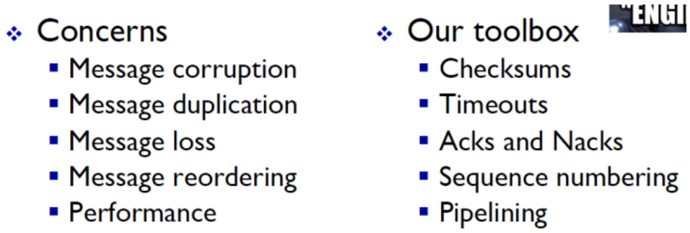
9331 期末复习讲义（续）

3. Transport Layer

3.1 概念

- Application layer is our boss, network layer is ours to command!
- Because…
  - Network layer only find path, doesn't guarantee delivery/order/path
- Therefore…
  - Transport layer need to be able to do it! Or not…
- Transport layer provide logical connection between **processes;** network layer provide communication between hosts
  - Segmentation and reassembling
  - Multiplexing and de-multiplexing: multiple sockets(processes/port)
  - Source port / **destination port**
  - Source IP / **destination IP**
  - Non-persistent HTTP will have **different** socket for each request(connection)
  - Welcoming socket/connection socket
- Connectionless demux/connection oriented demux
  - Connectionless: UDP. Defined by 2-tuple (dest IP + dest port)
  - Connection-oriented: Defined by 4-tuple (src IP and src port is also used)
- UDP
  - Segment header: 64 bits. Source port(return address), dest port, length(in byte, including header), checksum, payload
  - Checksum: 1's complements
    - Detect single (or odd number) bit error. Cannot detect 2 (or even) bit error!
    - Calculation: add together, wraparound carry bit, then take complement.
    - Calculated over header and data
    - Error correction unnecessary (periodic messages)
  - Application: DNS, DHCP, SNMP, Gaming, video/audio
- TCP (important)
  - Handshake/acknowledgement
    - Why these concerns?



- Concerns
  - Message corruption
  - Message duplication
  - Message loss
  - Message reordering
  - Performance
- Our toolbox
  - Checksums
  - Timeouts
  - Acks and Nacks
  - Sequence numbering
  - Pipelining

  - Feature: transport_part2 p21
    - Reliable, duplex, connection-oriented, flow controlled
  - segment header: 160+ bits
    - source port, dest port, seq(counting byte), ack#(counting byte), flags, checksum, receive window(expected byte#)

    - MSS(maximum segment size):
        - 1460 = MTU(maximum transmission unit, limit of IP datagram, 1500)-IP header(20)-TCP header(>=20)
    - Sequence number: ISN + k
        - ISN: initial sequence number
        - K: kth byte of stream
    - ACK sequence number
        - = next expected byte
        - =seq# + length
        - Cumulative ACK
    - Buffers out-of–order packet
    - Single timer
    - Fast retransmit: use duplicate(3) acks to trigger early retransmission
    - Rwnd: limit amount of in-flight data to rwnd value
    - Connection management
    - Syn-flooding and syn cookie

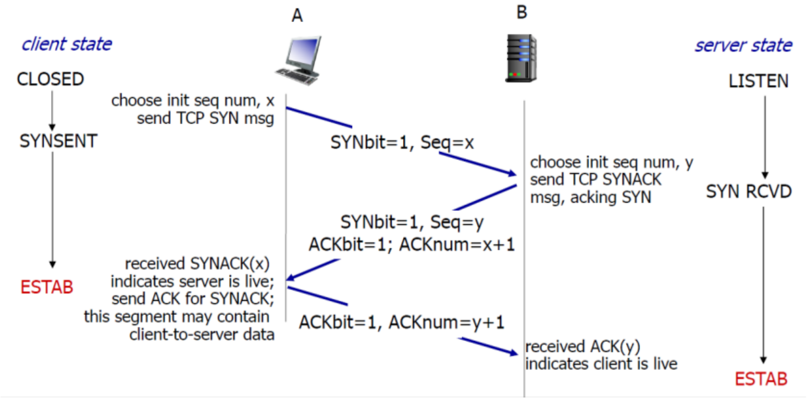
3.2 重点

- Develop a reliable transport protocol(rdt) step by step: p2p_transport
    - 1.0: assume network layer is reliable, transport layer doesn't have to do anything!
    - 2.0: deals with error bit. (partial) solution:
        - Error detection (with checksum)
        - Feedback: ack/nak
    - 2.1: deals with corruption of ack/nak, and possible duplication by:
        - Sequence number: sender add a number to pkt, receiver keeps track of which to expect
        - Stop and wait
        - Q: why only (0,1) sufficient for seq#?
    - 2.2 Nak free!
        - Ack last packet received
    - 3.0: deals with loss:
        - Timeout is introduced
    - Pipelined(sliding window) 3.0:
        - Deals with utilization: how?
        - Go-Back-N:
            - timer for oldest in-flight pkt
            - ack(n) means all packets up to n are received (cumulative ACK)
            - on timeout(n), retransmit packet n and all higher seq# in window
            - for receiver: out-of order pkt: re-ack
        - Selective repeat
            - Timer for each pkt
            - for receiver: Buffer out-of-order pkt, Ack individually
            - window size: <= (seq# range)/2
- TCP timeout estimation

  -
    ```
    EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT
          DevRTT = (1-β)*DevRTT +
                         β*|SampleRTT-EstimatedRTT|
                         (typically, β = 0.25)
    ```

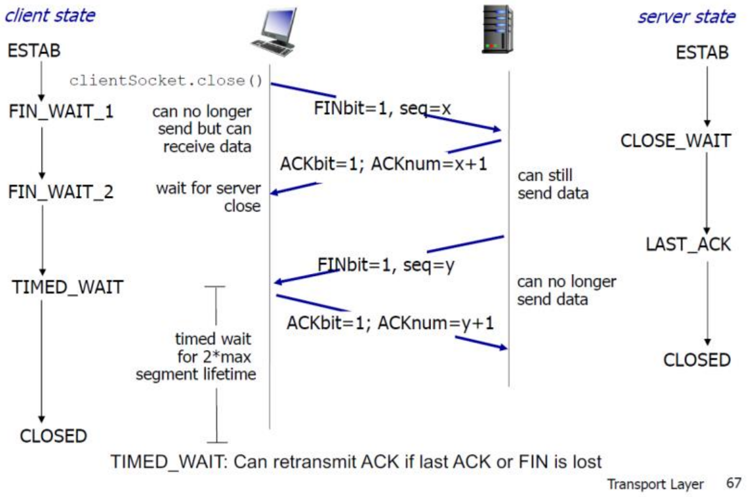
  -
    ```
    TimeoutInterval = EstimatedRTT + 4*DevRTT
    ```
  - Retransmission is excluded(p44)
- TCP connection management

