2017-1 Networks

37046 characters in 6018 words on 1079 lines

Florian Moser

February 20, 2018

1 Introduction & Fundamentals

1.1 key problems

reliability despite failures
Codes for error detection / correction, routing around failures

network growth & evolution
adressing & naming, protocal layering

allocation of resources
multiple access, congestion control

security against various threats
confidentiality of messages, authentication of communicating parties

1.2 reinvention

emergence of the web
Content Delivery Networks

digital songs / videos
peer-to-peer sharing

falling cost / bit
voice-over-IP calling

many hosts
IPv6

wireless advances
mobile devices

1.3 uses of networks

user communication
VoIP, video conferencing, messaging

resource sharing
many users access same resource

content delivery
uses replicas in the network to save transfer cost

computer communication
computer interact with each other

connect computer to the physical world
sensor data, IoT

1.4 value of network

Metcalfe's Law
value of network is proportional to count of nodes n: n^2 → because of more connectivity in large networks

1.5 component names

application, app, user
uses the network (skype, amazon, google)

host, end-system, edge device, node, source, sink
supportes apps (laptop, mobile, desktop)

router, switch, node, hub, intermediate system
relay messages between links (Access Point, cable / DSL model

link, channel
connects nodes (wires, wireless)

1.6 statistical multiplexing

sharing of network bandwidth between users according to the statistics of their demand
multiplexing means sharing
useful because demand is in bursts, and users are mostly idle

1.7 ethernet links

simplex
Daten können nur in eine Richtung fliessen

half-duplex
Daten können in beide Richtungen fliessen

full-duplex
Daten können in beide Richtungen gleichzeitig fliessen

1.8 wireless links

broadcast, message is received by all nodes in range

1.9 example networks

wifi, enterprise / ethernet, ISP, cable / dsl, mobile phone / cellular, bluetooth, satellite

1.10 network name by scale

PAN
Personal Area Network, vicinity (bluetooth)

LAN
Local Area Network, building (WLAN)

MAN
Metropolitan Area Network, city (cable, DSL)

WAN
Wide Area Network, country (ISP)

The Internet
network of all networks, planet (Internet with capital "I")

1.11 network fundamentals

between apps, network components
sockets hide complexity of network

traceroute
probes successive hops to find network path

1.12 network needs modularity because

make & break connections
find a path through the network
transfer information reliably
transfer information of arbitrary length
send as fast as the network allows
share bandwidth among users
secure information on transit
let many new host to be added

1.13 protocols & layers

protocol
like TCP

layer
like Layer 1

peer
the other party on the same layer

each protocol instance talks to its peer only

each protocol instance uses only the services of the next lower layer

uses encapsulation
lower layers outmost; like an onion

uses demultiplexing
passes message to each protocal it uses, done with demultiplexing keys in header

advantages of layering
information hiding & reuse

disadvantages of layering
adds overhead, application might care about lower layers but has no access

1.14 OSI seven layer model

principled standard to connect systems, but are guidelines, not strict! So multiple protocols in same layer possible, or same over multiple.

Layer 7
Application (provides function needed by users: FTP, DNS, SMTP, HTTP)

Layer 6
Presentation (converts different representations: Data Conversion, Compression, Encryption, MIME)

Layer 5
Session (manages tasks dialogs: Authentication, Authorization, SPDY)

Layer 4
Transport (provides end to end delivery: TCP, UDP)

Layer 3
Network (sends packets over multiple links: IP)

Layer 2
Data Link (sends frames of information: ARP, 802.11 (WLAN), Ethernet)

Layer 1
Physical (sends bits as signal: cable, wireless)

1.15 internet reference model

application
programs that use network services (HTTP, FTP, DNS)

transport
provides end-to-end data delivery (TCP, UDP)

internet
sends packets over multiple networks (IP)

link
sends frames over a link (Ethernet, 3G, 802.11, DSL, Cable)

1.16 standard bodies

Telecom
ITU (H.264, MPEG4)

Communications
IEEE (802.11, Ethernet)

Internet
IETF (HTTP/1.1, DNS, domain name registration)

Web
W3C (HTML5, CSS)

1.17 layer based names

(7) application
message (CDN, HTTP, DNS)

(4) transport
segment (TCP, UDP)

(3) network
packet (IP, NAT, BGP)

(2) link
frame (Ethernet, 802.11)

(1) physical
bit (wires, fiber, wireless)

1.18 devices

repeater
physical layer only

switch
link layer

router
network + link layer

middlebox / proxy
transport + network + link layer

1.19 a small history of internet

1969
ARPNET, 4 hosts, email

1974
TCP / IP in ARPNET

1983
Berkley Sockets (links network stack)

1985
NSFNET connects educational networks

1993
BGP, Webpages

1995
ISP's

1998
CDN

now
regional ISP's connect to transit ISP's connect to other ISP's and Content Providers at IXP's

