CSE 461: Midterm Review

Spring 2023

Administrivia

- Midterm on Monday 5/1
- HW3 due Monday 5/8
- Project 2 due Wednesday 5/10

Midterm Review Section

- Today: Go over selected practices together
  - Goal: Try applying the knowledge to solve problems, help you find blind spots
  - This is an **incomplete** sampling of what you should prepare for midterm

- What **YOU** should do after this section and before the exam:
  - Go through the lecture slides

Practice 1

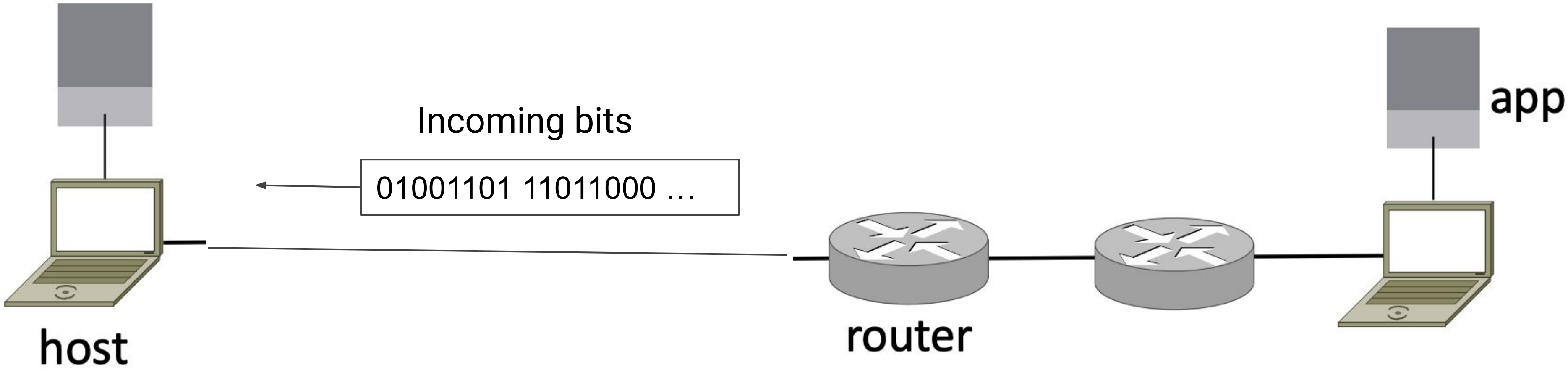
Layering

- Protocols and layers
- Encapsulation
- Demultiplexing

Practice Problems | Layering

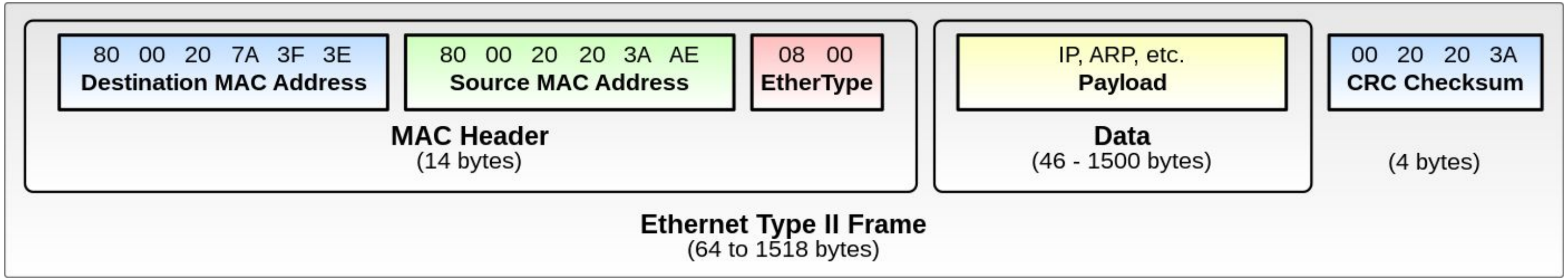
To extract application data from bits received from wire…

- Whose job it is?
- How?



Practice Problems | Layering

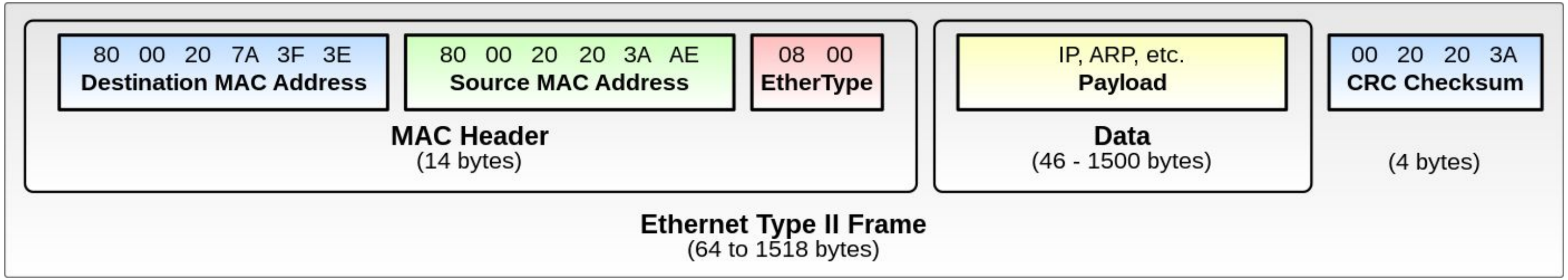
To extract application data from bits received from wire…
Step1: decode link layer (layer 2) header, extract the encapsulated layer 3 packet



Practice: which protocol are we using for the next layer?

Practice Problems | Layering

To extract application data from bits received from wire…
Step1: decode link layer (layer 2) header, extract the encapsulated layer 3 packet



Use demultiplexing identifiers, which is EtherType in Ethernet protocol:

EtherType values for some notable protocols[7]

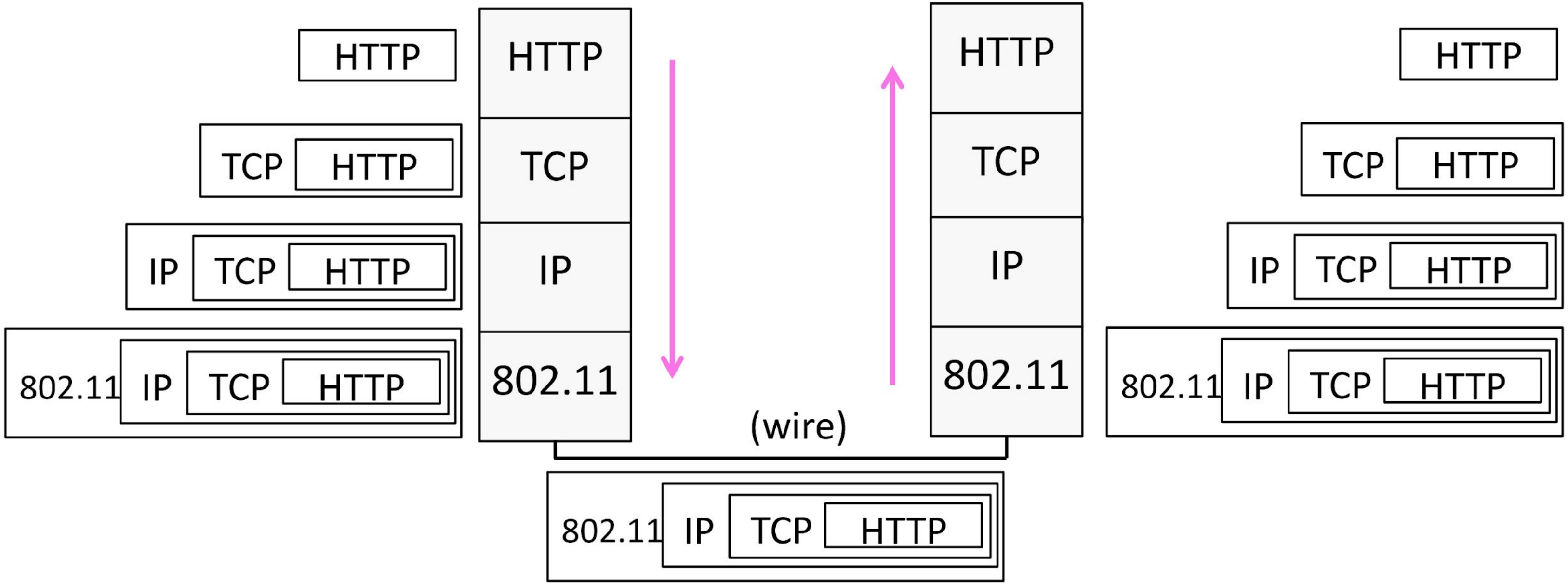
EtherType (hexadecimal)	Protocol
0x0800	Internet Protocol version 4 (IPv4)
0x0806	Address Resolution Protocol (ARP)

