1 Layers

Application (WEB, mail, DNS)
SSL (encrypt, decrypt, authenticate)
Transport (TCP, UDP) - port
Network (IP) - IP address
Link (DSL, WiFi, optical)
Physical (cooper, fiber, wireless)

1.1 Packet

All the layers form a packet (with every headers)
Encapsulation : add a header
Decapsulation : remove a header

1.2 Connection

Switch : queue (stores packets), forwarding table (store meta-data, where to send packets)
Router : network layer switch
Bottleneck link : slowest trafic rate

Average throughput : average rate (bits/sec) at which destination receives data (matters for bulk transfers), (approximatively average window size over average RTT)
Transmission delay : $\frac{\text{packet size}}{\text{link transfer rate}}$
Propagation delay : $\frac{\text{link length}}{\text{link propagation speed}}$
Queuing delay : time during packet sits inside a queue at a switch
Processing delay : time for switch to process the packet (after removed from queue and before transmission), independent of packet size

Packet loss : fraction of packets (%) from source to destination that are lost on the way
Packet delay : transmission delay $+$ propagation delay ($+$ queuing $+$ processing) (matters for small messages)

Packet switching : Packets treated on demand (efficient, no performance guarantees, congestion)
Connection switching : resources reserved in advance (performance guarantees + inefficient resource use)
Statistical multiplexing : users share resources, do not expect to be all active at the same time

Peering : Allows ISPs to exchange traffic directly (might be through an IXP)

Scalability : ability to grow, maintain system properties at a reasonable cost
Hierarchy : universal technique for scaling large systems, nodes that are high up make good attack targets
Caching : universal technique for improving performance, stale data challenge : dynamic fetch(introduce delay) or tolerate inconsistency, trashing the cache is a potential vulnerability

1.3 Attacks

Eavesdropping : sniffing
Impersonation : spoofing
Denial of service, Malware

2 Application Layer

Client : issues requests, Server : answers (/deny)
Interface : where two systems meet and interact
API : interface between application and internet
Network interface : interface between end-system and network, hardware or software that sends and receive packets
DNS name (hostname) : identifies a network interface/end-system
Process name/address : identifies a process (format : IP address + port number)
Client-Server architecture : separation of roles, server runs on dedicated infrastructure
Peer-to-Peer : peer may act as both server and client, personally owned end-system

2.1 WEB

Port 80
URL : identifies a web object (format : DNS name + file name)
Base file + referenced files (URL)
HTTP : connection-less, stateless
HTTP request types : GET = download file, POST = provide info, HEAD = get file metadata, PUT = upload file
HTTP response types : OK, Not found, Bad request, Moved permanently
Cookies : state created by server stored on client, links subsequent HTTP requests
TCP connections :
Persistent : reuse the same TCP connection for many HTTP requests
Parallel : exchange many HTTP requests and responses in parallel
Caching : proxy web server or web cache, reduces delay (improve performance) for web clients, conditional GET for data freshness

2.2 DNS

Uses client/server architecture, stateless, port 53
UDP for short exchanges (client-server), (TCP between DNS servers)
Recursively : DNS server asks another DNS server
Iteratively : DNS server returns the IP address of another DNS server

DNS servers : root (.com, .org, ...) > TLD > authoritative (must know the answer), each node knows how to reach its children
Local DNS server : answer requests from nearby DNS clients
DNS caching : DNS clients and servers cache name-to-I address mappings, reduces load at all levers, reduces delay, relies on TTL for freshness
RR : pice of information, types (A, CNAME, MX, ...)
Query : request for an RR (answer)
Message : set of queries and answers
Attacks : impersonate DNS server to provide incorrect mapping, DOS the servers, trash the cache of server to slow down its responses

2.2.1 RR types

A : address record, hostname to IP address
CNAME : canonical name record, aliases to main name
PTR : pointer record, IP address to canonical name
MX : mail exchange record, domain to SMTP email server hostname
NS : domain to authoritative nameservers hostname
SOA : start-or-authority, domain to administrative informations (address of primary authoritative nameserver, email of administrator, etc)