2 Physical Layer

2.1 Socket API

streams
reliably send a stream of bytes

datagrams
unreliably send seperate messages

lets application attach to to the local network at different ports

SOCKET
create a new communication endpoint

BIND
associate a local address with a socket

LISTEN
announce willingness to accept connections; give queue size

ACCEPT
passively wait for an incoming connection

CONNECT
actively attempt to establish a new connection

SEND
send some data over the connection

RECEIVE
receive some data from the connection

CLOSE
release the connection

normal procedure (client)
socket → connect → send → recv → close

normal procedure (server)
socket → bind → listen → accept → recv → send → close

for UDP omit listen, accept, connect and replace recv with recvfrom (has additional argument) and send with sendto (has additional argument)

2.2 scope of physical layer

concerns how signals are used to transfer message bits over a link
wires carry analog signals, but we want to send digital bits

2.3 simple link model

abstraction of physical layer

2.3.1 rate (or bandwith, speed, capacity)

bits/second

2.3.2 delay (or latency)

in seconds, related to length l = t + p

transmission delay
time to put message on the wire t = length of message / rate

propagation delay
time for bits to propagate through wire p = length / speed of signals

queueing delays
time spent waiting in buffers / queues of router / switches

processing delays
time spent being processed

2.3.3 important properties

if channel is broadcast & its error rate

2.4 metric units

important numbers
nano 10^-9, micro 10^6, mili 10^3, kilo 10^3, mega 10^6, giga 10^9

powers of 2 for data size (1KB = 2^10 bytes)

power of 10 for rates (1Mbps = 1'000'000 bps)

B is for bytes

b is for bits

example 1 values
d=50ms, r=10Mbps, m=1250bytes

example 1
L = 50ms + (1250∗8 / 10∗10^6)s

2.5 bandwidth delay product

the amount of data in the wire
BD = R ∗ D

example 1
BD = (10∗10^6) ∗ (50 ∗ 10^-3)

how long is a bit? fiber optic speed=2/3c, sending rate is 1Gbps → 0.2m

2.6 media

propagate signals that carry bits of information

wires
use din LAN & telefone lines, twists reduce radiated signal and reduce effect of external interference signal, shielding for isolation

fiber
long, thin, pure strands of glass, LED → optical fiber (reflecting the signal) → photodetector, mutimode (chaper) vs singlemode (expensive, but up to 100km)

wireless
sender radiates signal, many receivers, interference

2.7 signals

as signals propagate throught the wire at 2/3c, attenuated (geschwächt), attentuated a lot above a cutoff, noise is added

2.7.1 wire

signal over time can be represented by its frequency components (fewer frequencies degrades singal)

2.7.2 fiber

uses three frequency bands at the same time

2.7.3 wireless

speed of light, spread out, attenuate faster than 1/d^2,
interference with other signals of same frequency → spatial reuse

multipath
bouncing off objects messes up signals

2.8 modulation (baseband)

signals representing bits on a wire

2.8.1 NRZ

Non-Return To Zero → 1 is high, 0 is low

2.8.2 NRZI

Non-Return To Zero Inverted → invert signal if one occurrs

2.8.3 clock recovery

receiver needs frequent signal transitions to know the clock

manchester encoding
XOR with clock (clock up & down again represents one bit, starts at 0)
1 is high→low, 0 is low→high

4B/5B
map 4 bits into 5 bits to avoid long run of zeros (has at most 3 zeros in a row)

2.9 signals for fiber & wireless, modulates carrier

2.9.1 carrier

signal oscillating at desired frequency, modulate by changing

amplitude
some amplitude vs none

frequency
fast vs slow

phase
M for starting 1, W for starting 0 (→ place of courve at specific time represents value, M & W are meant graphically)

2.10 fundamental limits

key channel properties
bandwidth b, signal strength s, noise strength n, v signal levels

b limits rate of transitions

s & n limits amount of levels we can distinguish (v)

nyquist limit
maxmimum symbol rate is 2B, ignoring noise → R = 2B∗log2(v)bits/sec

shannon capacity
levels we can distinguish depends on Signal-To-Noise Ratio (SNR): SNR = 10log10(S/N) (in decibel) → C = B∗log2(1 + S/N) bits/sec

for wires & fiber
engineer SNR for data rate (B)

for wireless
adapt data rate to SNR (can't engineer for worst case)

2.11 DSL example

uses passband modulation (sends over multiple frequencies at the same time)
seperates up from downstream & telefone
modulations varies amplitude & phase

SNR
1 - 15 bits / symbol

3 LINK LAYER

3.1 scope of link layer

concerns how to transfer messages over one or more connected links
messages are frames of limited size
ontop of the physical layer
implemented by driver / hardware