  3-way handshake



Syn loss: timout(3s)

4-way termination



3.3 练习

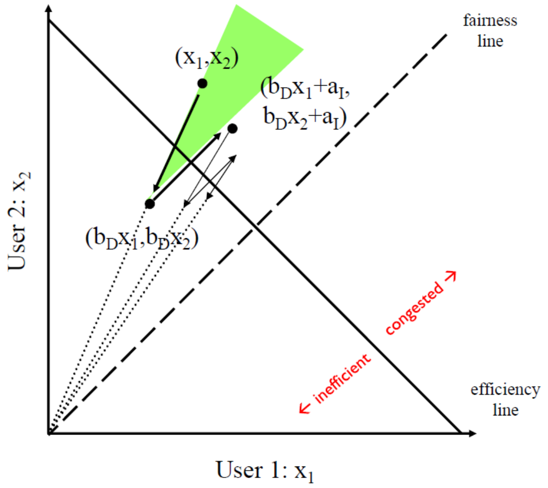
R8, R9, R14, R15, P14,P25

4. Transport Layer – Congestion control（期中考试之后）

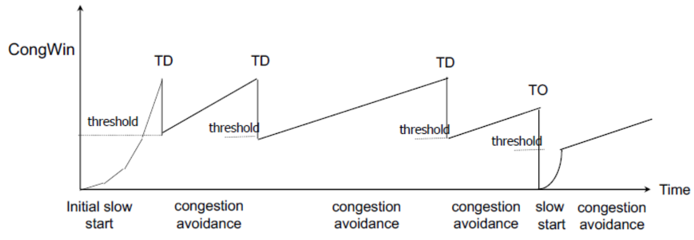
4.1 概念

- Congestion Collapse
- Load-Throughput-Delay graph
  - Why is the Cliff? Packet loss and congestion collapse

  - Solution: senders limit sending rate
- 2 approaches:
  - end2end(congestion inferred from loss & delay)
  - network-assisted(routers provide feedback)
- TCP's approach: controls number of packets in flight
  - $\text{rate} \approx \frac{cwnd}{RTT}$ byte/sec
- CWND: Congestion window, calculated by sender using algorithms below(SS, AIMD etc.)
- RWND: Advertised window, feedback from receiver
- Sender window = LastByteSent – LastByteAcked = min(CWND, RWND)
- MSS: Maximum Segment Size (TCP payload size), only for pedagogical purposes
- Dup ACK and Timeout: isolated loss vs. severe loss
- Rate adjustment:
  - 2 action: increase(upon receipt of ack)/decrease(upon loss)
  - 2 motive/phase: discovering bandwidth/adjusting to variation
- SS: Slow Start
  - Goal: discover/estimate bandwidth
  - Start slow/ramp up quickly
  - Double cwnd every **RTT:** exponential
- AIMD: Additive-Increase, Multiplicative Decrease
  - Adjust to varying bandwidth
  - Increase CWND by 1 every **RTT** (or cwnd = cwnd + 1/cwnd on every ACK)
  - Cut CWND in half after loss
  - When to stop slow start: slow start threshold
    - ssthresh =CWND/2 on every TD/TO
    - When CWND=ssthresh, switch from SS to AIMD
- Fairness(Important): AIAD vs AIMD

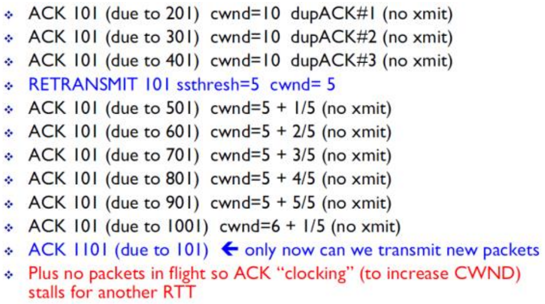


- Reno vs Tahoe vs New Reno: Tahoe set CWND to 1 on any loss, reno only cut in half on TD (Below is Reno). New Reno is Reno + fast recoverty

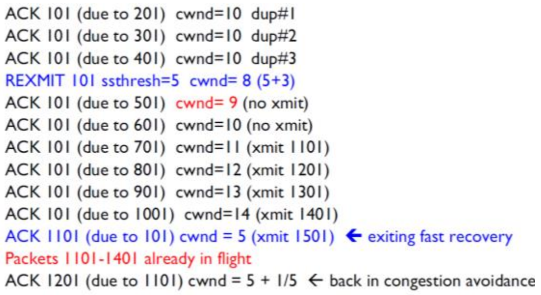


TD: Triple duplicate acknowledgements
TO: Timeout

- Fast recovery:
  - On TD: ssthresh = cwnd/2, cwnd = ssthresh + 3
  - cwnd = cwnd + 1 for each additional duplicate ACK
  - Exit fast recovery after receiving new ACK
  - set cwnd = ssthresh
  - No Fast recovery vs Fast recovery, 10 packet with packet size 100