Practice Problems | Layering

To extract application data from bits received from wire…
Step1: decode link layer (layer 2) header, extract the encapsulated layer 3 packet
Step2: decode network layer (layer 3) header, extract the encapsulated layer 4 packet
Step3: decode transport layer (layer 4) header, extract the encapsulated payload

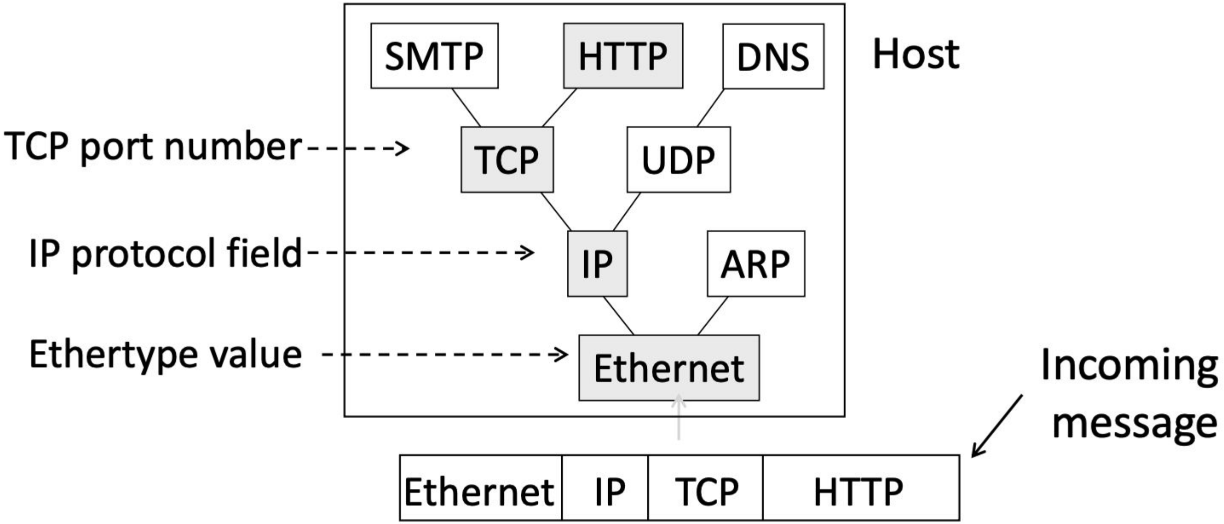
Done! Hand over the payload to application.

Encapsulation/Decapsulation



Find out the next layer: demultiplexing

- Done with demultiplexing identifiers in the headers



Practice 2

Bandwidth allocation

- Max-min fairness
- AIMD

Practice Problems | Bandwidth Allocation

Consider a network with four flows A, B, C, and D, all of which pass through the same 10 Mbps bandwidth link. At this moment in time,
flow A has a demand of 1 Mbps,
flow B has a demand of 2 Mbps,
flow C has a demand of 10 Mbps,
flow D has an infinite demand.
Assuming the link has no other flows, what would be the bandwidth allocation regarding max-min fairness?

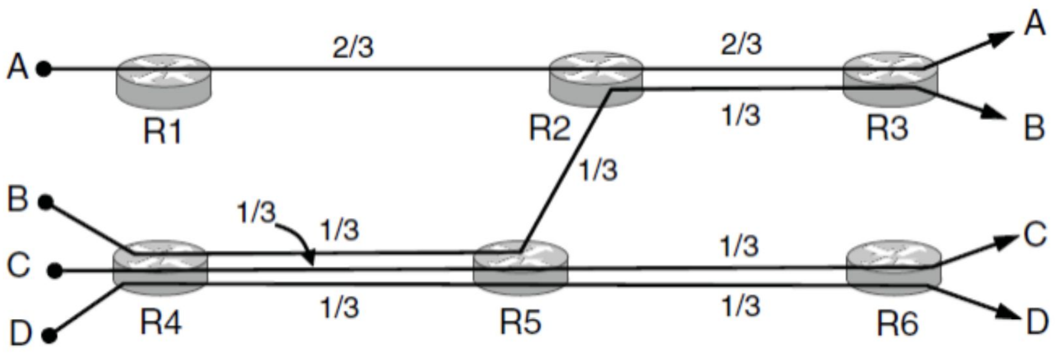
Practice Problems | Bandwidth Allocation

Consider a network with four flows A, B, C, and D, all of which pass through the same 10 Mbps bandwidth link. At this moment in time,
flow A has a demand of 1 Mbps,
flow B has a demand of 2 Mbps,
flow C has a demand of 10 Mbps,
flow D has an infinite demand.
Assuming the link has no other flows, what would be the bandwidth allocation regarding max-min fairness?

Flow A: 1 Mbps
Flow B: 2 Mbps
Flow C: 3.5 Mbps
Flow D: 3.5 Mbps

Max-Min Fair Allocation

1. Start with all flows at rate 0
2. Increase the flows until there is a new bottleneck in the network
3. Hold fixed the rate of the flows that are bottlenecked
4. Go to step 2 for any remaining flows



Practice Problems | AIMD

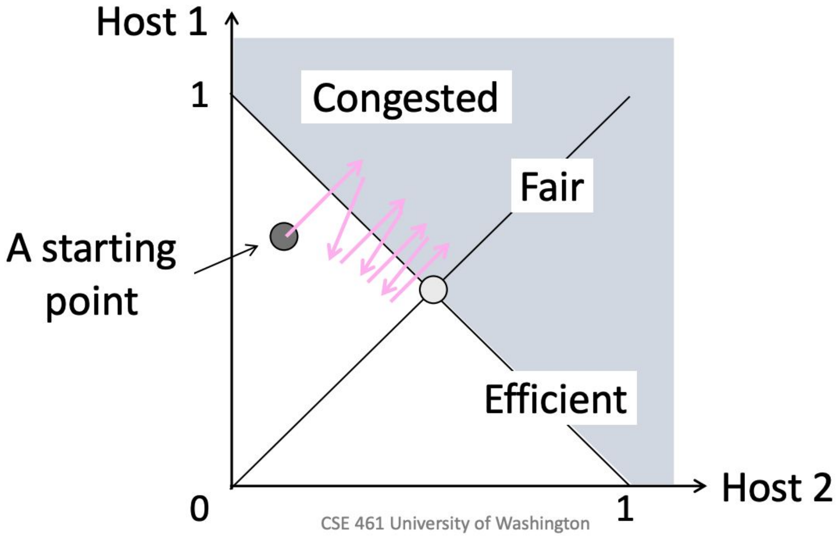
Consider a 10 Mbps capacity link shared by two flows A and B. Flow A has a bandwidth of 3 Mbps while flow B has an infinite bandwidth. For each RTT, each flow will increase the rate at which it is sending data by 1 Mbps if the path is not congested, while they will decrease the rate at which they send data by a factor of 2 (ie, cut in half) if the path is congested. Assuming that the flow A starts by sending data at a rate of 2 Mbps and flow B at a rate of 20 Mbps, what will be the rate at which each flow sends data after 1 RTT, 2 RTT, and 3 RTT have elapsed? You may assume that no other links in flow A or B are congested. Answer in Mbps.

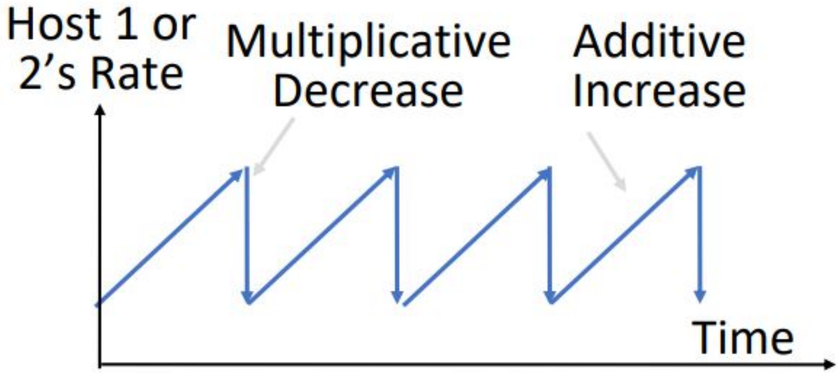
Practice Problems | AIMD

Consider a 10 Mbps capacity link shared by two flows A and B. Flow A has a bandwidth of 3 Mbps while flow B has an infinite bandwidth. For each RTT, each flow will increase the rate at which it is sending data by 1 Mbps if the path is not congested, while they will decrease the rate at which they send data by a factor of 2 (ie, cut in half) if the path is congested. Assuming that the flow A starts by sending data at a rate of 2 Mbps and flow B at a rate of 20 Mbps, what will be the rate at which each flow sends data after 1 RTT, 2 RTT, and 3 RTT have elapsed? You may assume that no other links in flow A or B are congested. Answer in Mbps.

