adleman), ECC (Elliptic Curve Cryptography)