2.3 BitTorrent

File distribution : time increases
Client-server : linearly with number of clients
P2P : sub-linearly with the number of peers
P2P scales better than client-server

Content : set of data files stored in a peer
Metadata file : special file, stores informations about the data files, from web server or from peer (.torrent file)
Steps to retrieve content : learn metadata file ID (magnet link), find metadata file location, get metadata file, find data file locations, get data files from peers
Tracker (end-system) or DHT (distributed) : knows the location (IP address of peers) of the files
DHT : file ID space partitioned (each peer : owns an ID range, knows the ranges owned by its neighbors, forwards request to neighbor whose range is closest to the targe file ID)

3 Transport layer

Header : source port, destination port, ...
Reliable data delivery : deliver message to the destination or signal failure, detect/recover packet loss (web, file transfer)
Guaranteed performance : minimum throughput (video calls), maximum E2E packet delay (voice, gaming)
Guaranteed security : (implemented by applications)
Confidentiality : reveal only to destination
Authenticity : come from claimed source
Data integrity : no change along the way
Segment : Transport layer headers + application layer message
Datagram : Network layer header + segment
MMS : dictated by network properties

TCP : reliable in-order data delivery, flow control, congestion control, connection-oriented, stateful (maintains state on local/remote process pairs), connection and ACKs costs
UDP : detection of packet corruption, connection-less, stateless, (does not really offer reliable data delivery, checksums in header)
No protocol offers guaranteed performance

UDP sockets : unique (IP address, port nb), may use same UDP socket for many remote processes
TCP sockets : listening & connection sockets, each connection socket has unique (local IP, local port, remote IP, remote port), use different TCP connection socket per remote process
Acknowledgment (ACK) : feedback from receiver to sender (for each segment), used by sender to detect & overcome data corruption (retransmit : recover corruption & loss)
Sequence number (SEQ) : identifier for data (added by sender to each segment), used by receiver to disambiguate data
Timeout : no arrival of expected acknowledgment, segment lost or delayed, used by sender to overcome data loss
TCP Retransmission triggers : Timeout : retransmit oldest un-ACKed segment, 3 duplicate ACKs : fast retransmit oldest un-ACKed segment - don't wait for timeout

Multiplexing : upon new message reception, create new packets, identify correct IPs & ports
Demultiplexing : upon new packet, identify correct destination process, many processes running in application layer
Checksum : redundant information, detect data (segment) corruption, sender adds checksum to each segment

3.1 Sender utilization

Stop and wait : poor sender utilization (sender waits for feedback)
Pipelining : better utilization (sender sends up to N un-ACKed segments), $N :=$ sliding window size
Go-back-N : receiver accepts no out-of-order segments, ACKs are cumulative (ACK for segment #10 tells all 10 have been received), sender retransmits all the un-ACK-ed segments (multiple retransmissions)
Selective repeat (SR) : receiver accepts $N - 1$ out-of-order segments, ACKs are selective (ACK for segment #10 is only for this one), sender retransmits only one segment

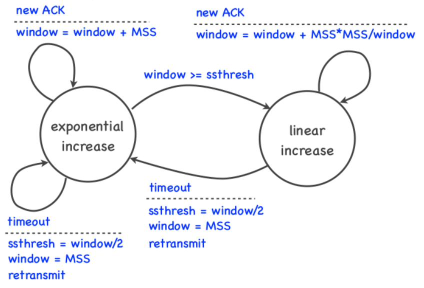
3.2 TCP

Bytes are implicitly numbered, SEQ is # of fist data byte, ACK is # of next expected data byte (cumulative), always send ACK and SEQ, retransmits 1 segment