3.2 framing methods

byte count
start frame with length field → difficult to resync after frameing error

byte stuffing
special flag value for start / end frame, escape flag value & escape code inside frame (called stuffing)

bit stuffing
flag are six consecutive 1s, after 5 in data insert 0 (remove this on receive)

3.3 PPP

point-to-point protocol
used for link framing in SONET

flag is 0x7E, escape is 0x7D

scan for both caracters, replace if found with escape caracter und real caracter afterwards XOR with 0x20

3.4 noise may flip received bits

add redundancy
error detection / correction with check bits

error code
send D+R bits, if receiver can't calculate R with D → frame has a faillure

hamming distance
number of bit flips needed to change D+R1 to D+R2
for distance d+1, d error will be detected
for distance 2d+1, up to d errors will be corrected

3.5 error detection

detect error & retransmit frame

3.5.1 parity

add one check bit for D data bits d = 1

3.5.2 internet checksum

used in web TCP, UDP

sender
arrange data in 16bit words, sum them, add carry over, negate to get sum

receiver
sum all received bits in 16bit workds, negate result to get 0000

weak
can freely swap data, 0-data is hidden

3.5.3 CRC

d = 4, used on links, ethernet, ADSL
n data bits, k check bits, n+k is evenly divisible by generator c

3.5.3.1 sender

extend n data with k (k = length(c) - 1) zeros, divide by generator value c, adjust k check bits by remainder

division
XOR fields if highest number non-zero, in each step take one new number

3.5.3.2 receiver

divide & check for 0 remainder

3.6 error correction

detect & correct error with accepted bigger overhead in frame size

3.6.1 hamming code

n = n^k - k - 1, n=4, k=3
check bits in positions p which are powers of two (starting at 1)
n = 4, k = 3 → message = 7, position 1,2,4 are check bits, check bit 1 covers 1,3,5; check bit 2 covers 2,3,6,7 etc

sender
write down message with empty check bits & filled in data bits, now calculate check bits to equal 0 (check bit 1: 0+1+1=0)

receiver
reconstruct check bits, append them to syndrome, if syndrome not 000 → error occurred. faulty bit at position it tells you

3.6.2 convolutional codes

Take a stream of data and output a mix of the recent input bits

3.6.3 LDPC

Low Density Parity Check → sparce matrix, best fit

3.7 detection vs correction

detection good if bursts of errors, no expected errors
correction good if some small all the time, no time for retransmission

3.8 in practice

correction with LDPC & convolution in physical
detection in link layer
correcting in application again (Reed-Solomon)

3.9 reliability

placed everywhere in the stack

3.10 ARQ

Automatic Repeat reQuest (acknowledges receive with ACK, if no ACK received by sender then retransmit)

timeout
too small = spurious resend, too big = link goes idle → easy in cable, hard in internet

how to avoid accepting duplicate frames

target
performance in common case, always correct

sequence numbers
ACK & Frame have it, to distinguisch them. one bit for stop-and-wait action

sliding window
generalization of stop and wait

3.11 multiplexing

sharing of resource

TDM
Time Division Multiplexing (high rate, fraction of time)

FDM
Frequency Division Multiplexing (low rate, all the time)

3.12 network traffic

3.12.1 bursty

TDM / FDM inefficient

3.12.2 multiple access

randomized
nodes randomize their resource access attempts (good for low load)

contention free
nodes order their resources (good for high load, guaranteed service)

3.13 ALOHA

random access, just send, if no ACK received then try to resend after a random timeout

3.13.1 improvements

divide time into slots

CSMA
Carrier Sense Multiple Access (listen for activity before sending)

CSMA/CD
... with Collection Detection, 2D distance needed (so everyone knows about a collision)

BEP
Binary Exponential Backoff: 1st coll wait 0 or 1 time, then 0 or 3 time, ...

3.14 Ethernet

uses BEP & CSMA/CD
no ACK or retransmission, CRC-32 for error detection

frame
preamle (8), dest (6), source (6), type (2), data (0-1500), pad (0-46), cks (4)

if dest is ff:ff:ff:ff:ff:ff then broadcast

3.15 wireless complications

cant listen while sending → no CSMA/CD
different areas of coverage

hidden terminals
terminals which can't hear each other but cause problems at receiver

exposed terminals
terminals which hear each other but cause no problems at receiver

MACA
sender issues RTC (with frame length), receiver issues CTS (with frame length), all nodes in range of CTS shut up

3.16 CSMA/CA

collision avoidance

sender avoid collisions by inserting small random gaps
random gap prevents to start at the same time

3.17 802.11 (wlan)

