CS168 NOTES

Physical ports: where links connect to switches

Logical ports: where applications connect to stack

Control plane: compute routing table

Data plane: use routing table to forward packets

statistical multiplexing: combine demands to share resources efficiently

Reservation: flow level: circuit switching

On demand: packet level: packet switching

level of utilization = A/P (on demand higher)

bandwidth-Delay Product(BDP) = bandwidth \* propagation delay

transmission delay (link)

* how long take to push all bits into a link
* packet size/Transmission rate of link

propagation delay (link)

* how ling take to move one bit from one end to the other
* link length/propagation speed of link

queueing delay (traffic)

* how long a packet sit in a buffer before processed
* Arrival process
  + Average rate A; peak rate P
* W average packet wait time
* L average # packets waiting in queue
* L = A \* W

\*\*Layering

Bits on wire (Physical)

Deliver packets across local net work (Datalink)(L2)

Deliver packets across country (Network)(L3)

Deliver packets reliably to process (Transport)(L4)(end-to-end)

Do sth with data (Application)

layer depends on layer below, and support layer above

where modules are implemented: end-to-end principle

where state is stored: fate-sharing

all layers at host; switch support Physical and Datalink; router support Physical Datalink and Network

header: L1,L2,L3…

\*\*Routing

necessary (only if): If routing state valid, no loops/dead end

sufficient (if): If no loops/dead ends, then routing state is valid.

ways to avoid loops

* tree like topology (Learning switch L2)
  + Local-Area-Networks
  + Ethernet
  + Spanning Tree Protocol (STP)
    - pick lowest ID as root
    - compute shortest paths to the root
    - break tie by ID
    - weakness
      * require loop-free topology
      * slow reaction to failures and host movements
* Global view (Link-state and SDN routing)
  + Enterprise and Carrier
  + intradomain
  + OSPF (link-state routing), RIP
    - Global flood: each router’s link-state
* Distributed computation
  + inter domain
  + BGP
  + Min Distance vector (RIP)
  + Bellman-Ford
    - d(u, v) = Min(nbr w)[c(u,w) + d(w, v)]
  + Protocol
    - When to send update?
      * When elements of DV change
      * periodically
    - Poisoned Reverse
      * ex: z->y->x, z告诉y d(z, x) = inf
    - Split Horizon
      * like poisoned reverse
      * no entry instead
    - Poison a route
      * send inf when no longer have path

\*\*Transport

goals for reliable transfer

* correctness
* efficiency
* timeless
* fairness: how well it plays with others

reliable iff it resends all dropped or corrupted packets

feedback from receiver

* ACK: yes, i got packet
* NACK: no

send packet, set timer, resend if no ACK when time expires, reset timer

RTT: sending 1st put to receive 1st ACK

windowed based also condition: RTT \* B = W \* Packet Size B \* RTT = 2 \* BDP

Individual ACK

* Strength
  + know fate of each packet
  + impervious to reordering
  + simple window algorithm
* Weakness
  + lots if ACK need retransmission

Cumulative ACK

* ACK the highest sequence nu,her for which all previous packets have been received
* Strength
  + recovers from lost ACKs
* Weakness
  + confused by reordering
  + incomplete info about which packets have arrived

Full information feedback

* list all packets received
  + highest cumulative ACK and additional packets
* Strength
  + Max info
  + Resilient form of individual ACKs
* Weakness
  + Require sizable overhead in bad cases

packets reordered

* individual, full: not a problem
* cumulative: create Dup ACKs

packets delayed

* timeout for all designs

packets duplicated

* produce duplicate ACKs

Detection of loss

* timeout
* arrival of subsequent packets (Dup ACKs)

Designing IP

IPv4 header (20 Bytes)

A: 4-bit Version

* version of IP protocol
* 4 for IPv4; 6 for IPv6

B: 4-bit header length

* typically 5

C: 8-bit Type of Service (TOS)

* Allow packets to be treated differently based on needs

D: 16-bit Total Length (Bytes)

* Number of bytes in packet
* max = 2^16-1

E: 16-bit identification

* Which fragments belong together

F: 3-bit flags

* R: Reserved: ignore
* D: DF: Dont fragment
* M: MF: More fragments coming

G: 13-bit fragment offset

* portion of original payload this fragment contains

H: 8-bit Time ti Live (TTL)

* Decremented at each loop, packet discarded if reaches 0, and send “time exceeded” to source

I: 8-bit Protocol

* Identify high-level protocol
* 6 for Transmission Control Protocol (TCP)
* 17 for User Datagram protocol (UDP)