- ACK 101 (due to 201) cwnd=10 dupACK#1 (no xmit)
- ACK 101 (due to 301) cwnd=10 dupACK#2 (no xmit)
- ACK 101 (due to 401) cwnd=10 dupACK#3 (no xmit)
- RETRANSMIT 101 ssthresh=5 cwnd= 5
- ACK 101 (due to 501) cwnd=5 + 1/5 (no xmit)
- ACK 101 (due to 601) cwnd=5 + 2/5 (no xmit)
- ACK 101 (due to 701) cwnd=5 + 3/5 (no xmit)
- ACK 101 (due to 801) cwnd=5 + 4/5 (no xmit)
- ACK 101 (due to 901) cwnd=5 + 5/5 (no xmit)
- ACK 101 (due to 1001) cwnd=6 + 1/5 (no xmit)
- ACK 1101 (due to 101) ← only now can we transmit new packets
- Plus no packets in flight so ACK "clocking" (to increase CWND) stalls for another RTT



ACK 101 (due to 201) cwnd=10 dup#1
ACK 101 (due to 301) cwnd=10 dup#2
ACK 101 (due to 401) cwnd=10 dup#3
REXMIT 101 ssthresh=5 cwnd= 8 (5+3)
ACK 101 (due to 501) cwnd= 9 (no xmit)
ACK 101 (due to 601) cwnd=10 (no xmit)
ACK 101 (due to 701) cwnd=11 (xmit 1101)
ACK 101 (due to 801) cwnd=12 (xmit 1201)
ACK 101 (due to 901) cwnd=13 (xmit 1301)
ACK 101 (due to 1001) cwnd=14 (xmit 1401)
ACK 1101 (due to 101) cwnd = 5 (xmit 1501) ← exiting fast recovery
Packets 1101-1401 already in flight
ACK 1201 (due to 1101) cwnd = 5 + 1/5 ← back in congestion avoidance

4.2 习题

P40. Consider Figure 3.58. Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions. In all cases, you should provide a short discussion justifying your answer.

a. Identify the intervals of time when TCP slow start is operating.

b. Identify the intervals of time when TCP congestion avoidance is operating.

c. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

d. After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

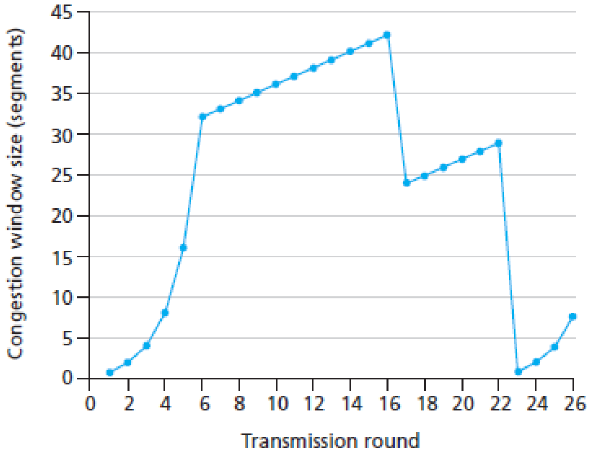
e. What is the initial value of `ssthresh` at the first transmission round?

f. What is the value of `ssthresh` at the 18th transmission round?

g. What is the value of `ssthresh` at the 24th transmission round?

h. During what transmission round is the 70th segment sent?

i. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of `ssthresh`?



Answer:

a) TCP slowstart is operating in the intervals [1,6] and [23,26]

b) TCP congestion avoidance is operating in the intervals [6,16] and [17,22]

c) After the 16th transmission round, packet loss is recognized by a triple duplicate ACK. If there was a timeout, the congestion window size would have dropped to 1.

d) After the 22nd transmission round, segment loss is detected due to timeout, and hence the congestion window size is set to 1.

e) The threshold is initially 32, since it is at this window size that slow start stops and congestion avoidance begins.

f) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 16, the congestion windows size is 42. Hence the threshold is 21 during the 18th transmission round.

g) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 22, the congestion windows size is 29. Hence the threshold is 14 (taking lower floor of 14.5) during the 24th transmission round.

h) During the 1st transmission round, packet 1 is sent; packet 2-3 are sent in the 2nd transmission round; packets 4-7 are sent in the 3rd transmission round; packets 8-15 are sent in the 4th transmission round; packets 16-31 are sent in the 5th transmission round; packets 32-63 are sent in the 6th transmission round; packets 64 – 96 are sent in the 7th transmission round. Thus packet 70 is sent in the 7th transmission round.

i) The threshold will be set to half the current value of the congestion window (8) when the loss occurred and congestion window will be set to the new threshold value + 3 MSS . Thus the new values of the threshold and window will be 4 and 7 respectively.

j) Threshold is 21, and congestion window size is 1.

k) Round 17, 1 packet; round 18, 2 packets; round 19, 4 packets; round 20, 8 packets; round 21, 16 packets; round 22, 21 packets. So, the total number is 52.

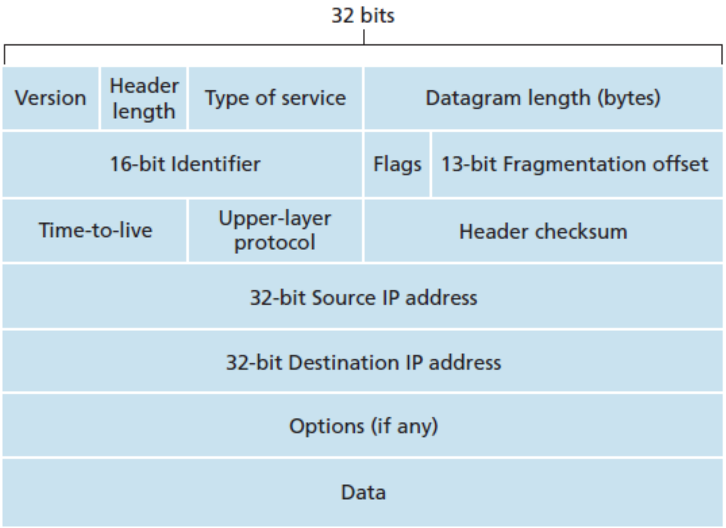
P46. Consider that only a single TCP (Reno) connection uses one 10Mbps link which does not buffer any data. Suppose that this link is the only congested link between the sending and receiving hosts. Assume that the TCP sender has a huge file to send to the receiver, and the receiver's receive buffer is much larger than the congestion window. We also make the following assumptions: each TCP segment size is 1,500 bytes; the two-way propagation delay of this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is, ignore slow start.
a. What is the maximum window size (in segments) that this TCP connection can achieve?
b. What is the average window size (in segments) and average throughput (in bps) of this TCP connection?
c. How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?
Answer:
a) Let W denote the max window size measured in segments. Then, W*MSS/RTT = 10Mbps, as packets will be dropped if the maximum sending rate exceeds link capacity. Thus, we have W*1500*8/0.15=10*10^6, then W is about 125 segments.
b) As congestion window size varies from W/2 to W, then the average window size is 0.75W=94 (ceiling of 93.75) segments. Average throughput is 94*1500*8/0.15 =7.52Mbps.
c) 94/2 *0.15 = 7.05 seconds, as the number of RTTs (that this TCP connections needs in order to increase its window size from W/2 to W) is given by W/2. Recall the window size increases by one in each RTT.