802.11 b/g/n on 2.4 Ghz, n/a on 5 Ghz, different amplitudes phases for different SNR, 6-50Mbps + error correction, n uses multiple attennas
uses CSMA/CA; RTC CTS optional, ARQ for frames (with ACK's), CRC-32 for error detection

frame
frame control (2), duration (2), receipient, trasmitter, address 3, sequence (2), data (0-2312), check sequence (4)

3.18 CSMA

good for low load (immediate access, little overhead)
not good under high load (high overhead due to collisions, access time varies)

3.19 Contention Free Multiple Access

3.19.1 Turn Taking

Token ring passes around Permission To Send

advantages
fixed overhead with no collisions (good under high load), regular change to send (no unlucky nodes, predictable quality of service)

disadvantages
more things can go wrong (what if tokes is lost, who is leader), higher overhead at low load

in pratice is random better than turn taking

3.20 Hub (or repeater)

physical layer only
all ports wired together to single wire

3.21 Switch

link layer
multiple frames may be switched in parallel, chooses the correct output
input & output buffers
100 Mbps for all ports intead of one

backward learning
forwards unknown destination address to all ports, remembers where ACK comes from

3.22 switch spanning tree

forwarding loops are a problem
subset of links that reaches all switches
broadcast will go up to the root of the tree & down all branches

3.23 spanning tree algo

all switches run same algorythm
start with no info
operate in parallel & send messages
search for the best solution

3.23.1 outline

elects root tree (lowest address)
shortest distance to root (lowest address to break ties)
turn off ports for forwarding if not part of tree

example messages send
(C,C,0) → (C, A, 1)

forwarding behaves as usual, but uses only the active links

4 NETWORK LAYER

4.1 why do we need it

switches don't scale wordwide (would have to broadcast itself to all other switches...), only work on one network tech, no traffic control

4.2 network layer approach

scaling
hierarchy in the form of prefixes

heterogenity
IP for internetworking

bandwidth control
lowest cost routing, later QOS (quality of service)

provides datagram (connectionless, message) & virtual circuit (connection-oriented, telefone)

4.3 routing

process of deciding in which direction to send traffic
network wide global & expensive

4.4 forwarding

process of sending the packet on its way
node process local & fast

4.5 store & forward package processing

router receive compelte packet and store it temporarily before forwarding
statistical multiplexing to share bandwidth
router has buffers for that, FIFO queues

4.6 datagram model

packet has destination address which is used by the router to forward it on its way
each router has forwarding table with next hop

4.7 IP (Internet protocol)

frame
version (4), Internet Header Length IHL (4) 5 ⇒ 5∗32bit, differentiated services (8) for QOS, total length (16) measured in bytes,
identification (16), nothing (1), don't fragment (1), more fragments (1),
fragment offset (14) angegeben in 8 bytes,
TTL (8) for ICMP, protocol (8), header checksum (16)
source (32),
destination (32),
options (0 or more)
payload (0- (2^16 - header size))

standard size is 20 bytes, max size is 2^16 (65536)

4.8 virtual circuit model

similar to phone connection, but no bandwidth is reserved (statistical sharing of links)
packets only contain a short label to indentify the circuit, only valid in circuit context
each router has forwarding table keyed by circuit, gives output line and next label to be placed on packet
connection established, circuit is set up (path is chosen, circuit info stored in routers)
data transfer, circuit is used (packets are forwarded along the path)
connection teardown, circuit is deleted (circuit info is removed from the routers)

4.9 MPLS

multi protocol label switching
virtual circuit technology widely used by ISP's
ISP sets up circuit before packet enters, adds MPLS label to IP packet at igress, removed at egress

4.10 datagram vs virtual circuit

setup phase
not needed vs required

router state
per destination vs per connection

address
packet carries full address vs packet carries short label

routing
per packet vs per circuit

fallures
easier to mask vs difficult to mask

quality of service
difficult to add vs easier to add

4.11 internetworking

connect multiple networks together

differences in networks
service model (datagram, VC)
adressing (what kind)
QOS (quality of service (with priority vs without)
packet sizes (bigger vs smaller)
security (encrypted, source authenticated, verfied forwarding,...)

internetworking hides the difference with a common protocol (IP)

4.12 IP prefixes

IPv4
written like 256.256.256.256

allocated in blocks
128.0.0.0/32 is one address, 192.168.12.3/16 are 2^16 addresses

more specific
higher prefix → less IPs

less specific
lower prefix → more IPs

forwarding with longest matching prefix

special address
0.0.0.0/0 matches all IP's

hosts in same network have same prefix

hosts send all trafic which does not belong to the network to the nearest router

splits
split in more specific for subnets

aggregation
announce less specific prefix (/18 & /18 & /18 → /16)

routers can split & aggregate on their own

4.13 allocating IP addresses

IANA International Assigned Numbers Authority delegates to regional internet registries
RIR delegates to companies in region

