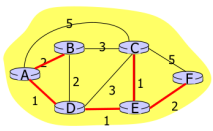
TCP
Connection-oriented, Reliable transport
Flow control, Congestion control
no timing, minimum bandwidth guarantees
UDP
no connection, not realiable
no flow or congestion controll
no timingt or bandwidth guarantee
best effort, Checksum: 1er-Kompl. 16bit Words of data + 12byte header
HTTP
stateless,asynchron, TCP, application layer
request, response
HTTP/1.1 GET, HEAD, POST, PUT, DELETE. TRACE, CONNECT
non-persistent: 1 object 1 TCP connection
Persistent(1.0): multiple object 1 TCP conn.
Pipelined persistent(1.1): multiple request without waiting of response. Answer in order
CDN:authoritative DNS route redirect requests
DNS: Record A (name, ip), NS (name, Auth. DNS), CNMAE (alias), MX (mail), AAAA (IPv6)
recursive: DNS does the work
iterative: DNS answer (to local DNS) with other DNS-Root: 13 weltweit, anycast (Anfrage an mehrere, Antwort von Host mit kürzester Route)
Anfragen an lokalen Router immer rekursiv, A
Anfragen an root- TLD-Server immer iterativ, NS
Anfragen an Domain-Server sind iterativ, A
TCP Reliablity:
Seq. #: Number of first byte in segment
ACK: Seq # of next byte expected; cumulative
Echo:
$A \rightarrow Seq = 42, ACK = 79, data = C \rightarrow B$
$A \leftarrow Seq = 79, ACK = 43, data = C \leftarrow B$
$A \rightarrow Seq = 43, ACK = 80$
Retransmission Timeout: 1 Timer for 1 Packet based on RTT, dynamisch:

$$\text{Neue RRT} = \text{alpha(alte RTT)} + (1 - \text{alpha})$$

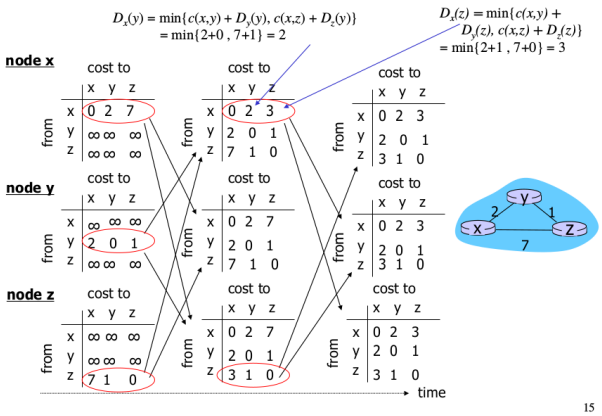
alpha meistens = 0,875
$\beta = 2$ exponentially back-off
Flow Control: sender won't overrun receiver's buffers by transmitting too much, too fast
RCV informs about window size, Sender transmitted unACK data < window size
Ideal Window:= delay · bandwidth
Congestion Control: Avoid Packet Drop (Bufferoverflow & delay)
Slowstart exp increase in window size
Congestion Avoidance: linear increase. Loss: threshold=cwnd/2
Tahoe: Slow Start, Cong Avoid., Timout/3 dup Acks: cwnd=1, slowstart
Reno: Slow Start, Cong Avoid., Fast Recovery, Timout: cwnd=1 slow start, 3 dup Acks: Fast Recoery, Cong Avoid.
C.Collapse: More load less throughput. **End-End CC:** No Feedback, **Network Assist:** Routers give Feedback
AIMD: Increase bandwidth till loss. cwnd increase at RTT. Loss cwnd/2
TCP Fairness: N TCPs, 1/n bandwidth;AIMD fair, MD decrease throuhput proportionally
IPv4: 32bit
Classes (wasted) A (0.0.0.0 to 127.255.255.255), B (128.0.0.0 to 191.255.255.255), C (192.0.0.0 to 223.255.255.255), D(mulitcast) (224.0.0.0 to 239.255.255.255)
Private: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 **LinkLocal:** 169.254.0.1/16 **LoopBack:** 127.0.0.0/8 **Multicast:** 224.0.0.0/4
IPv6: 128bit **Loopback:** ::1/128 **Global Unicast:** 2000::/3 **Unique Local:** FC00::/7 **Mulitcast:** FE00::/8 **LinkLocal Unicast:** FE80::/10
Multicast - adressieren einer Gruppe von Hosts mit einer Nachricht, link-local Adressen: nicht-routbare Adressen
Forwarding decision: Longest prefix match, table: Network/Mask, Nexthop, Interface **Routing:** build/update forwarding table
global: all router know topologie and costs (Link state algorithmen)
decentral: router knows neighbours and costs, exchange info (distance vector)
static vs. dynamic
Dijkstra: link state, k iteration know way to k.