After 1 RTT: Flow A = 1 Mbps, Flow B = 10 Mbps
After 2 RTT: Flow A = 0.5 Mbps, Flow B = 5 Mbps
After 3 RTT: Flow A = 1.5 Mbps, Flow B = 6 Mbps

AIMD - Additive Increase Multiplicative Decrease





Network Layer

- Network Service Models
- IP Address and Forwarding
- DHCP, ARP, ICMP
- NAT, IPv6
- Routing Algorithms
- BGP

Motivation

- What does the network layer do?
  - Connect different networks (send packets over multiple networks)

- Why do we need the network layer?
  - Switches don't scale to large networks
  - Switches don't work across more than one link layer technology
  - Switches don't give much traffic control

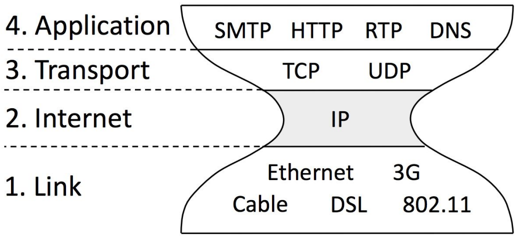
Network Service Models

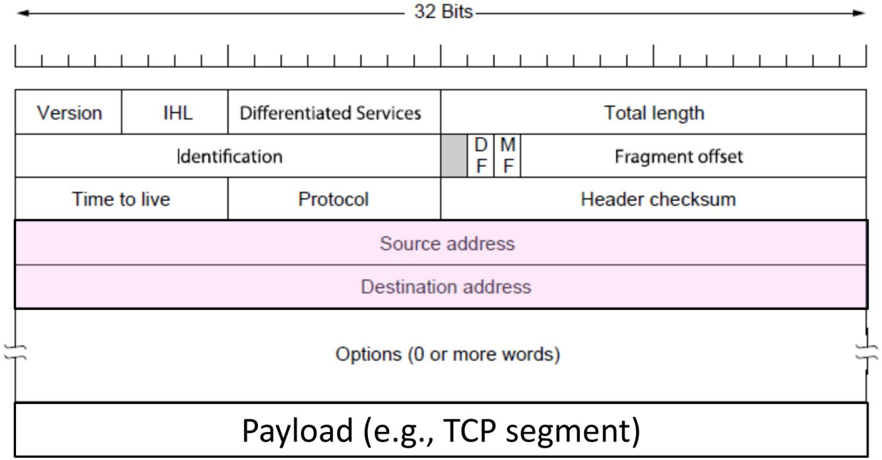
Datagrams	Virtual Circuits
Connectionless service	Connection-oriented service
No setup	Connection setup required
Packets contain destination address	Packets contain label for circuit
Routers looks up address in its forwarding table to determine next hop	Router looks up circuit in forwarding table to determine next hop
Example: IP	Example: MPLS

Internetworking - IP

- How do we connect different networks together?

- **IP - Internet Protocol**
- Lowest Common Denominator
  - Asks little of lower-layer networks
  - Gives little as a higher layer service

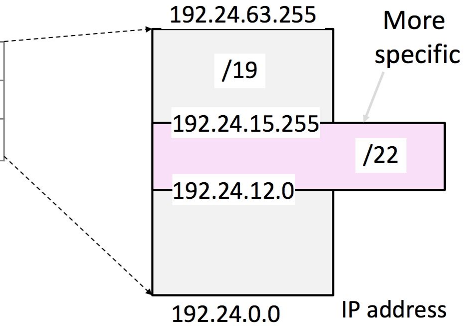




IP Addresses Prefix and Forwarding

- IP prefix   a.b.c.d/**L**
  - Represents addresses that have the same first L bits
  - e.g. 128.13.0.0/16 -> all 65536 addresses between 128.13.0.0 to 128.13.255.255
  - e.g. 18.31.0.0/32 -> 18.31.0.0 (only one address)
- **Longest Matching Prefix**
  - Find the longest prefix that contains the destination address, i.e., the most specific entry

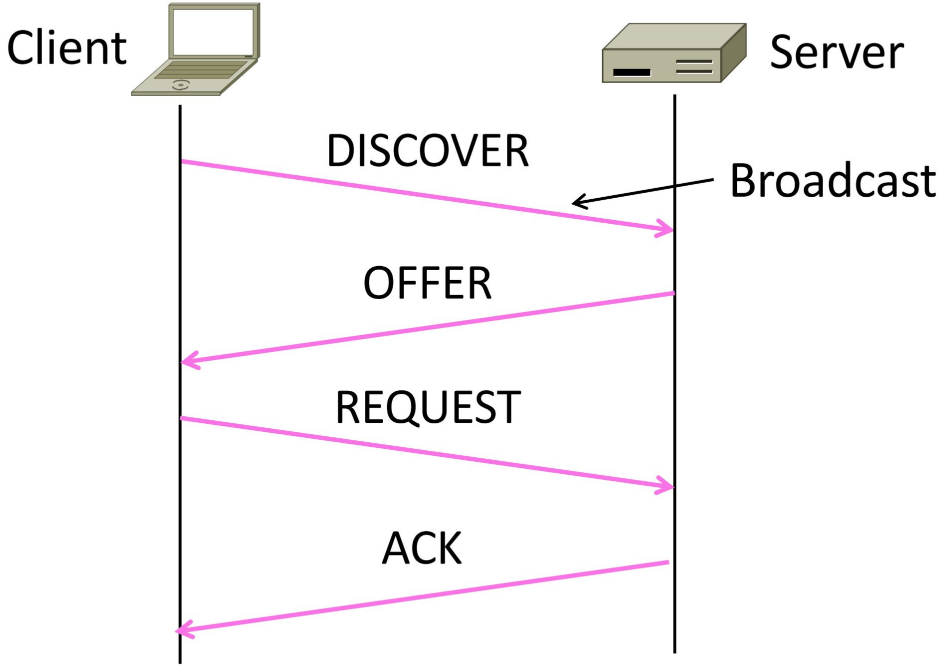
Prefix	Next Hop
192.24.0.0/19	D
192.24.12.0/22	B



192.24.6.0 → D
192.24.14.32 → B
192.24.54.0 → D

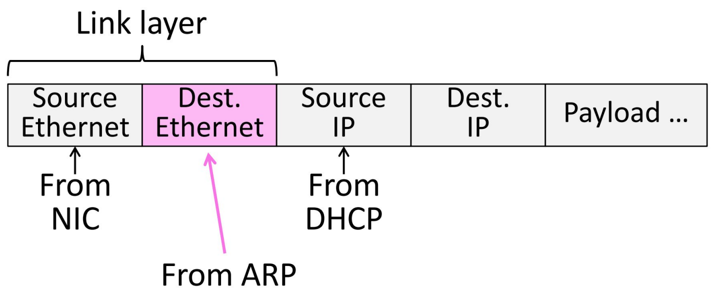
DHCP - Dynamic Host Configuration Protocol

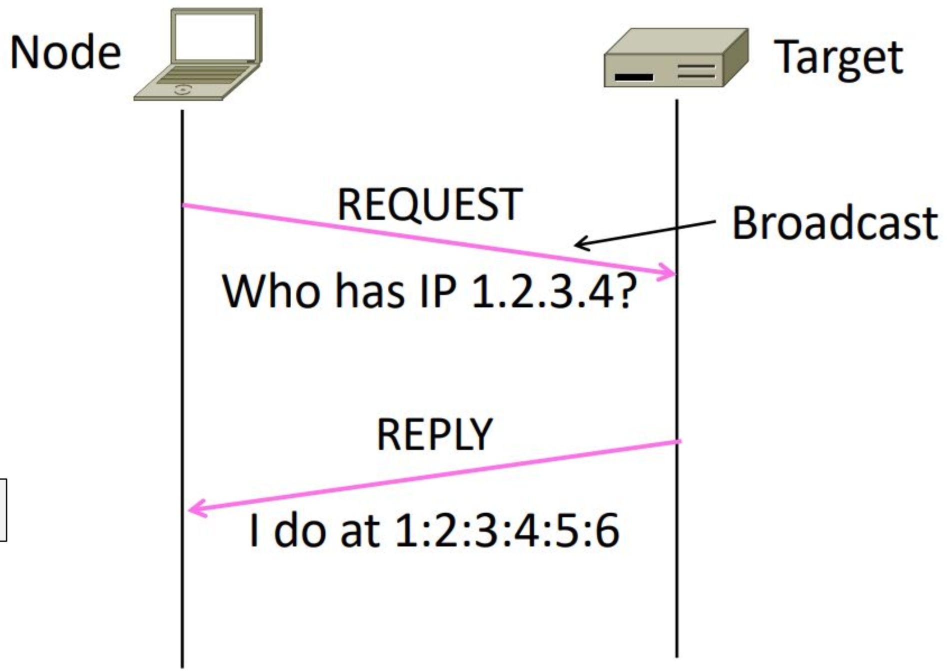
- Bootstrapping problem
- Leases IP address to nodes
- UDP
- Also setup other parameters:
  - DNS server
  - IP address of local router
  - Network prefix