4.14 longest matching prefix

performance
less specific prefixes reduce table size, but longest match finding more complex than table lookup

4.15 DHCP

Dynamic Host Configuration Protocol, leases IP addresses to nodes

4.15.1 provides

network prefix, address of local router ("default gateway"), DNS server, time server, own IP address

4.15.2 on top of UDP

4.15.3 procedure (DORA the explorer)

discover
send broadcast to all nodes in network (address is all 1's, IP is 255.255.255.255, Ethernet is ff:ff:ff:ff)

offer
router sends response with IP it can offer

request
send a request for the IP

ack
router confirms

4.15.4 to renew an existing lease the client can start at request

4.16 ARP

address resolution protocol, matching IP addresses to link addresses (MAC)

4.16.1 problem

node needs link layer address to send packet to right destination

4.16.2 node maps with ARP IP addresses to lokal link layer address (needed for example for ethernet)

4.16.3 on top of ethernet (link layer, directly)

4.16.4 procedure

request
who has 192.168.0.1? tell 192.168.0.12 (this packet has source mac & ip and target ip correctly, but target mac is all 0)

reply
192.168.0.1 is at 00:23:23:01:01:22

4.17 packete fragmentation

connect networks with different max size packets
split up packets or discover largest size

MTU
maximum transmission unit (examples: 1.5k ethernet, 2.3k wifi)

want to use max size for efficiecy

4.18 fragmentation

split up too large packets
classic method but slow for routers
use fields of IP header, Total Lenght (always corresponds to the packet), Offset (indicates position), MF (more fragments, 1 on all but last), DF (dont fragment)

bad because
more work for routers, magnifies package loss, security vulnerabilities

4.19 discovery

find largest packet that fits on path
this is what is in use today
try to send large packet, hosts will respond if to big with their MTU

4.20 ICMP

error handling in forwading
sits on top of an IP packet (so inside)

4.20.1 if router has problem with IP packet

it sends back an ICMP with the error information and discards packet

4.20.2 ICMP has type, code, checksum & start of offending package

4.20.3 type/code

destination unreachable (net)
3 / 0 happens by lack of connectivity

destination unreachable (host)
3 / 1 happens by lack of connectivity

destination unreachable (fragment)
3 / 4 used for path MTU discovery

time exceeded (transit)
11 / 0 used by traceroute, TTL

echo request
8 / 0 used for ping

echo reply
0 / 0 used for ping

4.21 TTL

in IP header, decremented every router hop, causes ICMP error if hits 0
protects against forwarding loops
used by traceroute which starts with TTL 1

4.22 IPv6

1.1 billion internet hosts, 4 billion addresses in IPv4
128bit addresses, since 1998 standart, serious deployment since 2011

representation is 8 groups of ffff, can be abbreviated
remove leading zeros in groups, cut groups of all zeros (at max one time) and put : there
2001:0db8:0000:0000:0000:ff00:0042:8329 → 2001:db8:ff00:42:8329

frame
version (4), different services (8), flow label (20) same flow label means same treatment

payload length (16) contrary to IPv4 without header, next header (8) header extension or next package type, hop limit (8) TTL
source address (128)
destination address (128)

how to deploy? incompatible with IPv4

dual stack
support both

translators
convert packets

tunneling
IPv6 islands connected with IPv4 tunnels (IPv4 packet wraps around IPv6 in old environment) → need to insert IPv4 between IPv6 & link layer for tunnel

4.23 Middleboxes

sits inside network but does more than simple package processing to add new functionality

4.23.1 advantages

rapid deployment, control over many hosts

4.23.2 disadvantages

breaking layering interferes with connectivity; side effects

4.23.3 NAT

Network Address Translation, motivated by IP address scarcity, many internal hosts connected using a few IP's, uses TCP ports to distinguish hosts / connections
rewrite source & destination IP/port

disadvantages
can only deliver incoming packages if a connection has been setup (peer-to-peer & skype difficult)
doesn't work well with UDP
breaks apps that use direct IP (as FTP)

advantages;
many hosts between one IP
easy to deploy
firewall

4.24 bandwidth allocation

load sensitive routing
seconds at traffic hotspots

routing
minutes bc equiment faillures

traffic engineering
hours bc network load

provisioning
months bc network customers

4.25 delivery models

unicast
one sender, one receiver

broadcast
one sender, all are receiver

multicast
one sender, multiple specific receivers

anycast
multiple sender, multple receivers

4.26 goal of routing algorythms

correctness
find path that work

efficientcy
use network bandwidth well

fait paths
dont starve any nodes

fast convergence
recover quickly after changes

scalability
work great in bigger networks

4.27 rules of routing algorythms

all nodes are alike; no controller
nodes communicate only with messages, any learn only by them
nodes operate concurrently
messages / nodes / links may be lost at any time