5. Network Layer

5.1 概念

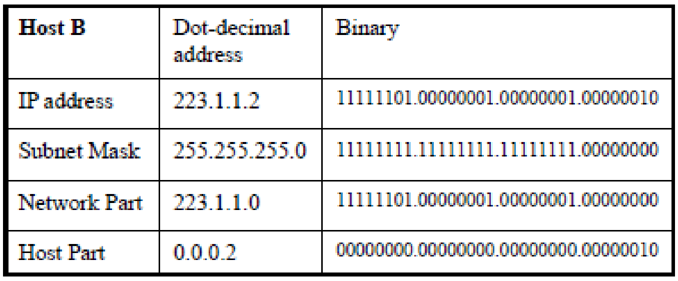
- Common IP protocol, supports any data-link/transport layer protocol
- Handled in every host/router
- Basic function: forwarding and routing
    - forwarding: move packets from router's input to appropriate output
    - routing: determine route from source to dest(can be done in an ad-hoc way or centralised way)
- Data plane vs Control Plane
    - Traditional or Per-router control plane
    - SDN control plane

- Network Layer service models
    - Bandwidth, loss, order, timing and congestion feedback
    - IP guarantees none: Best effort delivery of packet(globally)
- Network layer function:
    - routing protocol: path selection, determine forwarding table
    - IP protocol: addressing convention, datagram format, packet handling convention
    - ICMP protocol: error reporting, router signalling
- Potential problem and solution
    - Header corrupted: header checksum
    - Loop: TTL
    - Packet too large: fragmentation
- IP packet structure



    - Version: 4 or 6
    - Header length: number of **32-bit words** in header, typically 5
    - Total length: **bytes** in packet
    - ID: identification number
    - TTL: decrease by 1 at every router, until 1
    - Upper-layer protocol: TCP(6)/UDP(17)/…
    - Src IP addr/dest IP addr: 32 bit
    - Header checksum
        - 2-byte 1’s complement of **header**
        - discard if corrupted
        - recalculated at every router
    - TOS: type of service, differentiated services(voice, video etc)
- IP fragmentation
    - Reason: links can have different MTU(MTU includes 20 byte IP header)
    - Break big datagram to smaller one. Only one TCP header for each segment in this case. Normally TCP will adapt to MTU(**MTU discovery**) so TCP segment will be less than MTU-header, thus one TCP header per IP datagram
    - Offset is (Number of data bytes)/8
    - identification number remain same

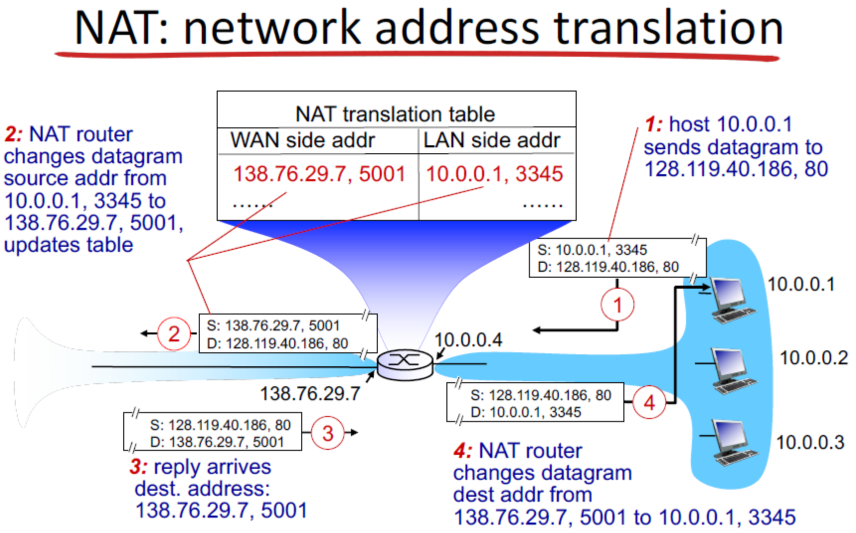
- Addressing(important)
  - Subnet and host part
  - Subnet: can physically reach each other without intervening **router**
  - Original->classful->classless
    - Original: 8 bit network, 24 bit host
    - Classful: only 3 network size
      - Class A 0xxxxxxx|host|host|host|
      - Class B 10xxxxxx|xxxxxxxx|host|host|
      - Class B 110xxxxx|xxxxxxxx| xxxxxxxx |host|
    - Subnet/Supernet
    - Classless: CIDR, classless inter-domain routing(slash notation)



Host B	Dot-decimal address	Binary
IP address	223.1.1.2	11111101.00000001.00000001.00000010
Subnet Mask	255.255.255.0	11111111.11111111.11111111.00000000
Network Part	223.1.1.0	11111101.00000001.00000001.00000000
Host Part	0.0.0.2	00000000.00000000.00000000.00000010