ARP - Address Resolution Protocol

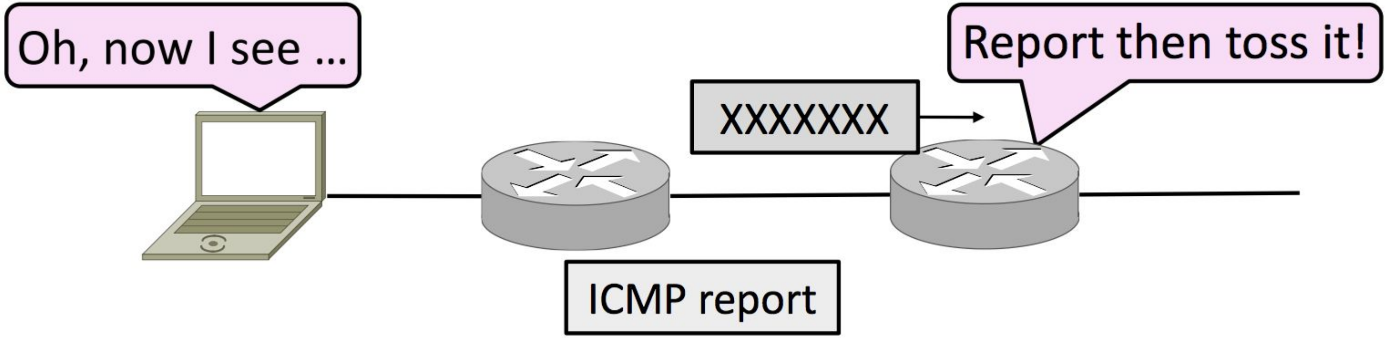
- MAC is needed to send a frame over the local link
- ARP to map an IP to MAC
- Sits on top of link layer





ICMP - Internet Control Message Protocol

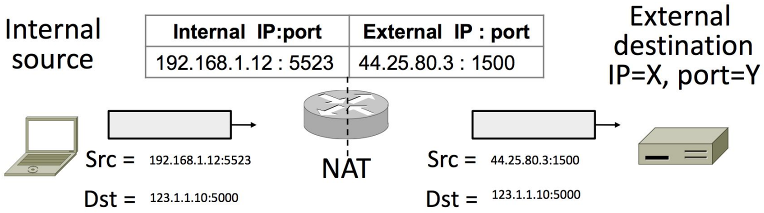
- Provides error reporting and testing
- Companion protocol to IP
- Traceroute, Ping



NAT - Network Address Translation

- One solution to **IPv4 address exhaustion**
- Map many private IP to one public IP, with different port number
- Pros: useful functionality (firewall), easy to deploy, etc.
- Cons: Connectivity has been broken!
- Many other cons...

What host thinks	What ISP thinks
Internal IP:port	**External IP : port**
192.168.1.12 : 5523	44.25.80.3 : 1500
192.168.1.13 : 1234	44.25.80.3 : 1501
192.168.2.20 : 1234	44.25.80.3 : 1502



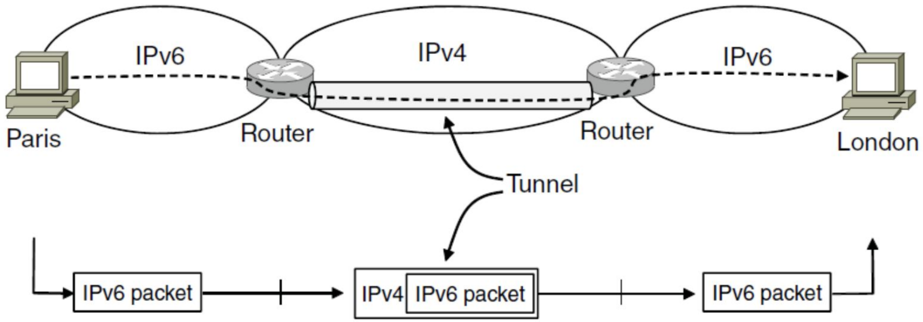
IPv6

- A much better solution to IPv4 address exhaustion
- Uses 128-bit addresses, with lots of other changes
- IPv6 version protocols: NDP -> ARP, SLAAC -> DHCP
- Problem: being incompatible with IPv4. Solution: Tunnelling

What's my IP

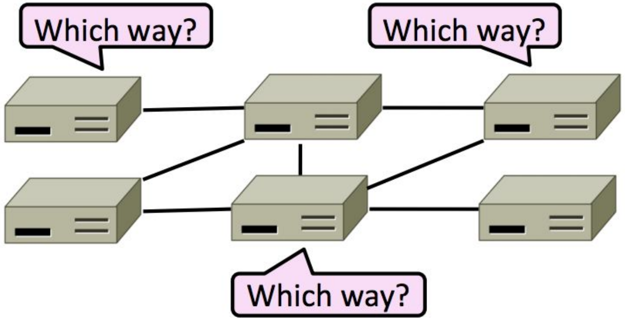
2601:602:8b00:5f0:30b3:2d19:3fe:db9e

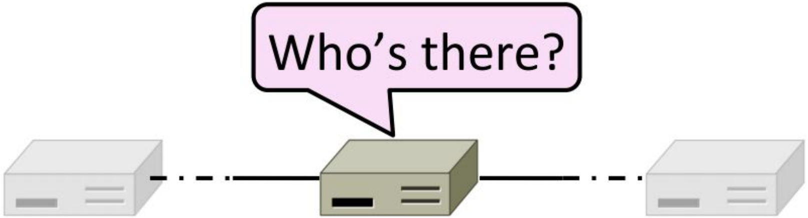
Your public IP address



Routing

- The process of deciding in which direction to send traffic
- Delivery models: unicast, broadcast, multicast, anycast
- Goals: correctness, efficient paths, fair paths, fast convergence, scalability
- Rules: decentralized, distributed setting

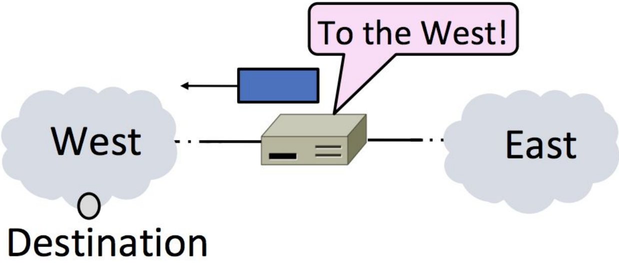




Techniques to Scale Routing

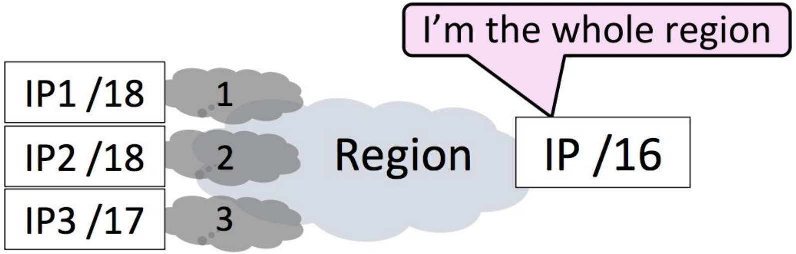
Hierarchical Routing

- Route first to the region, then to the IP prefix within the region



IP Prefix Aggregation and Subnets

- Adjusting the size of IP prefixes
  - Internally split one large prefix
  - Externally join multiple IP prefixes



Best Path Routing

Distance Vector Routing

Each node maintains a vector of distances and next hops to all destinations.

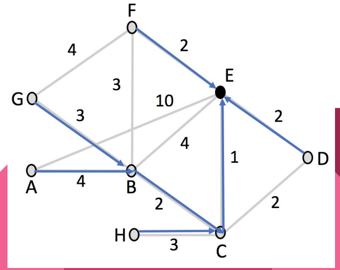
Problems: Count-to-infinity scenario when removing links.

Link State Routing (widely used)

Phase 1. **Topology Dissemination**: Each node floods neighboring links. They learn full topology.