Dijkstra's algorithm: Example

Step	start N'	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D		2,D	infinity
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					



BellmanFord: Distance vector, node periodically/when update sends vectors to neighbours



15

NAT: Source IP/Port $<->$ own IP/Port
InternetRouting: Router divided in AS.
Intra-AS: IGP, RIP (BellmanFord), OSPF (Dijkstra)
Inter-AS: BGP (node share route infos), AS-PATH (all AS), NEXT-HOP, Prefix + Attribute = Route, route selection: local pref, shortes AS-PATH, Best MED, closet HOP, addition crit., IP address of peer
In-band: HTTP (control and data 1 channel), **out-of-band:** FTP sperate for Control and Data
HardState: installed/removed by RCVR, valid unless told otherwise, better if more overhead, greater consistency
SoftState: installed/refresh by RCVR, invalid unless refreshed, robust, easier error recovery
LinkLayer: error dectetion/correction, broadcast channel (multiple access), addressing (MAC), (opt. realiable, flow control)
Point-to-Point vs. Broadcast (Channel Part., Random Access, Taking Turns)
MAC: 48bit, unique in ROM, **ARP:** IP, MAC, TTL, **IPv6 Neighbour Discovery:** IPv6, MAC, TTL
Ethernet: shared medium, connectionless, unreliable, CSMA/CD (Carrier Sense Multiple Access / Collision Detection), **expont. Backoff**: 1. choose k of $\{0, 1\}$, delay k*512 bit, 2.-9. k from $\{0, 1, 2, 3\}$, from 10. $\{0, 1, 2, 3, ..1023\}$
Signaling: Exchange of messages among network entities to enable (provide service) to connection/call (setup, teardown, update, end-end, end-network, netw-netw), **SIP:** Applicationlayer, available indepent from location, **RSVP:** reserve ressource end-end für QoS, sofstate, multicast
Indirection: via a third. ?Every problem in CS can be solved by adding another level of indirection.? **Multicast:** single transmit multiple rcvr. ack list at sender
Mobility: indirect routing: via homeagent, permanment address, care-of-address: redirect data, triangle routing; direct routing: sender gets new address, more transparent; registration: foreign agent registers mobile user at home agent
CDN: local/closer replicates, DNS - load balacing
Virtualization: (VPN) Networks perceived as being private networks by customers using them, but built over shared infrastructure owned by service provider
Overlay network: Create ?logical? links, RON: Resilient Overlay Networks: by building application overlay network, can increase performance, reliability of routing
QoS: Network provides applications with levels of performance guarantees needed for applications to function.
Types (service classes): Best-effort (elastic apps), Hard real-time (real-time apps, bounded loss / delay), Soft real-time (tolerant apps, probabilistic loss / delay)
Principles: Traffic **specification** (App, which class, how much bandwidth), Traffice **classification** (mark package to identify), Traffic **isolation** (protoct classes against eachother, app violates rule), Call **Admission** (Network blocks call, if it cannot meet needs), Resource **sharing** (resources as efficiently as possible)
QoS Internet: IETF Integrated Services (IntServ) - reserves ressources at routers on path, call admission: R-Spec: QoS Need, T-Spec: Traffic-Character, RSVP: transmit R-T-Spec independent from routing, IETF Differentiated Services (DiffServ): priors Packet-Flow, qualitive service classes, like a cable, marks packets, prio sheduling in core router
Internet End-to-End Argument: Netzwerkschicht: best effort Daten/Paket-Lieferung, Transportschicht: End-to-End Fehlerkontrolle (TCP), rest Application
Internet Design: Survivabilit (stateless, state at Endnode, recover from errors), Servicetype (TCP/UDP), IP over everything (distri Managm, cost efficent)