efficiency

Slotted Aloha

Maximum efficiency of slotted ALOHA is achieved by finding p* : $p* = argmax_p(N * p * (1-p)^{N-1})$ For $N \to \infty$: max efficiency: $1/e \approx 0.37$

Pure ALOHA

Successful transmission prob: $p \cdot (1-p)^{2(N-1)}$ Max efficiency: $1/2e \approx 0.18$

CSMA/CD

$$\text{Efficiency} = \frac{1}{1 + 5 \cdot \frac{d_{prop}}{d_{trans}}}$$

CSMA/CD vs CSMA/CA

Carrier Sense Multiple Access (Collision detection / Collision Avoidance)

Collision detected

CSMA/CD

Stop transmitting and wait random amount of time before transmitting again

CSMA/CA

Start random backoff counter, while channel is busy, freeze it. When zero: frame is transmitted and wait for ACK. No ACK: max backoff increased and new timer is started

Binary Exponential Backoff

N subsequent collisions: backoff time chosen from $\{0, 1, 2, ..., 2^n - 1\}$

Taking Turns protocols

Polling protocol

Master node polls each node in round-robin fashion

Advantages

Eliminates collisions and empty slots that occur in random access protocols

Disadvantages

Polling causes delays, master node: single point of failure

Token-passing protocol

Only node holding the token can transmit, passing token after max number of frames (or done)

Advantages

eliminates collisions and empty slots

Disadvantages

Complex failure cases: single node failure: crash, lost token

"Package names"

- Application: message
- Transport: Segment
- Network: Datagram
- Link: frame

Application Layer

Communication between processes

- Client: Process that initiates connection between pair of processes
- Server: Process that waits to be contacted to begin the session
- Socket: The API between application and transport layer

Persistent vs non-persistent connections

Non persistent (HTTP/1)

Start and close connection after each object

Persistent (HTTP/1.1)

open connection, send all objects, close connection

HTTP/1.1: pipelining

HTTP/1.1 supports pipelining to further reduce RTT

Caching

Why? reduces response time, avoid bottleneck links, reduce traffic on connection to internet

HTTP/2

Reduces latency throug multiplexing, request prioritization, server push and compression. Aims to solve HOL blocking problem

HTTP/2 HOL blocking solution: Framing

Each message (object) is broken into small frames request and response frames are interleaved over single TCP connection

DNS

Resource records (types)

- A: Provides standard hostname to IP mapping
- NS: route queries along the chain (to new DNS server)
- CNAME: alias hostnames to canonical hostnames
- MX: alias mail server names to canonical mail server names

Transport Layer

Connectionless (de)multiplexing using UDP

Client generaly uses random port number, server binds to known application port number, UDP socket is identified by two tuple (destination IP, destination Port)

Connection-oriented (de)multiplexing in TCP

TCP sockets are identified by 4-tuple: Source port, dest. port, source IP, dest. IP ; Many to one mapping from sockets to processes

UDP

UDP is barebones

Why UDP instead of TCP?

Finer application-level control over what data is sent, and when ; No connection establishment (less delay) ; No connection state (less overhead) ; Smaller packet overhead

UDP checksum

1s complement of sum of all 16-bit words in UDP segment (+ pseudo header)

Limitations of UDP checksum

Optional: all 0's means it hasnt been calculated ; All 1 bit errrors detected ; not all 2 bit errors are caught ;

Automatic Repeat reQuest (ARQ) protocols

based on 3 fundamental capacities: Error detection ; Receiver feedback ; retransmission

Standard ARQ has fundamental flaw

If ACK is corrupt, resent packet : introduces duplicate packets ; use timeout to detect lost packet: very hard to calculate timeout

Stop and wait: bad performance (not pipelined)

Go-Back-N

Window size: N
Sender vars: base, nextseqnum
We use cumulative ACKS
Timer for oldest unacknowledged packet: on timeout: retransmit all unACKd packets Receiver vars: expectednextseqnum
Out of order packet: discard and sent ACK with highest in-order seq num

Selective Repeat SR

Windows size: N again
Sender vars: base, nextseqnum
recvr vars: rcv_base: sequence number of oldest unACKd packet
Sender: Data is only transmitted if the next available seq num valls in senders window
timer for each packet: restransmit when expired Receiver: packet in window: buffer it and send selective ACK

How many distinct seq nums?

Window size N must be less than or equal to half the sequence number space

TCP

seq num: position of its first byte in the stream
ACK number: next byte that receiver expects

Estimating RTT

Vars: SampleRTT, EstimatedRTT (exponential weighted moving average), DevRTT
$EstimatedRTT = (1-\alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$
alpha usually 0.125
$DevRTT =$
$(1-\beta) \cdot DevRTT + \beta * |SampleRTT - EstimatedRTT|$ beta usually 0.25

Managing timeout

Timeout: Interval = 2 * interval
EstimatedRTT updated:
$interval = EstimatedRTT + 4 \cdot DevRTT$

Fast retransmission

Lost packets can also be detected by counting duplicate ACKS After 3 duplicate ACKS, oldest non-ACKd segment is retransmitted

Flow Control

Ensures that sender does not send faster than receiver can receive
$rwnd = RcvBuffer - [LastByteRCVD - LastByteRead]$
Sender: $LastByteSent - LastbyteAcked \leq rwnd$