Phase 2. **Route Computation**: Each node runs Dijkstra algorithm (or equivalent)

Algorithm details available in lecture slides

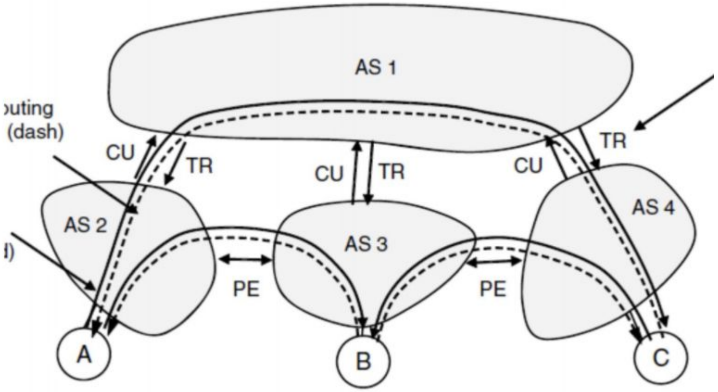


BGP - Border Gateway Protocol

- Internet-wide routing between ISPs (ASes)
  - Each has their own policy decisions
- Peer and Transit (Customer) relationship
- Border routers of ISPs announce BGP routes only to other parties who may use those paths.
- Border routers of ISPs select the best path of the ones they hear in any, non-shortest way

BGP example

- Transit (Provider & Customer)
  - Provider announces everything it can reach to its customer
    - AS1 to AS2: you can send packet to AS4 through me
  - Customer only announces its customers to provider
    - AS2 to AS1: you can send packet to A through me
- Peer (ISP 1 & ISP 2)
  - ISP 1 only announces its customer to ISP 2 and vice versa
    - AS2 to AS3: you can send packet to A through me



Transport Layer

- Service Models
- TCP vs UDP
- TCP Connections
- Flow Control and Sliding Window
- TCP Congestion Control
- Newer TCP Implementations

Service Models

- Transport Layer Services
  - Datagrams (UDP): Unreliable Messages
  - Streams (TCP): Reliable Bytestreams

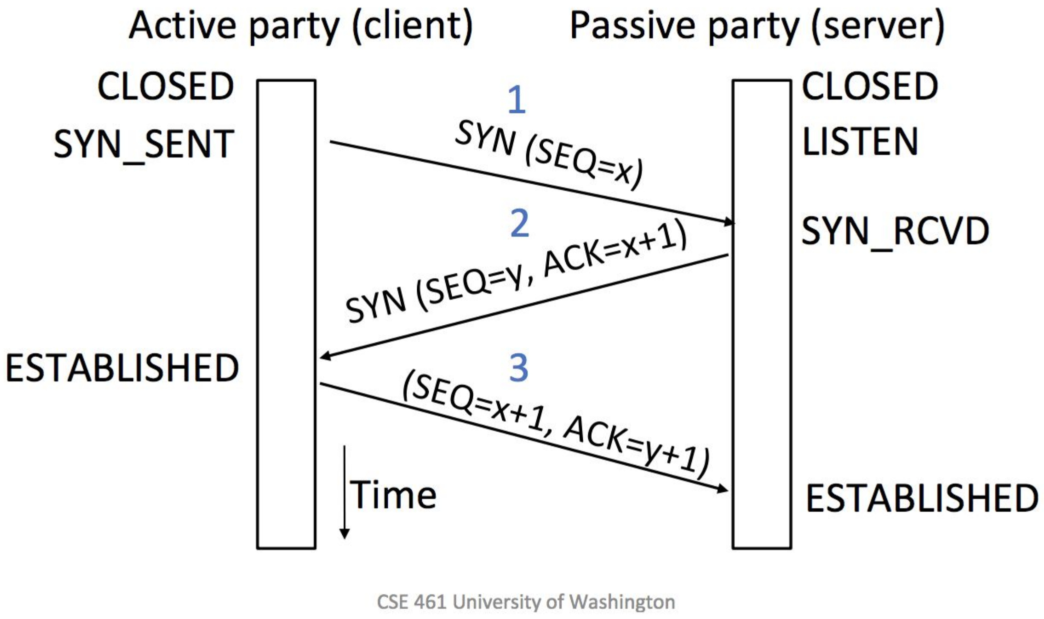
- Socket API: simple abstraction to use the network
  - Port: Identify different applications / application layer protocols on a host

TCP vs UDP

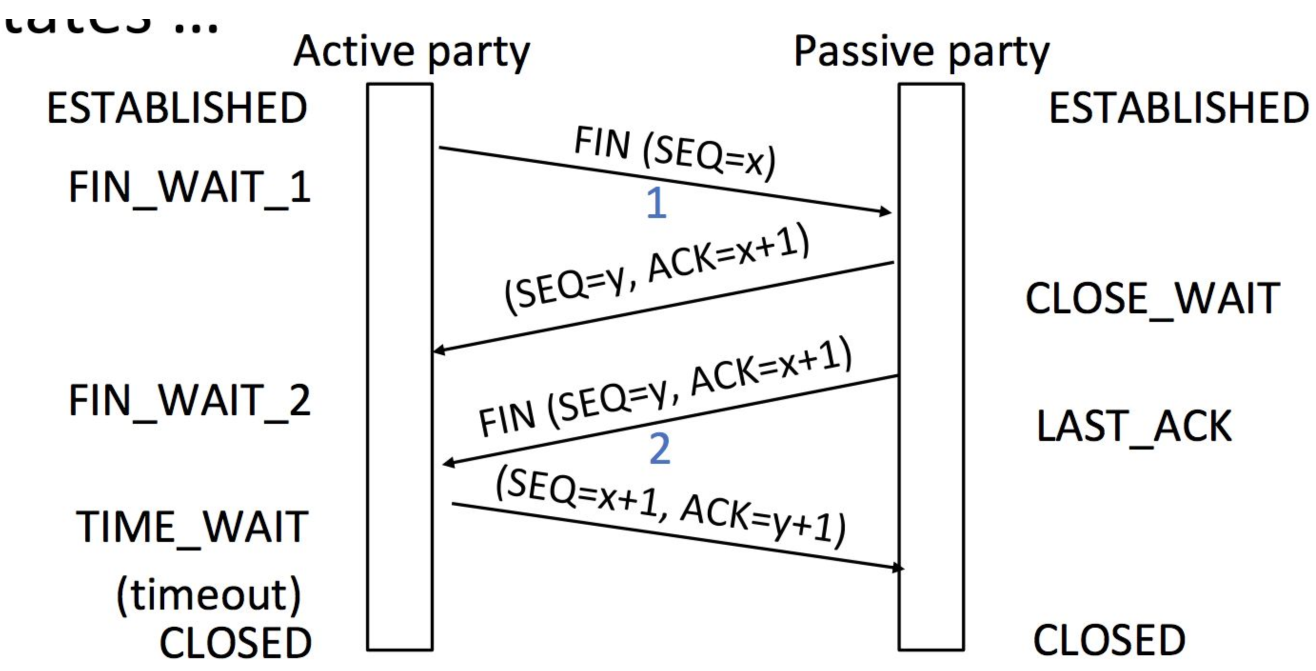
TCP (Streams)	UDP (Datagrams)
Connections	Datagrams
Bytes are delivered once, reliably, and in order	Messages may be lost, reordered, duplicated
Arbitrary length content	Limited message size
Flow control matches sender to receiver	Can send regardless of receiver state
Congestion control matches sender to network	Can send regardless of network state

TCP Connection Establishment

Three-way handshake

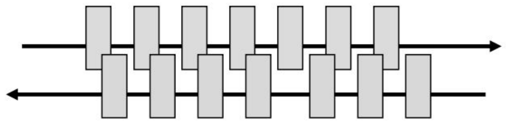


TCP Connection Release

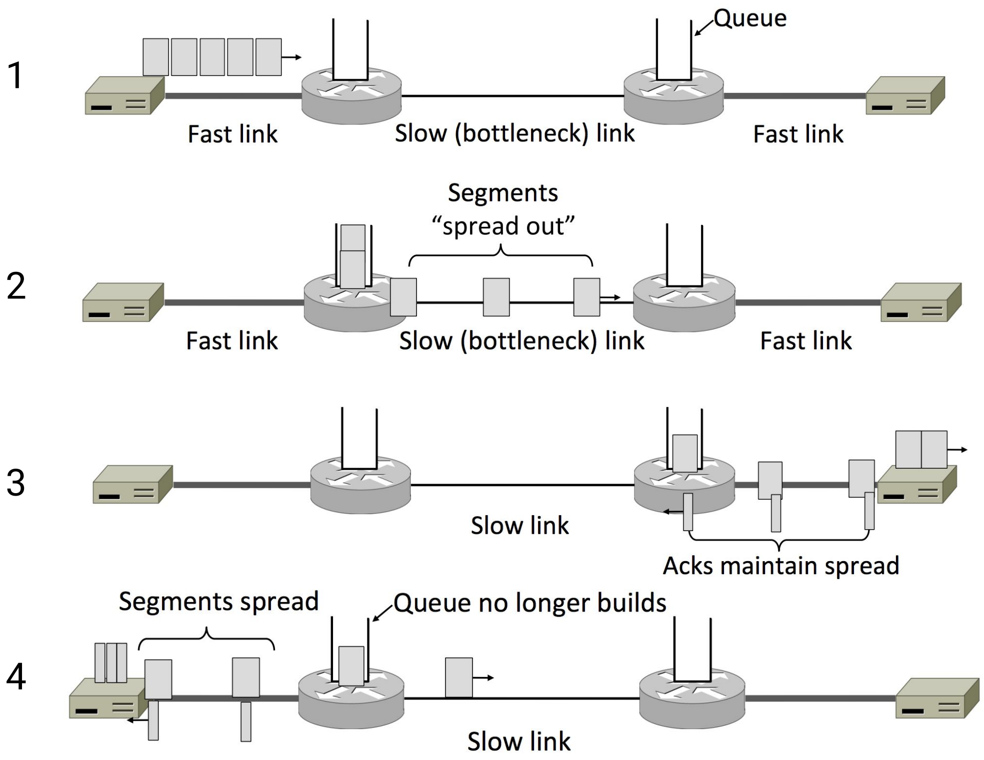


Flow Control - Sliding Window Protocol

- Instead of stop-and-wait, sends W packets per 1 RTT
  - To fill network path, W = B * RTT / packet_size
- Receiver sends ACK upon receiving packets
  - Go-Back-N (similar to project 1 stage b): not efficient
  - **Selective Repeat**
    - Receiver passes data to app in order, and buffers out-of-order segments to reduce retransmissions
    - ACK conveys highest in-order segment
      - As well as hints about out-of-order segments
- **Selective Retransmission** on sender's side