      - A.b.c.d/x, x is subnet portion of address in bit
      - Broadcast Address: host part all 1
      - Subnet address: host part all 0
      - Other: normal host
      - X increased as closer to host
- DHCP
  - Allow host to dynamically obtain its IP address from network server when it joins network
  - Host(DHCP discover)->Server(offer)->Host(request)->Server(ack)
  - Can also return address of first hop router, DNS server and network mask
  - Based on UDP. Sent to MAC FFFFFFFFFFFF
  - Port 67(server side), 68(client side)
  - Attack: DoS, masquerading
- Route aggregation
  - hierarchical addressing allows efficient advertisement of routing information
  - When switching networks, add specific route rule
  - Longest prefix matching
- ICMP
  - Network layer protocol, works above IP
  - Used in traceroute, by using TTL
- NAT
  - Private address:
    - 10.0.0.0/8 (16,777,216 hosts)
    - 172.16.0.0/12 (1,048,576 hosts)
    - 192.168.0.0/16 (65536 hosts)
  - Translation: change, remember, change back
  - Pro: saves ISP addresses, can change internal address/ISP without affecting other side
  - Con: port number limit: 60000 connections. Breaks layer constraint.

    - NAT practical issues:
        - Modifies port# and IP address, checksum recalculation needed
        - Port and IP may be in payload: need to modify application layer data! Encryption/length variation/TCP sequence number
        - Traversal problem workaround: port forwarding/UPnP IGD/relay



NAT: network address translation

NAT translation table

WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
……	……

2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table

1: host 10.0.0.1 sends datagram to 128.119.40.186, 80

3: reply arrives dest. address: 138.76.29.7, 5001

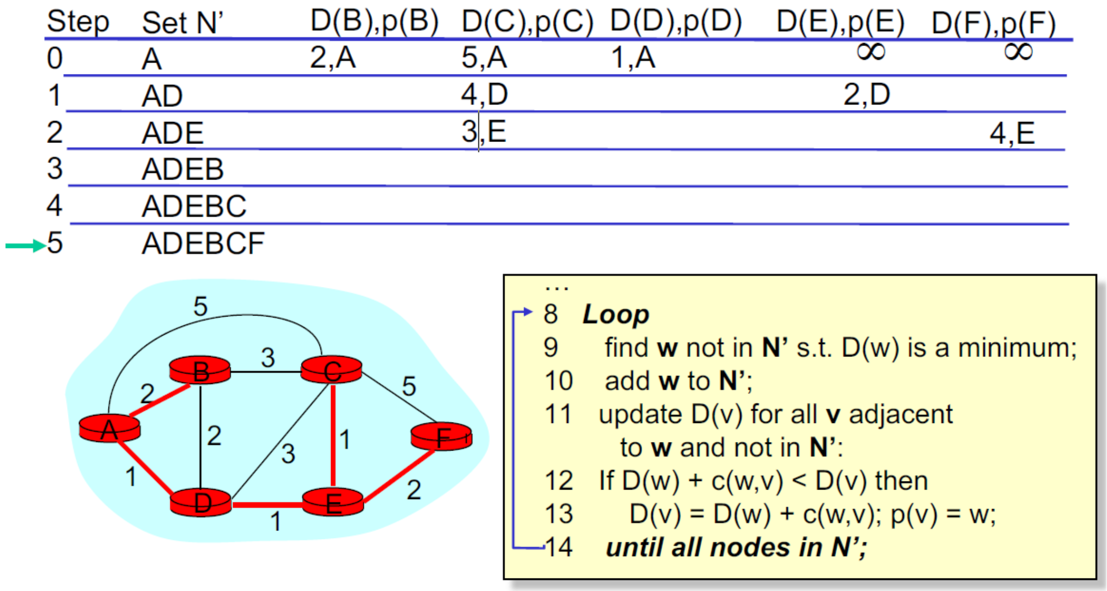
4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

- IPv6:
    - 128 bit address
    - 40 byte header
    - No fragmentation allowed
    - Colon hexadecimal notation + zero compression
    - Tunnelling: using IPv4 as transport

6. 网络层-路由

- Routing algorithm determine end2end path, forwarding table determine local forwarding
- 2 network control plane structure:
    - Per-router control
    - Logically centralized control
- 2 approaches for per-router control plane:
    - Link State(OSPF), Global
        - Router maintain topology(link cost) of whole network
        - Exchange LSA to all routers
        - Converge quickly
        - Limited network size
    - Distance Vector(RIP), decentralised
        - Router maintain next hop&cost
        - Changes propagate iteratively from neighbour to neighbour
        - Converge after multiple rounds
        - Scales to large network
- Graph: vertex/node/router, edge/link, weight/load
- Link state routing

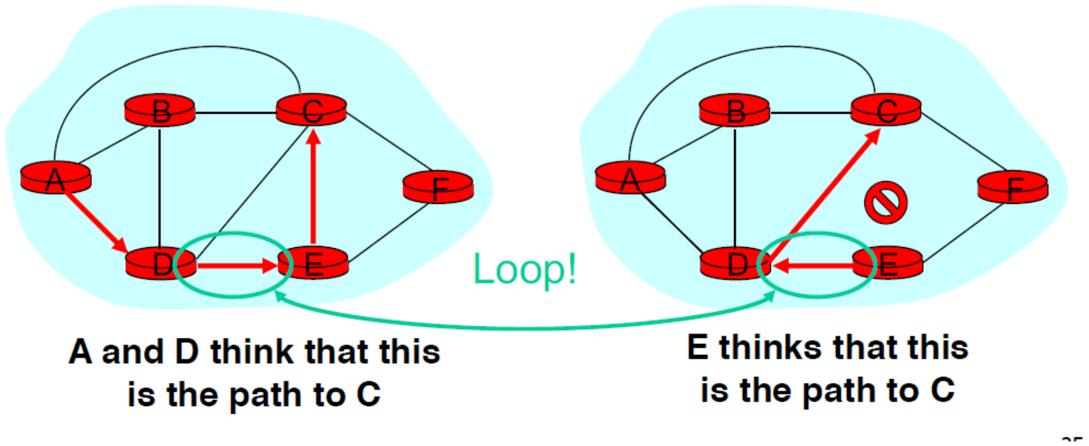
- Transmit LSA (Link State Advertisement), forward to all neighbour except incoming, do not forward previously seen LSAs
- Eventually each router learns entire network topology, then use **Dijkstra** to compute shortest path