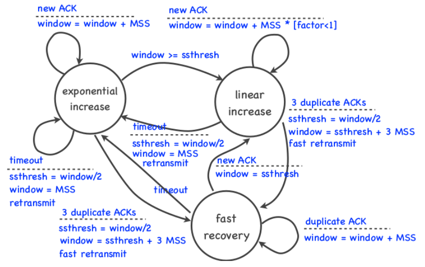
Connection setup : 3-way handshake, "TCP client" end-system initiating the handshake, "TCP server" the other end-system, first 2 segments carry SYN flag (1 bit in TCP header), "TCP connection (established)" = resources allocated for communication
Flow control : goal is not overwhelm the receiver, using "receiver window", receiver communicates spare room in rx buffer in TCP header field
Sender window : minimum of receiver window and congestion window

3.3 Congestion control

Goal is not overwhelm the network, using congestion window (nb of un-ACK bytes sender can transmit without congestion), sender estimates on its own
Bandwidth-delay product : max amount of traffic that sender can transmit until receiving first ACK, = max congestion window size, rate * RTT
Self-clocking : sender guesses the "right" congestion window based on the ACKs, ACK means no congestion (increase window), no ACK means congestion (decrease window)
Increase window size :
Exponentially (slow start) : by 1 MSS for every ACKed segment, so window doubles every RTT, when congestion not expected
Linearly (congestion avoidance) : by $\frac{\text{MSS}^2}{2}$ for every ACKed segment, so by 1 MSS every RTT, when congestion expected
Basic algorithm (Tahoe) : Set window to 1 MSS, increase exponentially
On timeout reset window to 1 MSS, set ssthresh to last window / 2, when reaching sstresh, transition to linear increase



Basic algorithm (Reno) : Same as Tahoe + 3 duplicate ACKs, set window to sstresh (+inflation), retransmit, when reaching sstresh



3.3.1 Attacks

Connection hijacking : attacker impersonates TCP server (or client), fake valid segment if the TCP header is predictable, solution is make TCP header (SEQs) unpredictable

SYN flooding : attacker exhausts incomplete connections buffer (spam SYN), one small resource affect all TCP communication, solution is pass state to TCP client (hash of secret in SEQ)

4 Network Layer

4.1 Functions

Forwarding : local process on a router determining output link for each packet, read network-layer header & search forwarding table for output link

Routing : network-wide process populating forwarding tables (map IP prefixes to output links), routing algorithm on centralized network controller or routers themselves, (longest prefix matching)

4.2 Types

Virtual-circuit/Datagram

Possible guarantees : in-order delivery, maximum delay, minimum throughput, security (confidentiality, authenticity)

Virtual-circuit network : uses connection switching = network-layer connections, appropriate for performance guarantees, forwarding state is per connection (input link, VC #, output link, VC, #), populated by connection setup (routing), attack by overflowing VC table

Connection : set of entities maintain state allowing synchronization

IP/Datagram network : uses packet switching = no network-layer connections, appropriate for best-effort (without any promises), forwarding state is per destination prefix (destination prefix, output link), populated by routing, smaller forwarding tables (no per-connection state in routers), simpler routers (no connection setup and teardown in routers), match most specific entry (longest prefix match)

Location-dependent addresses : address embeds location (address proximity implies location proximity), reduces forwarding state (per destination prefix, not per destination)

IP prefix format : IP prefix = range of IP adresses, mask = number of MSBs to take into account

IP subnet : informal contiguous network area not containing any routers, end-systems and incident routers have same IP address prefix

IP address assignment : number from 0 to $2^{32} - 1$, each organization obtains IP prefixes (from ISP, ...), network operator assigns IP addresses to (router interfaces manually, end-systems DHCP/manually)

NAT : resolves IP address depletion problem, private address spaces, requires per-connection state at border routers of private IP subnets, (router/NAT gateway uses per-connection state (IP, origin TCP port, new TCP port) to rewrites & reroute packets it receives to correct private ip), small number of end-systems

4.3 IP Routing

Least-cost path : find least-cost path from eah source router to each destination router