4.28 best paths

latency, avoid circuitous paths
bandwidth, avoid small pipes
money, avoid expensive links
hops, reduce switching
→ approximate best by assigning cost to each link, best path is the path with the lowest cost

4.29 shortest path routing

dijkstra
have set of nodes, all distance infinity, choose shortest node N, relax all distances to other nodes
superlinear runtime
gives complete source / sink tree
requires complete topology

sink tree / source tree
shortest paths to all nodes to / from source node

4.30 distance vector routing

distributes version of bellman ford
very slow convergence after some faillures

4.30.1 settings

nodes know the distance to their nearest neighbour
nodes communicate only with their neighbour
all nodes run same algo
nodes and links may fail, message may be lost

4.30.2 each node maintaince a distance vector to all other nodes (distance to target & next neighbour to deliver)

4.30.3 → send own DV to all nodes (own distance set to 0), and continue till table full

4.30.4 if node disconnected, count to infinity scenario

4.30.5 fixes

split horizon
omit nodes learned from neighbour when sending him updates

poisoned reverse
set node distance to inf for nodes learned from neighbour

4.31 RIP

Routing Information Protocol
DV protocol
uses split horizon, poisoned reverse, 16 counts as infinity (limits network size)
30s resend, 180s means timeout

4.32 flooding

send message to all nodes in network
send message only to all other neighbours if not already sent → one node may still receive multiple copies of message
remembered using source & sequence number, only higher number will be send
use ARQ to make it reliable

4.33 link state approach

better dynamics than DV but more computation needed
same setting as DV used

4.33.1 procedure

flood topology, each node learns full toplogy with Link State Packet LSP
all neighbours receive packet with info of all neighbours source has, this packet gets then resend to all neighbours of those neighbours etc

each node computes own forwarding table using dijkstra

4.33.2 handling changes

resend LSP

4.33.3 things that can go wrong

sequence number reaches max, node crashes and forgets seq number.

4.33.4 fixes

include age on LSP's and forget old stuff

4.34 distance vector vs link state

correctness
distributed bellman-ford vs dijkstra

efficient paths
approx shortest path (both)

fair paths
approx shortest path (both)

fast convergence
slow vs fast (flood and compute)

scalability
excellent vs moderate

IS-IS
Intermediate System to Intermediate System

OSPF
Open Shortest Path First

4.35 multi path routing

allow multiple routing paths from one node to another

4.35.1 advantages

redundancy, performance, reliability

4.35.2 ESMP (Equal Cost Multi-Path Routing)

keeping path which have ties
modify Dijkstra & Bellman-Form

forwarding
source-destination pair (a "flow") should always go to the same path to avoid jitter

4.36 impact on routing growth

bigger forwarding table (larger router memory)
routing messages grow (all nodes need to be informed of larger topology)
routing computation grows (calculations grow faster than size of network)

4.37 hierachical routing

route to regions, not individual nodes
→ done with bigger prefixes
penalty is longer paths
one route inside a region from outside (saves time in resolving)
but multiple ways out

4.38 routing with multiple parties

networks group hosts as IP prefixes
networks are richly interconnected at IXP (Internet eXchange Points)

4.38.1 issues

scaling to very large networks, let parties choose routes by policy

4.38.2 hot potato routing

pass packet as fast as possible into network where it belongs (can lead to assymetries)

4.38.3 routing policies

TRANSIT
payed, like customer → ISP

PEER
free, traffic from respective customers can be exchanged

4.39 BGP

computes interdomain routes in internet
different parties are called Autonomous Systems (AS)
border routers of AS annouce BGP routes

annoucement
IP-prefix, path vector, next hop (A1, Swisscom, Orange, Router2)

policy implementation
only annouce path other parties can actually use → border routers of AS then select the best path

example exercise
[B, (AS1, AS2)] is one packet. Use PEER whenever possible (cause free), second choice is TRANSIT

5 TRANSPORT LAYER

5.1 trandsport layer services

provide different kinds of data delivery accross the network

unreliable messages
UDP datagrams

reliable bytestreams
TCP streams

5.2 TCP vs UDP

connections vs datagrams
bytes are delivered once, in order, reliably vs messages can be lost, reordered, lost
arbitrary length vs limited
flow control matches sender to receiver vs can send messages regardless of state
congestion control matches sender to network vs can send messages regardless of state

5.3 ports

client ports
chosen by OS, ephemeral (kurzlebig)

server ports
well-known chosen, <1024 needs admin priviledges

5.4 UDP

used by
VoIP (unreliable), DNS & RPC (message oriented), DHCP (bootstrapping)

buffers output from app, sends all asap

pseudoheader
source IP (32), destination IP (32), 0 (8), protocol (8), UDP length (16)

header
source port (16), destination port (16), UDP length (16), UDP checksum (16) 0 means no checksum, checksum over UDP segment & pseudoheader