J: 16-bit Header Checksum

K: 32-bit Source IP address

L: 32-bit Destination IP Address

M: options

* Field Size(bits) Description
* Copied 1 set if field copied to all fragments
* Class 2 0=control, 2=debugging/measurement
* Number 5 specifies option
* Length 8 size of entire option
* Data variable option-specific data
* EX: Record Route, Strict Source Route, Loose Source Route, Timestamp, Traceroute, Router Alert

N: payload

Read packet correctly

* ABD

Get packet to the destination

* KL

Get responses to the packet back to source

Carry data

Tell host what to do with packet once arrived

* I

Specify any special network handling of the packet

* CM

Deal with problems that arise along the path

Potential Problems

* Header Corrupted
  + Checksum
    - if not correct, router discards packets
    - checksum recalculated at every router
* Loop
  + TTL
* Packet too large
  + Fragmentation
    - Every link has Maximum transmission unit (MTU)
    - router split a packet into multiple fragments if size too big
    - must reassemble to recover original packet
      * where
        + destination

IPv6

* version; Traffic Class; Flow Label; Payload Length; Next Header; Hop Limit; Source Address; Destination Address
* motivated by address exhaustion
* 4 times as big
* philosophy of changes
  + Don’t deal with problems, leave to ends
    - Eliminated fragmentation
    - Eliminated checksum
  + Simplify handling
    - New options mechanism
    - Eliminated header length
  + Provide general flow label for packet
    - not tied to semantics
    - provided great flexibility

Routing vs Forwarding

* Routing
  + Control Plane
  + Compute paths the packets follow
  + Distributed protocol leads to state at each router
  + slowly converge
  + MUST scale to size of network
* Forwarding
  + Data Plane
  + Directing a data packet to an outgoing link
  + Individual router using routing state
  + Must be done quickly

Addressing requirements

* Scalable routing; efficient forwarding
* host must recognize packet is for them

L2 addressing

* host addressing: MAC addresses
  + Numerical address associated with a network adapter
  + 48 bits
  + unique
* forward based on switch address
* Switch : Host

L3 addressing

* host addressing: Network
* forward based on Network field
* Network : Host
* Aggregation
  + Single forwarding entry used for many individual hosts
  + EX: Scaled L2: aggregate was switch; L3: aggregate was network
  + advantage
    - fewer entries
    - more stable
* Mapping between identifier and locator
  + scaled L2: used learning to map host to switch
  + L3: uses DNS
    - name -> identifier
    - IP address -> locator

Original Internet address

* 8 for network address
* 24 for host address

Glassful Addressing

* A: first byte < 128 : 0\*\*\*\*\*\*\* /8
  + 126 nets
  + 16M hosts
* B: 128-191: 10\*\*\*\*\*\* /16
  + 16K nets
  + 65k hosts
* C: 110\*\*\*\*\* /24
  + 2M nets
  + 254 hosts
* D: 1110\*\*\*
  + Multicast
* E: 11110\*\*
  + Experimental

CIDR

* Classless Interdomain Routing
* must specify address and mask
* flexible division between network and host address

Hierarchical addressing

* 12.34.158.0/23
* 23 bits network address, 9 bit host address

Forwarding

Router

* input linecard
  + Update IP header
    - TTL;Checksum; Option; Fragment
  + Lookup output port for the destination IP Address
  + Queue the packet at the switch fabric

Efficient Packet lookup

* Original or Glassful
  + Hash table
* CIDR
  + Longest prefix match (LPM)
    - Record port associated with first match, and only override when it matches another prefix during walk down tree
    - If ever leave path, you are done, last match prefix is answer

Disjoint prefixes to LPM table

* Start with disjoint prefixed marked
* move prefix upward one at a time
* delete prefixes no longer needed due to LPM
* delete subtree no longer needed to reach prefix

Discovery

* Address Resolution Protocol (ARP)(link layer)
  + every host main ARP table {IP address : MAC address}
  + ARP header
    - 16 hardware type (1 for ethernet)
    - 16 protocol type (0x0800 for ipv4)
    - 8 hardware address length (6 for ethernet)
    - 8 protocol address length (4 for ipv4)
    - 16 for opcode (1-request, 2=reply)
    - 48 sender hardware address
    - 32 sender protocol address
    - 48 target hardware address
    - 32 target protocol address
* Dynamic Host Configuration Protocol (DHCP)(app layer)
  + own IP
  + own net mask
  + IP for local DNS name server
  + IP for first hop router (Default)
  + One or more local DHCP servers maintain required information
  + client broadcasts a DHCP discovery message
  + One or more DHCP servers responds with a DHCP offer message
  + client broadcast a DHCP request message
  + Selected DHCP server responds with an ACK