Link-state routing algorithm for source u : input (router graph & link costs), output (least-cost path from source router u to every other router), centralized (runs either on router u or on separate computer "network controller" for all routers), Dijkstra, each entity has complete view of network, converges faster

Distance-vector routing algorithm : "distributed" algorithm (all routers run it "together", neighbors exchange and react), Bellman-Ford (problems : routing loop, long to resolve "count-to-infinity", poisoned reverse = set cost to infinity), each entity obtains incrementally new information about network, uses less bandwidth

Internet routing challenges : scale (link-state would cause flooding, distance-vector would not converge), administrative autonomy (IPS may not want least-cost routing, may want to hide link costs)

Internet routing : each router learns (1 or a few) 1 route to each (foreign AS) IP subnet in local AS, not to each IP subnet of each foreign AS

Intra-AS routing : run by all routers in same AS, goal = propagate routes within local AS (each router advertizes routes to its local IP subnets, potentially routes to other ASes that it learned through BGP, border router talk to external neighbors eBGP, and other border routers of local AS iBGP), OSPF, RIP

Hierarchy : scale = state not per IP subnet (router forwarding entries per local IP subnets and foreign IP prefixes, may communicate with all other local routers and external neighbors), administrative autonomy (AS chooses its own intra-AS routing protocol)

5 Link Layer

5.1 IP subnet

Takes packet from one end of one physical link to other end

Error detection : receiver detects and drops corrupted packets, relies on checksums

Reliable data delivery : sender/receiver detect corruption and loss, try to recover, relies on checksums & ACKs & retransmissions, only for error-prone links (wireless)

Medium Access control (MAC) : Sender manages access to shared medium (wireless), listens for ongoing transmissions or collisions, backs off and retries later

5.2 Internet

(Network of networks)

Takes packet from one end of one IP subnet to other end

MAC address : 48-bit number (format 1A-2B-DD-78-CF-CC), value of each byte as hexadecimal, flat (not hierarchical like IP address, not location dependent)

L2 forwarding : local switch process determines output link of each packet, relies on forwarding table (maps destination MAC addresses to output links)

L2 vs IP forwarding : L2 similar to IP (L3) forwarding, L2 (flat addresses, no way to group MAC addresses with prefixes, forwarding table size is number of active destination MAC addresses in IP subnet), IP (or L3) (hierarchical addresses grouped in IP prefixes, forwarding table size is number of IP prefixes in the world)

L2 learning : switch learns from traffic, switch adds MAC x to link y mapping when packet arrives at link y, switch broadcasts when it does not know destination MAC

L2 learning vs IP routing : serves similar role, L2 learning (relies on actual traffic, switches do not exchange, explicit routing information), IP routing (relies on routing protocol, routers exchange explicit routing messages)

Spanning tree : subgraph including all nodes + some edges, cannot remove an edge without disconnecting a node, useful for loop-free broadcasting (broadcast traffic propagated only along tree, prevents forwarding loops)

ARP : goal is to map IP address to MAC address, broadcast request (reaches every end-system and router in local subnet) & targeted response

ARP vs DNS : similar role, ARP (relies on broadcasting, no logically centralized map, each entity knows its own MAC), DNS (logically centralized map, stored in DNS servers)

Basic Ethernet : ARP, L2 forwarding and learning

Three levels of hierarchy : IP subnet (L2 forwarding, learning) < AS (IP L3 forwarding, intra-domain routing) < Internet (IP L3 forwarding, inter-domain routing BGP)

Switch and router addresses : both switches and routers have both MAC and IP addresses (one IP per network interface), to be reachable by administrator, for link testing, router needs IP address to respond to ARP requests, router acting as NAT gateway needs IP address for NAT

6 Security

Confidentiality : Only sender and receiver understand content of message ; symmetric = A encrypt message with K - B decrypt with K, asymmetric = A encrypts message with B's public key - B decrypt with his private key