Step	Set 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	∞	∞
1	AD		4,D		2,D	
2	ADE		3,E			4,E
3	ADEB					
4	ADEBC					
→5	ADEBCF					

```
…
Loop
  find w not in N' s.t. D(w) is a minimum;
  add w to N';
  update D(v) for all v adjacent
    to w and not in N':
  If D(w) + c(w,v) < D(v) then
    D(v) = D(w) + c(w,v); p(v) = w;
until all nodes in N';
```

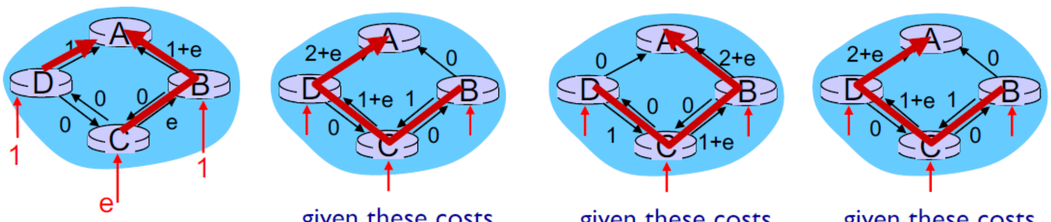
- Issue: scalability
  - LSA messages: O(N*E), every node's LSA need to travel on every link at least once
  - Dijkstra: O(N*N), or O(N*log(N) + E)
  - LS topology database: O(E)
  - Forwarding table: O(N)
- Issue: Transient Disruptions
  - Forwarding loops



A and D think that this is the path to C

E thinks that this is the path to C

  - Oscillation



initially

given these costs, find new routing…. resulting in new costs

given these costs, find new routing…. resulting in new costs

given these costs, find new routing…. resulting in new costs

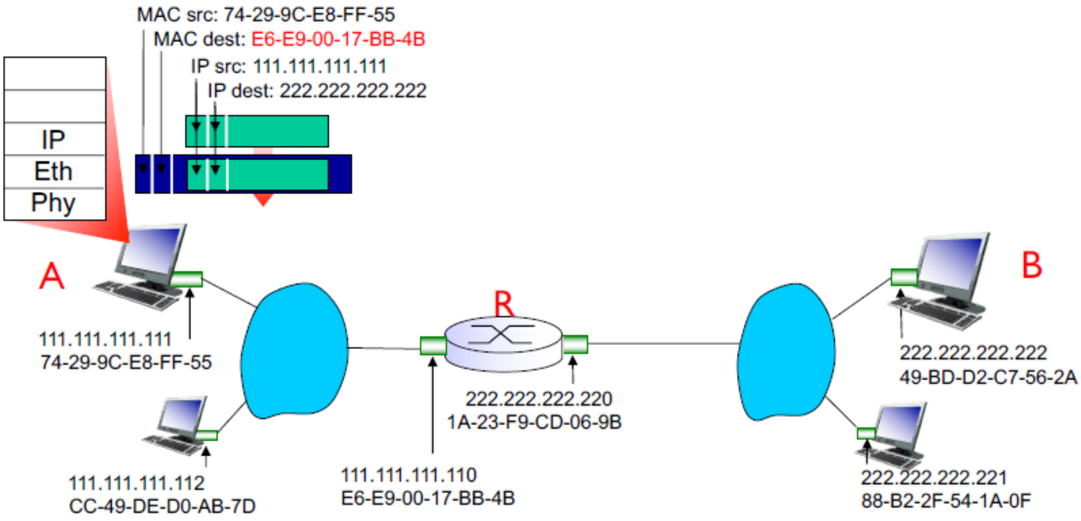
- Distance vector routing
    - Distance(cost to destination) and vector(destination network)
    - Routing table: destination, cost, next hop
    - Message exchanged between routers: destination, cost
    - Core: When J tells K it's cost to X is P(J,X), K will check: If p(K,J)+p(J,X) < p(K,X): use the route via J. Otherwise, use the existing route
    - Algorithm:
        - Periodically, each router sends a copy of its table (destination, cost columns only) to directly connected routers
        - When router K receives table from a neighbouring router J, K updates its table if:
            - – J knows a shorter route for a given destination
            - – J knows a destination K didn't know about
            - – K currently routes to a destination through J and J's cost to that destination has changed
    - Note: if cost of an attached link changed, then update table too.
    - Count to infinity + poisoned reverse:
        - When one link cost changes, takes long time for bad news to travel
        - If B routes through C to get to A: B tells C its (B's) distance to A is infinite

习题：P4, P10, P19, P23, P26, P30

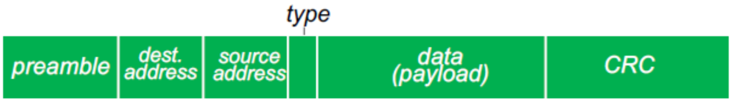
7. 数据链路层

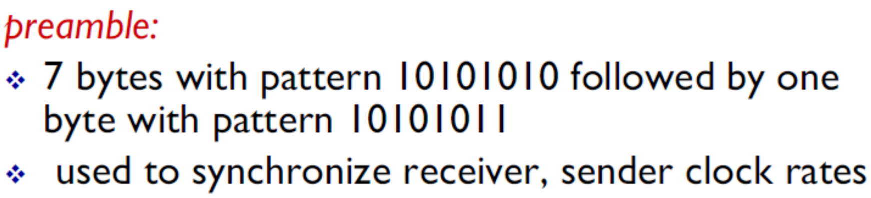
- Links: communicating channels connecting adjacent nodes
    - Can be wired or wireless
    - Link layer packet is called frame
    - Link layer is responsible for reliable delivery between physically adjacent nodes
- Transportation analogy
- Services provided
    - Framing(adding header including MAC), link access(shared media)
    - Reliable delivery between adjacent nodes
        - Optional for low bit error link
        - Why both link-level and end-end reliability?
    - Flow control
    - Error detection/correction
    - Half duplex and full duplex