Flow Control - ACK Clock



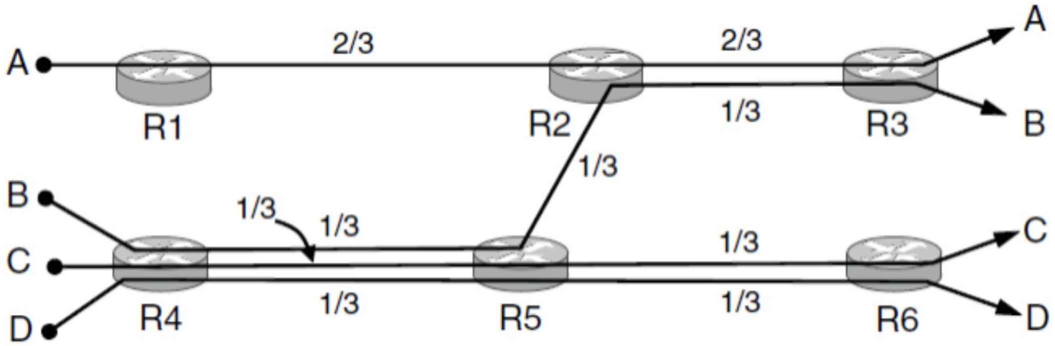
Flow Control - Sliding Window Protocol (2)

- Flow control on receiver's side
  - In order to avoid loss caused by user application not calling recv(), receiver tells sender its available buffer space (WIN)
  - Sender uses lower of the WIN and W as the effective window size
- How to set a **timeout** for retransmission on sender's side?
  - Adaptively determine timeout value based on smoothed estimate of RTT

$$SRTT_{N+1} = 0.9*SRTT_N + 0.1*RTT_{N+1}$$
$$Svar_{N+1} = 0.9*Svar_N + 0.1*|RTT_{N+1} - SRTT_{N+1}|$$
$$TCP\ Timeout_N = SRTT_N + 4*Svar_N$$

Max-Min Fair Allocation

1. Start with all flows at rate 0
2. Increase the flows until there is a new bottleneck in the network
3. Hold fixed the rate of the flows that are bottlenecked
4. Go to step 2 for any remaining flows

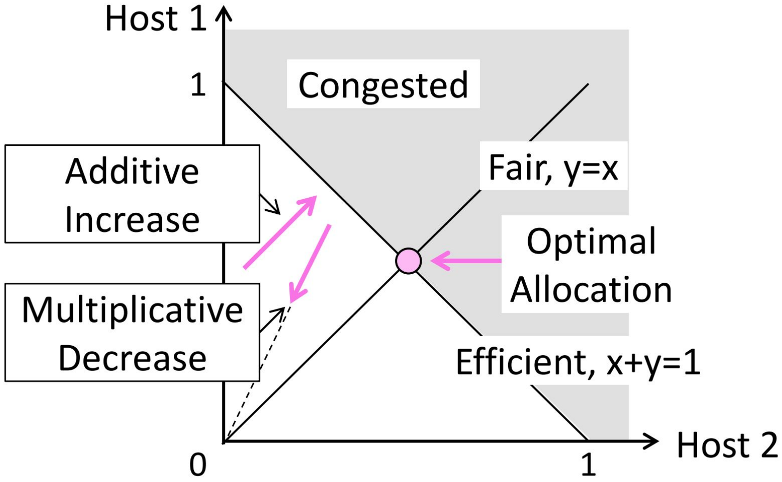


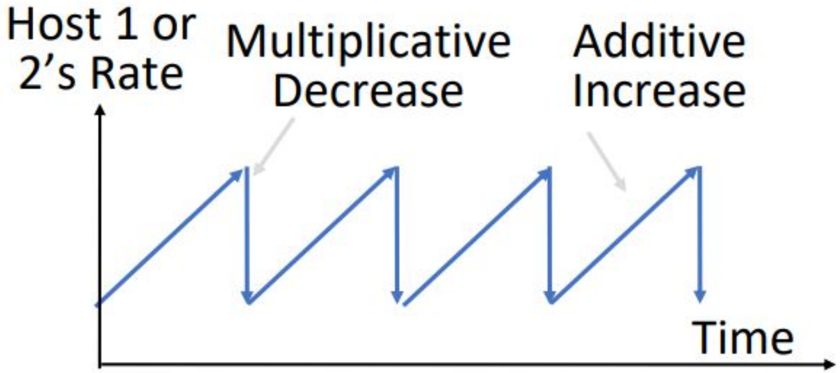
TCP Bandwidth Allocation

- Closed loop: use feedback to adjust rates
  - NOT open loop: reserve bandwidth before use
- Host driven: host sets/enforces allocations
  - NOT network driven
- Window based
  - NOT rate based
- Congestion signal
  - Packet loss, Packet delay, Router indication

AIMD!

AIMD - Additive Increase Multiplicative Decrease

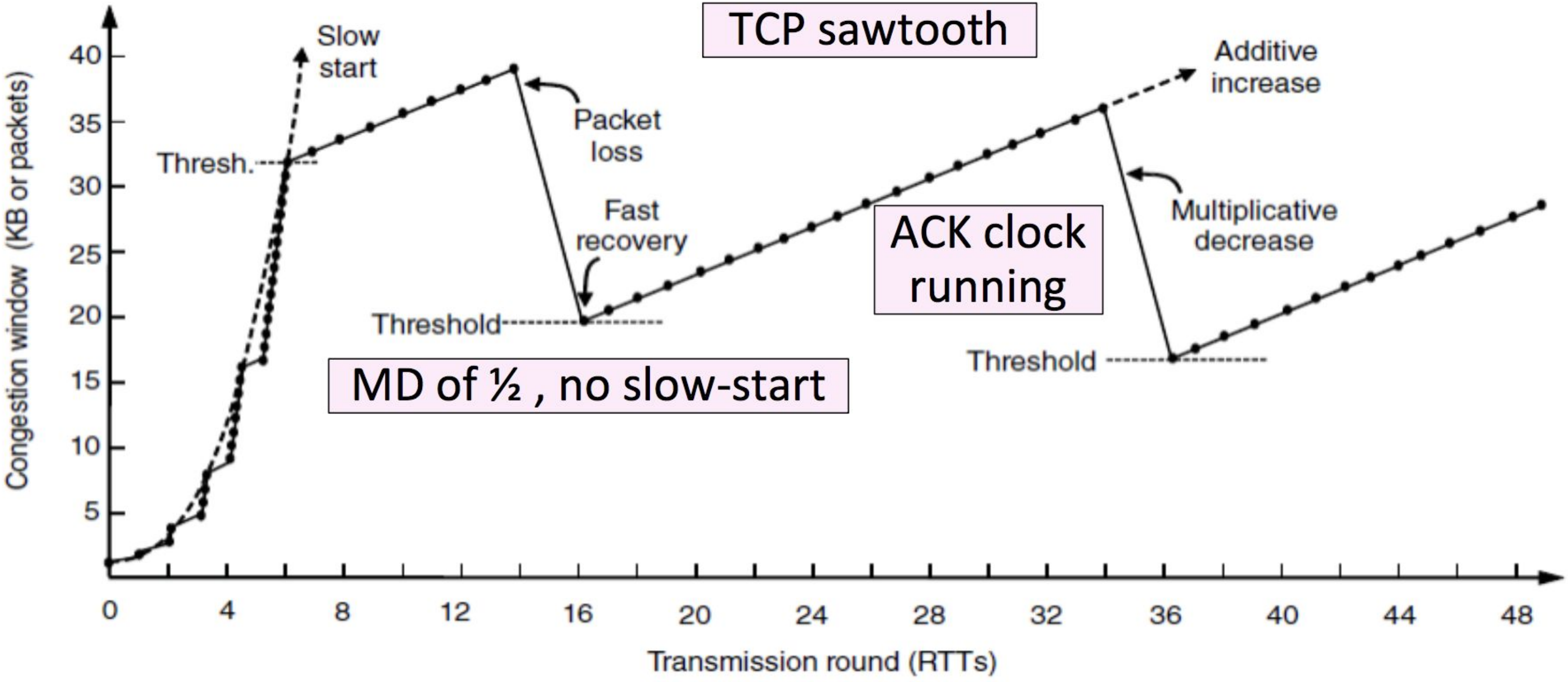




AIMD

- **Slow-Start** (used in AI)
  - Double cwnd until packet timeout
  - Restart and double until cwnd/2, then AI
- **Fast-Retransmit** (used in MD)
  - Three duplicate ACKs = packet loss
  - Don’t have to wait for TIMEOUT
- **Fast-Recovery** (used in MD)
  - MD after fast-retransmit
  - Then pretend further duplicate ACKs are the expected ACKs