5.5 TCP

need established connection before can send data
need to agree some parameters, for example MSS maximum segment size

three way handshake
- send SYN(x) (x is ISN initial sequence number)
- receive SYN(y)ACK(x+1)
- send ACK(y+1)

release connection
- active party sends FIN(x), passive sends ACK(x+1)
- passive sends FIN(y), active sends ACK(y+1)
- active after sending the ACK to FIN request of passive waits for timeout (ACK may have been lost, and passive may resends FIN)

header
source port (16), destination port (16)
sequence number (32)
acknowledgment number (32)
TCP header length (4), empty (4), various (8) (CWR, ECE, URG, ACK, PSH, RST, SYN, FIN), window size (16) relative to ack in bytes
checksum (16), urgent pointer (16),
options (n∗32)

sliding window
three duplicate ACK treated as loss

5.6 sliding window

need W = 2DB (bandwidth delay product) to fill network

5.6.1 LAR

Last Ack Received

5.6.2 LFS

Last Frame Send

5.6.3 Go-Back-N

simplest version
send while LFS - LAR $\leq$ W

on receive
if sqe number of ACK is LAR + 1 $\rightarrow$ advance sliding window (by increasing LAR by one) and send one more packet
otherwise discard packet

uses a single timer to detect losses, on timeout resends all buffered packets starting at LAR+1

needs W+1 seq. numbers

5.6.4 Selective Repeat

used by TCP
ack has highest received seq number + hints about out-of-order segments
uses timer per unacked segment, resends only specific ones
needs 2W seq. numbers

5.7 flow control

slow down enthusiatic sender
send WIN (flow control window, available space in receive buffer) together with ACKs to slow down sender (which takes the lower of W or WIN as window size)

5.8 timout

detecting losses with timouts

problem
too long wastes capacity, too short resends too often

adaptive timeout
estimation based on last timouts, 4th variance

6 CONGESTION CONTROL

6.1 congestion

traffic jam in networks
buffers are fifo, discard packets when full
congestion collapse!
delay & loss rise sharply with more load

6.2 bandwidth allocation

fair
every sender gets reasonably share of network

efficient
most capacity is used but no congestion

transport (reason for congestion) & network layer (notices congestion) must work together

hard because
sender/receivers change frequently, network is distributed & chaotic, capacity lacks in some parts

solution
senders adapt based on their view of the network and do this continuously, design adaptation that it is fair & efficient

6.3 fair bandwidth allocation

exact fairness hard to archive (A$\rightarrow$B$\rightarrow$C uses more resource than A$\rightarrow$B, efficency vs fairness contradict each other)
therefore $\rightarrow$ avoid starvation

6.3.1 max-min fair

property
increasing the rate of one flow decreases the rate of another

alogrythm
start with rate 0, increase till full, then go to others

6.4 bandwidth allocation models

open loop vs closed loop (open: reserve bandiwdth before transfer, close: use feedback to adjust rates)
host vs network support (who is the authority?)
window vs rate based (how is allocation expressed)
$\rightarrow$ TCP is window, host-driven, closed loop

6.5 AIMD

Additive Increase Multiplicate Decrease
sawtooth pattern, used by TCP

6.5.1 feedback signals

packet loss
TCP NewReno, Cubic TCP (Linux); Hard to get wrong, but hears about congestion late

packet delay
Compound TCP (Windows); Hear about congestion early, but need to infer congestion

router indication
TCP with Explicit Congestion Information ECI; hear about congestion early, but needs router support

6.5.2 implementations

TCP Reno
congestion window (cwnd) put over sliding window, adapted when package loss occurrs

6.6 TCP behaviours

6.6.1 flow control

ACK clocking
on ACK receive send new packets $\rightarrow$ automatically helps to adjust to slow links, packets do not queue up

adapting sending rate to receiver based in its max available sliding window (which is inside ACK of packet)

6.6.2 congestion control

Slow-start
doubling every cwdn every RTT, till first timout, restart and start with AI at cwdn / 2 $\rightarrow$ helps to cover wide range of network speeds

Fast Retransmit
three duplciated ACK treated as loss, resend packet if possible $\rightarrow$ can repair package loss before timeout, but still allows some reordering

$\rightarrow$ infering non-loss from ACK
as we continue to receive same ack, we still know new packages are received at receiver, so we can advance cwnd anyways

Fast Recovery
do fast retransmit, then MD cwnd, next pretend further received ACK are expected ACK's

6.6.3 TCP transmission rate determined by

minimum of flow control and congestion control
that way we solve congestion, adapt fast to a good sending rate and react fast to changes in connection quality