Congestion Control

2 types: End to end and Network-assisted

TCP Congestion Control

$LastBytesent - LastByteAcked \leq min(cwind, rwnd)$
How is perception perceived? Timeout or duplicate ACKs
State 1: slow start
cwnd is initialized to 1 MSS and increased by 1 when ACK. When timeout occurs: ssthresh is set to cwnd/2 and cwnd is reset to 1 MSS When cwnd ¿= ssthresh TCP transitions into Congestion AVoidance mode Optional: when 3 duplicate ACKS: TCP goes into fast recovery
State 2: Congestion Avoidance:
TCP only increases cwnd by 1 MSS every RTT (instead of doubling): when ACK arrives: $cwnd = cwnd + MSS^2/cwnd$
TCP goes back into slow start when timeout occurs or fast recovery after 3 duplicate ACKs
State 3: Fast recovery:
ssthresh is set to cwnd/2 and cwnd = ssthresh + 3 * MSS
When timeout occurs: TCP oges to slow start ; When an ACK arrives for missing segment, go back to collision avoidance (resetting cwnd to ssthres)

Network Layer

Forwarding vs routing

Forwarding

Move packet from routers input link to appropiate output link ; Local decision at very short time ; typically implemented in hardware

Routing

Determine route or path taken by packets from sender to receiver ; Network-wide process that takes longer ; typically implemented in software

Destination based forwarding

Entries contain ranges of addresses; longest prefix matching is used

Queuing in routers

When input or output queues (buffers) become full, packet loss occurs ; HOL-blocking occurs when a packet in an input queue must wait because its blocked by packet at front of line ; Output port contention occurs when multiple packets are destined for the same output line at the same time

Packet scheduling techniques in router queues

µ FIFO ; Priority queuing ; Round robin and weighted fair queueing (WFQ)

Internet Protocol IP

DHCP

A DHCP server offers several functions to hosts within an organization: automatically assign IP addresses, provide additional information(subnet, local dns, standard gateway, ...) ; Consists of 4 steps: Discovery, Service offer(s), request, acknowledgement ; An assigned IP has a lifetime, which can be extended with a renewal request

NAT

Not enough IP addresses to give every host in every network a unique IP address. ; NAT offers soltion by associating a set of private addresses with single public one ; Each internal address:port is associated with a unique port

Routing algorithms

The centralized link-state routing algorithm

Assumes entire network topology and costs are known
Dijkstra:
Initialization: N' = S ; $\forall v \in N : D(v) = c(s, v)$ c(s,v) = ∞ if they arent direct neighbours
Repeat: $w = argminD(v)$; $N' = N' \cup \{w\}$;
$\forall neighvofw \wedge v \notin N' : D(v) = min(D(v), D(w) + c(w, v))$
Until: N' = N
Distance Vector: Only relies on information obtained from neighbours
Every node x keeps track of: Estimate distance vector $D_x = [D_x(y) : \forall y \in N]$; Cost c(x,v) to each of its direct neighbours ; Distance vector of each of its neighbours
Initialization: $\forall y \in N : D_x(y) = c(x, y)$; $\forall$ neigh w: send D_x
Repeat: (whenever link cost changes or DV received from neighbour) $\forall y \in$: $D_x(y) = min(c(x, v) + D_v(y))$ for all v ; If D_x changed for any destination: send D_x to all neighbours.

Routing on the internet

Internet is split up into Autonomous systems. Two different types: intra-AS (Interior Gateway Protocol IGP such as OSPF), inter-AS (EGP such as BGP) ; Advantages of approach: scalability and Administrative Autonomy
OSPF: Based on link-state and Dijkstra's shortes path ; Routers broadcast link-state information to all other routers in AS when a link changes and periodically ; Link weights are manually configured by admin
BGP: Routers maintain semi-permanent TCP connection: iBGP (within AS, done as a mesh all routers to all routers), eBGP (two routers in different AS (only direct neighbours)) ; BGP advertisements: AS-PATH: list of ASs through which the message has passed , NEXT-HOP: IP address of router that begins AS-path ; BGP route selection algorithm: If any routes have locale preference, choose highest ; From remaining choose with shortest AS-PATH ; from remaining ones : closest NEXT-HOP ; remaining ones: selected based on BGP identifier.
IP-ANYCAST: BGP can be used to advertise same IP for multiple servers, allowing routers to select nearest one.

Link Layer

Cyclic Reduncancy Check (CRC) codes

Based on polyomial arithmetic: Each bit string can be represented as a polynomial with binary coëfificients ; All calculations done in mod 2 arithmetic without carrier or borrows ; Addition and substraction are thus identical and equal to bitwise XOR ; Multiplication by 2^k can be calculated by left shifting a bit pattern k places.
approach: Sender and receiver agree on r + 1 bit pattern, known as generator G ; Sender appends r bits R to data D so that the rsulting d + r bit patterin is divisible by G ; Receiver divides the bits, if remainder is not zero, error occured.

How does sender compute R ?

R = remainder($\frac{D \cdot 2^r}{G}$)

error detection probability of crc generators

r = num of bits
Can detect up to r - 1 consecutive bit erros ; Can also detect consecutive bit errors with length greater than r + 1 with probability $1 - 0.5^r$; Can detect any number of odd bit errors

Code Division Multiple Access (CDMA)

Each sender is assigned a unique M-bit code ; The code changes at a much faster rate than sequence of data bits ; Each transmitted bit is encoded by multiplying it by the code ; Codes must be carefully chosen (orthogonal)

Receiver

Single sender: d_i: original bits: $d_i = \frac{\sum_{m=1}^{M} Z_{i,m} \cdot c_m}{M}$

Receiver

interfering signals are assumed to be additive. The value received during the mth mini-slot of the ith bit slot:
$Z^*_{i,m} = \sum_{s=1}^{N} Z_{i,m}s$

Link-layer switches: forwarding and filtering

Filtering: determine wheteer a frame should be forwarded or dropped
Forwarding: Determine interface to which frame should be directed (based on switch table)