TCP Reno



TCP sawtooth
ACK clock running
MD of ½ , no slow-start
Slow start
Thresh.
Packet loss
Fast recovery
Threshold
Additive increase
Multiplicative decrease
Threshold
Congestion window (KB or packets)
Transmission round (RTTs)
40
35
30
25
20
15
10
5
0
4
8
12
16
20
24
28
32
36
40
44
48

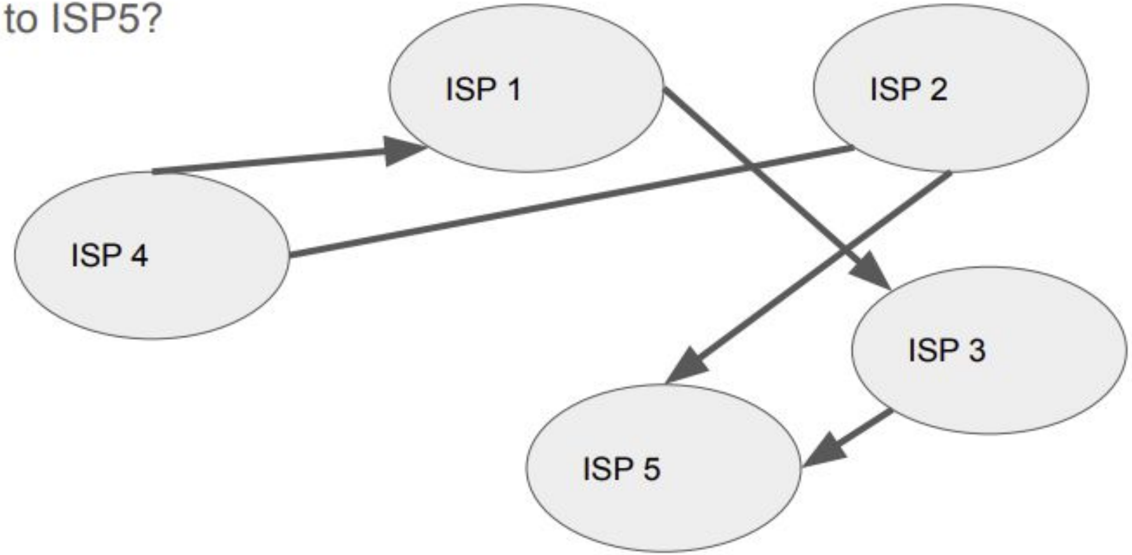
Network-Side Congestion Control

- Explicit Congestion Notification (**ECN**)
  - Router detects the onset of congestion via its queue. When congested, it marks affected packets in their IP headers
  - Marked packets arrive at receiver; treated as loss. TCP receiver reliably informs TCP sender of the congestion

Practice Problems | BGP

Customer → Provider

ISP4 to ISP5?

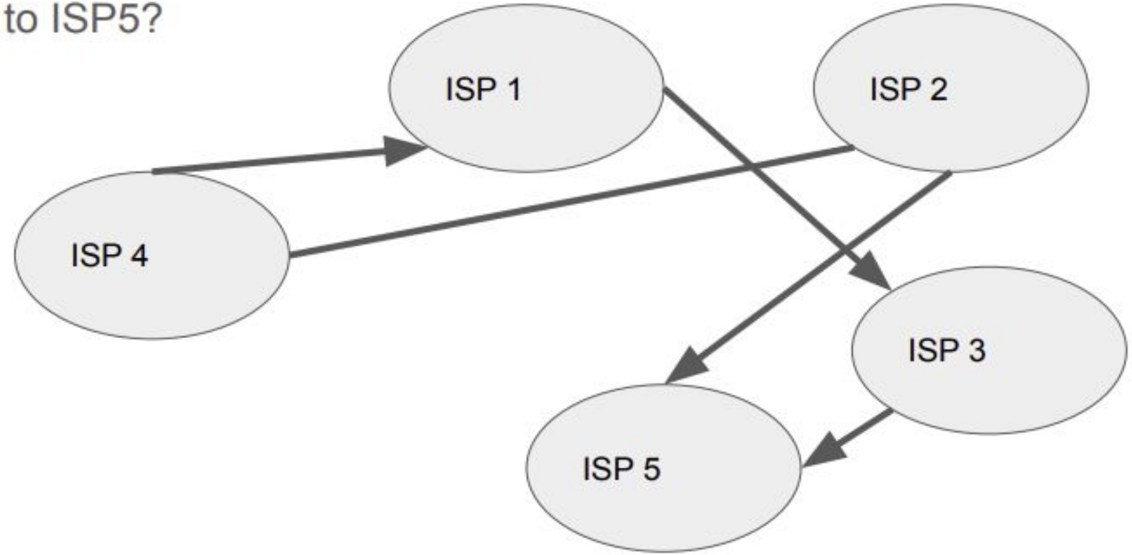


Practice Problems | BGP

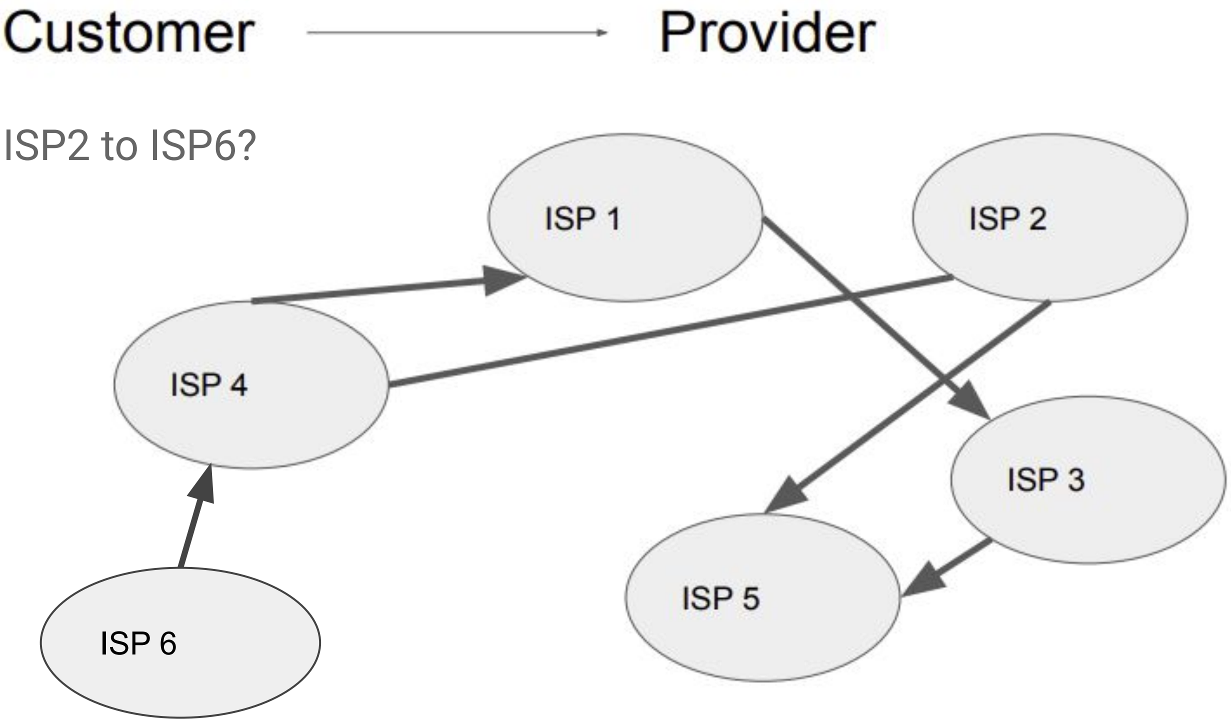
Customer → Provider

ISP4 - ISP1 - ISP3 - ISP5

ISP4 to ISP5?