- Implemented in adaptors, aka NIC
- MAC address: 48 bit, often written as hexadecimal notation.
  - Compare with IP: hard-coded vs configured, flat vs hierarchical, portable vs not portable, interfaces on same network vs globally
- ARP: Address resolution protocol, like DNS for link layer-> network layer
  - ARP table: IP, MAC, TTL
  - ARP discovery: broadcast ARP query to FFFFFFFFFFFF
  - No manual configuration
- Addressing in different LAN: need to find router info via DHCP
  - Walk through: send datagram from A to B via R



- ARP cache poisoning
  - Hacker can fake ARP response, pretending to be router
  - Solution: port security(i.e. one MAC per port), static ARP, monitoring tool
- Ethernet
  - Topology and collision domain
    - Bus: same collision domain, need **CSMA/CD**
    - Start: uses switch, no collision
    - Switched LAN vs broadcast LAN
  - Ethernet frame structure:
    - Preamble, addresses(when do we pass to network layer?), type, data, CRC



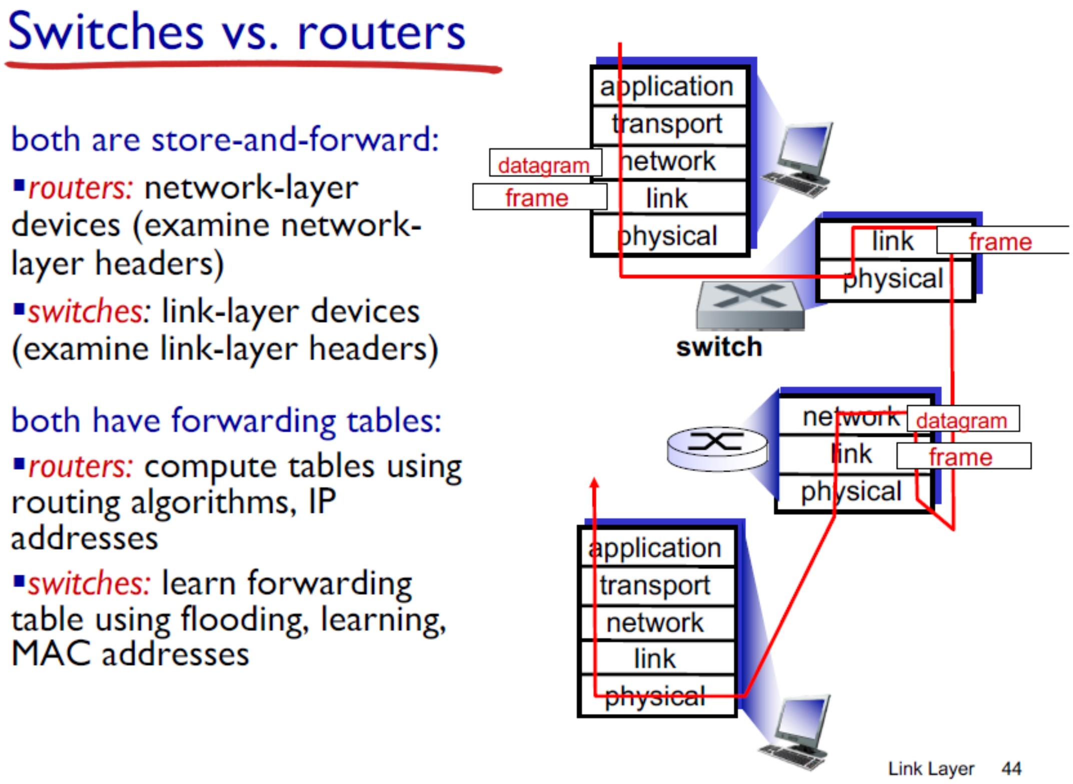


preamble:

- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

  - Connectionless, unreliable

  - MAC protocol: Unslotted CSMA/CD with binary backoff
  - 100BASE-TX, 100BASE-FX
- Ethernet switch
  - Store and forward Ethernet frames
  - **Selective** forward based on MAC
  - Uses CSMA/CD
  - Transparent, plug-and-play, self-learning
  - Separate collision domain on different links, possible to transmit multiple pair of ports
  - Self-Learning: record incoming data, send to know dest or flood(can be sniffed by hacker)



Switches vs. routers

both are store-and-forward:

- *routers:* network-layer devices (examine network-layer headers)
- *switches:* link-layer devices (examine link-layer headers)

both have forwarding tables:

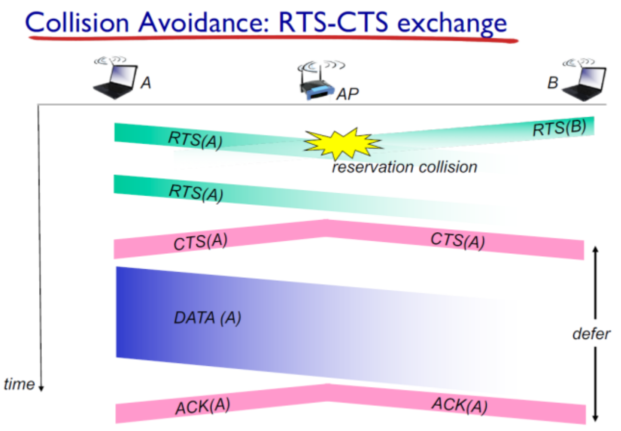
- *routers:* compute tables using routing algorithms, IP addresses
- *switches:* learn forwarding table using flooding, learning, MAC addresses