6.7 TCP Tahoe

Slow-start
start with cwnd = 1, cwnd + 1 for each ACK

additive increase
switch to AI when cwdn > ssthresh, +1 segement size each RTT, set ssthresh = cwnd / 2 after loss, slows start after timeout

6.8 TCP Reno

fast retransmit, fast recovery, slow-start, Multiplicate Decrease at 0.5

6.9 TCP NewReno

now has ECN, can handle multiple package losses

6.10 ECN

Explicit Congestion Notification
classic TCP need to experience congestion before it can react $\rightarrow$ ECN is set by router
marks IP header of packets, receiver sends this nofification back in ACK,

sender treats package as "lost" (adapts its cwnd)
for this to work bot senders & router have to be upgrated

6.11 TCP diagrams

start with slow start, at threshhold start with AI → if no threshhold continue with slow start till timout

if packet lost
set new threshhold at current cwnd/2 → continue there

if package timout
start with additive

7 APPLICATION LAYER

7.1 application layer

examples
http, DNS

in multiple packes, or multiple aggregated in one...

includes presentation & session layer too...

session
series of related network interactions in support of an application task

presentation
identify type of content & encode it properly: MIME, gzip

7.2 DNS

7.2.1 lookup service

name
high level resource identifier (url)

address
lower-level resource locator (IP)

resolution
mapping name to address

7.2.2 hierachical structure of URL's

7.2.3 starting with TLD, top level domains (generics & country)

7.2.4 Zone

contiguous portion of namespace (.edu, .eth.edu)
each zone has a nameserver to inquire

7.2.5 resource recods

SOA
start of authority, has main zone entries

A
IPv4 address of host

AAAA
IPv6 address of host

CNAME
canonical name for alias ("redirect")

MX
mail exchanger for domain

NS
nameserver of domain or subdomain

7.2.6 iterative vs recursive query

NS completes resolution and responds with final answer (caching serverside) vs NS responds with who to contact next (fire & forget for server)

7.2.7 caching

DNS entry has TTL

7.2.8 root nameserver configured by DHCP

7.2.9 13 root nameservers, 250 instances

7.2.10 ontop of UDP, ARQ, ID field links messages (16bit)

7.2.11 DNSSEC secure version of DNS

7.3 IP anycast

multiple locations advertise same IP, client picks best route (shortest, cheapest)

7.4 http

ontop of TCP, part of browser

7.4.1 commands

GET
read webpage

HEAD
read header

POST
append to webpage

PUT
store a webpage

DELETE
remove webpage

TRACE
echo incoming request

CONNECT
connect through a proxy

OPTIONS
query options

7.4.2 codes

1xx
information, 00 aggree to handle request

2xx
successfull, 00 all OK, 04 no content

3xx
redirection, 01 moved, 04 still valid

4xx
client error, 03 forbidden, 04 not found

5xx
server error, 00 internal, 03 try again later

7.4.3 header fields

browser capabilities
User-Agend, Accept-(Encoding, Charset, Language)

caching related
If-Modified-Since, Date, Etag, Expires, Cache-Control

browser context
Cookie, Referrer, Authorization, Host

Content delivery
Content-(Encoding, Length, Range, Type)

7.4.4 PLT

page loading time

7.4.5 HTTP/1.0

uses one connection for each request → multiple TCP connection, slow-starts, ...

7.4.6 decrease PLT

smaller images, gzip, caching CDN, avoid TCp slow start
multiple connections to server, which are not closed

7.4.7 HTTP/1.1

multiple connections to server

7.4.8 HTTP/2.0

includes SPDY
Multiplexed (parallel) HTTP requests on one TCP connection, client priorities, compressed HTTP headers, server push of resources

mod_pagespeed
better content structures, compile webpage to load faster, compression, minifying, etc

7.5 http caching

web caching
try to locally cache with Expires header, heuristics
revalidate copy with server with LastModified or ETag → get full response or Not-Modified back (304)

web proxy
intermediate between server & pool of clients
larger, shared cache for users
application of organisational access policies
→ CDN places site replicas directly inside ISP

Zips Law
kth's item's popularity is 1/k

7.6 peer-to-peer content delivery

7.6.1 delivery with server/client structure

advantages
efficient, scales up for popular content, reliable

disadvantages
dedicated infrastructure, central control

7.6.2 goal of p2p is to remove disadvantages

7.6.3 challenges

limited capabilities (how to depliver to all others) → use distribution tree
participation incentives (why help others?) → download & upload connected
decentralization (how to find content) → DHT (distributed hash tables), index which lists who to contact for content, this list is also shared amoung the peers

7.7 bittorrent

get torrent description
contact tracker to join & get list of peers (with at least one seed peer)
use DHT to find more
trade pieces with different peers
favor peers that upload fast, choke those who upload bad