Authenticity : Message is from whom it claims to be ; A appends MAC, A appends digital signature (using her private key) - B checks (using A's public key), use nonce to prevent replay attacks - A append hash of nonce + key + message (like MAC with nonce) (shared key)

Integrity : Message was not changed along the way, same as authenticity

Ciphertext : should (ideally) reaveal no information about plaintext

Encryption : plaintext in, ciphertext out

Decryption : ciphertext in, plaintext out

Symmetric key crypto : parties share same key, key used both for encryption and decryption (RC4, AES, blowfish), fast, challenge is how to share key (out-of-band secret sharing), $k(k(t)) = t$

Asymmetric key cryptography : parties use different keys (public $k+$, private $k-$), $k-(k+(t)) = t$, $k+(k-(t)) = t$, (RSA, DSA), public key not secret, private key is secret, computationally expensive

Cryptographic hash function : map large input space to small hash space, hash (ideally) reveals no information on input, hard to identify two inputs having same hash

MAC : hash(key, text), proof this plaintext was sent by entity knowing the key (shared key)

Digital signature : generate $k-$(hash(text)), verify $k+$(...) == hash(text), proof text sent by entity knowing private key matching public key

Nonce : number A is supposed to use once to generate message for B within given deadline, A include nonce in hash, prevents replay attacks (if P replays A's message to B, nonce will be same so message rejected)

Man in the middle : can break confidentiality, M convinces A to use M's public key instead of B's - M decrypts and re-encrypts for B, cause is no way to verify public keys

Public-key certificates : rely on trusted CA (entity that A and B trust), CA produces certificate of B's public key (digitally signed : CA-$k-$(hash(B owns B-$k+$))), A needs CA's true public key to check certificate signature

Bootstrapping is unavoidable : secure communication requires some form of shared state, asymmetric crypto reduces bootstrapping information

Securing TCP applications : server sends its public key and certificate, client creates and sends a symmetric master key (encrypted with server's public key), both use master key to create 4 session keys (1 for encrypting client to server, 1 for creating mac for client to server, same for server to client), client organizes data in records (with sequence number)
- creates MAC for each record + sequence number (using a session key), encrypts data + MAC of record (using another session key), sequence numbers avoids reordering attacks

PGP : symm-k(A-$k-(h(msg)){+}{+}msg$) $++$B-$k+$(symm-k)

Operational security : network operator allow minimum amount of traffic, use filtering table telling border router which packets to drop and which ones to forward, (allow/deny - src IP - dst IP - protocol - scr port - dst port)

7 Acronyms

ISP	Internet Service Provider
IXP	Internet eXchange Provider
DSL	Digital Subscriber Line
CMTS	Cable Modem Termination System
IP	Internet Protocol
MAC	Media Access Control
DNS	Domain Name System
DDOS	Distribute Denial Of Service
E2E	End-To-End
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
SSL	Secure Sockets Layer
URL	Uniform Resource Locator
HTTP	HyperText Transfer Protocol
API	Application Programming Interface
TLD	Top-Level Domain
TTL	Time To Live
RR	Resource Record
P2P	Peer-To-Peer
DHT	Distributed Hash Table
ID	Identity Document
VDI	Virtual Desktop infrastructure
RT(T/D)	Round-Trip Time/Delay
SMTP	Simple Mail Transfer Protocol
MMS	Maximum Segment Size
SR	Selective Repeat
VC	Virtual-Circuit
DHCP	Dynamic Host Configuration Protocol
NAT	Network Address Translation
AS	Autonomous System
BGP	Border Gateway Protocol
OSPF	Open Shortest Path First
RIP	Routing Information Protocol
AES	Advanced Encryption Standard
MAC	Message Authentication Key
CA	Certificate Authority
PGP	Pretty Good Privacy
MAC	Medium Access Control
ARP	Address Resolution Protocol
DSA	Digital Signature Algorithm
LAN	Local Area Network