习题：P31

8. 无线网络

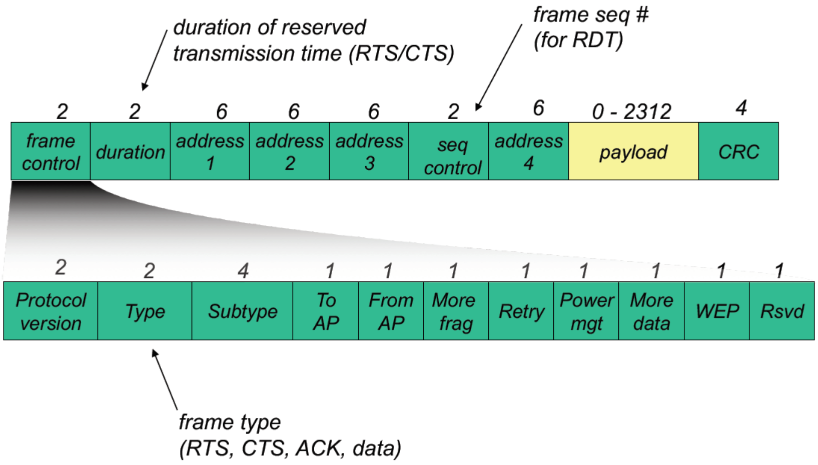
- Widely used, convenient
- Challenges: **wireless** and mobility
- Wavelength * frequency = speed of light: λ*f = c
- Elements:
  - Wireless hosts (laptops, mobile phone)
  - Base station (Wi-Fi AP, cell tower)
  - Wireless link (host<->base station, backbone link)
    - Indoor wireless link technologies: 802.11 b, (a,g), n (speed increase)
    - Outdoor: 4G:LTE, WiMax, 3G: WCDMA-HSPDA, CDMA2000-1xEVDO
- Modes:
  - Infrastructure mode

        - Base station connects mobiles into wired network
        - Handoff: transfer connection
    - Ad hoc mode
        - No base station
        - Nodes<->nodes
- Wireless link characteristics
    - Decreased signal strength
        - FSPL(Free Space Path Loss): $(4\pi d/ \lambda)^2$
    - Interference
    - Multipath propagation
        - Self interference
- SNR: signal to noise ratio
    - BER: bit error rate
    - To improve BER: increase power->increase SNR
    - SNR also varies according to modulation technique/rate
- Hidden terminal:
    - Hosts may not hear each other interfering at another host
- Exposed terminal
    - Carrier sense would prevent a ransmission even there is no interference
- 802.11 LAN architecture
    - Base station is AP
    - BSS(basic service set, or celll): hosts and AP, sometimes hosts only(adhoc)
- Channel
    - 11 channel, with overlap
    - Host scan channels, listening for beacon frames(containing AP's name(SSID) and Mac)
    - Then select AP to associate with, and use DHCP to get IP address
    - Passive scanning and active scanning: beacon frame vs Probe request/response
- Multiple access
    - Avoid collision(**CSMA/CA**). Hard to detect collision, because of hidden terminal/exposed terminal
    - Goal: detect if receiver can hear sender. Tell senders who might interfere with receiver to shut up
    - Sender
        - If sense channel idle for DIFS(distributed inter frame space), then transmit entire frame
        - If sense busy, binary exponential random back-off
            - Timer only count down when channel idle
    - Receiver
        - Send ACK after SIFS(shortest inter frame space)
    - RTS(request to send)/CTS(Clear to send)
        - Sender sends small RTS
        - Receiver send CTS, heard by all nodes
        - Sender transmit after CTS, other defer transmission



Collision Avoidance: RTS-CTS exchange

- 802.11 frame
  - Contains 3 MAC: AP MAC, Host MAC, Dest(Router) MAC
  - 3 MAC is slices to 2 (Host and router) on the 802.3 link from AP to router



- Rate adaption: when SNR decrease (distance increased), BER increase. When BER too low, switch to lower transmission rate
- Power management: node can sleep until next beacon, and next beacon contains data to be sent to node.

9. 网络安全

- 4 elements:
  - Confidentiality (cryptography and more)
  - Authentication
  - Message integrity
  - access and availability
- Threats
  - Eavesdrop（窃听）
  - Insert
  - Impersonation
  - Hijacking: take over ongoing connection
  - Denial of service
- Cryptography

  - m: original/plaintext message
  - K_A(m): cyphertext, m encrypted with K_A
  - m = $K_B(K_A(m))$, B is decryption key, A is encryption key
    - If A=B, symmetric
  - Substitution cipher: easy to break. Brute force, statistic, know-plain text etc.
  - Playfair Cipher: same row: right; same column: below; other: diagonal
  - Stream Cypher(one bit at a time) vs Block Cypher(break to equal size block)
    - Keystream vs block mapping
    - Exclusive or (XOR)
    - Cipher Block Chaining: use last encrypted block as XOR input, to prevent same block encrypted as same cyphertext
    - c(i) = K_S(m(i) XOR c(i-1))
    - m(i) = K_S(c(i)) XOR c(i-1)
    - first block: IV
  - Public key cipher
    - RSA:
      - N=pq, z = (p-1)(q-1)
      - E is relative prime with z
      - D: ed mod z = 1
      - $$m = \underbrace{(m^e \bmod n)}_{c}{}^d \bmod n$$

$$\underbrace{K_B^-(K_B^+(m))}_{\substack{\text{use public key first,}\\ \text{followed by}\\ \text{private key}}} = m = \underbrace{K_B^+(K_B^-(m))}_{\substack{\text{use private key}\\ \text{first, followed by}\\ \text{public key}}}$$

    - Used in SSL to exchange symmetric key
- Authentication
  - Using a nonce R, to prove someone do have the key
  - Man in the middle: hard to prevent only using cryptograph
  - CA: certification authority, binds public key to particular entity
- Message integrity
  - Digital signature(using asymmetric encryption)
  - Message digest(hash function, should be hard to find same one), MD5, SHA

10. 辨析-各种表

- ARP table: IP, MAC address, TTL. Exist on every node.
- Forwarding table
  - on router: IP, port
  - on switch: switch table
- Routing table (in DV routing): Destination IP, Min Distance (cost), and next hop. Exist on every router.
- Switch table: MAC address, port, TTL. Exist on self-learning switches