Practice Problems | BGP



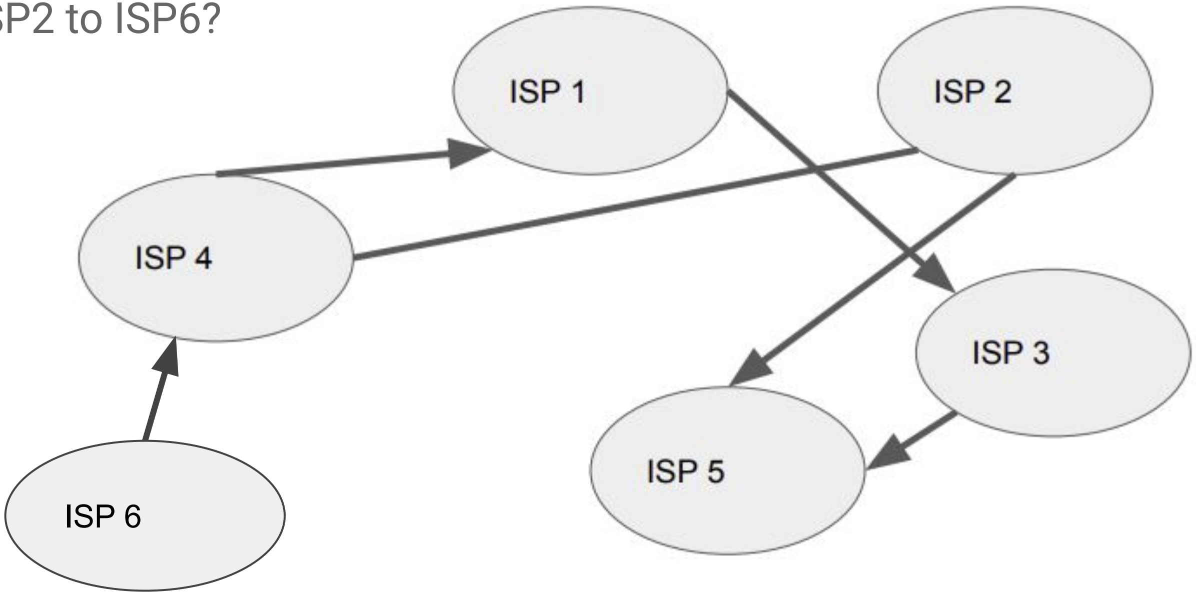
ISP2 to ISP6?

Practice Problems | BGP

Customer → Provider

ISP2 - ISP4 - ISP6

ISP2 to ISP6?



Practice Problems | Subnetting

Suppose you are given the prefix 192.168.0.0/16. You are asked to split the prefix into exactly 4 equal subnets, labelled 1-4. Give the prefix of each subnet. Under which subnet would the address 192.168.151.12 fall?

Practice Problems | Subnetting

Suppose you are given the prefix 192.168.0.0/16. You are asked to split the prefix into exactly 4 equal subnets, labelled 1-4. Give the prefix of each subnet. Under which subnet would the address 192.168.151.12 fall?

To split the prefix into 4 equal subnets, we'll need to borrow 2 bits from the host portion of the address to uniquely identify 4 different subnets with the same prefix. Hence, the first subnet will fix its 17th and 18th network address bit (from the left) to be 00, second will be 01, third will be 10, fourth will be 11:

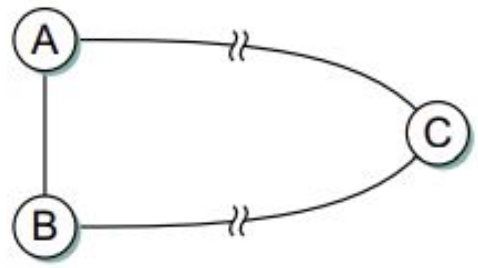
Subnet 1: 192.168.0.0/18
Subnet 2: 192.168.64.0/18
Subnet 3: 192.168.128.0/18
Subnet 4: 192.168.192.0/18

Looking at the different subnets, 192.168.151.12 would exist under subnet 3

Practice Problems | Link-State Routing

Suppose that nodes in the network shown in the figure below participate in link-state routing, and C receives contradictory LSPs: One from A arrives claiming the A–B link is down, but one from B arrives claiming the A–B link is up.

a) How could this happen?
b) What should C do? What can it expect?



Practice Problems | Link-State Routing

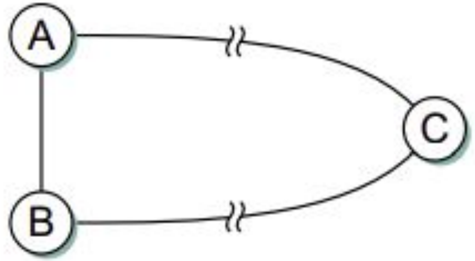
Suppose that nodes in the network shown in the figure below participate in link-state routing, and C receives contradictory LSPs: One from A arrives claiming the A–B link is down, but one from B arrives claiming the A–B link is up.

a) How could this happen?

This could happen if B sent an LSP packet just before the A-B link was down, followed by A sending an LSP packet after the link was down (or vice-versa)

b) What should C do? What can it expect?

In Link-State routing, flooding takes place periodically and new routes are constantly being calculated. Hence, eventually the old LSP would be updated by its sender with a newer version, and reports from the two sides would come to an agreement again.

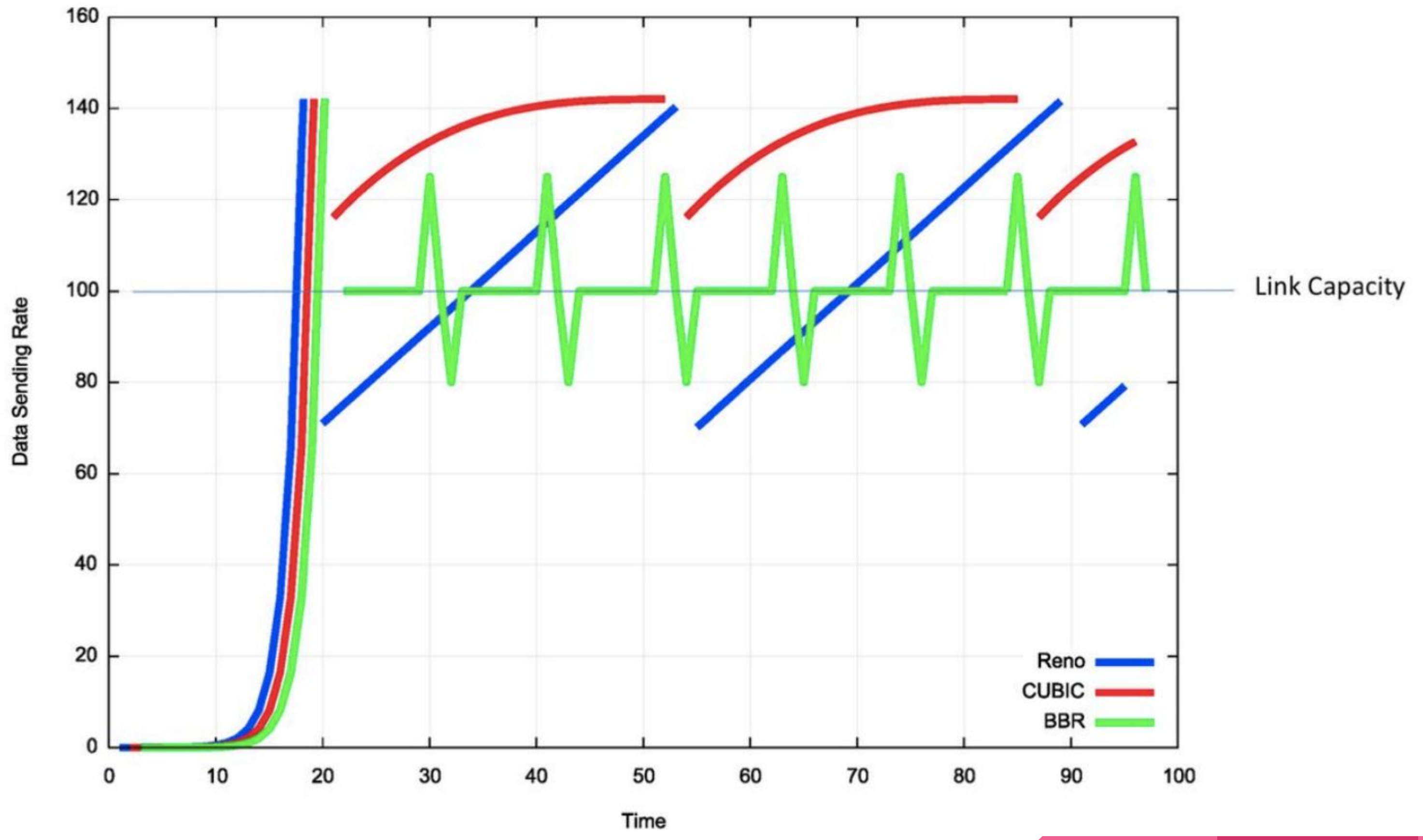


TCP CUBIC

- Problem with standard TCP?
  - Flows with lower RTT’s “grow” faster than those with higher RTTs
  - Flows grow too “slowly” (linearly) after congestion

TCP BBR

- **Bufferbloat Problem**
  - performance can decrease when buffer size is increased
- **Model based** instead of loss based
  - Measure RTT, latency, bottleneck bandwidth
  - Use this to predict window size


160
140
120
100
80
60
40
20
0
Data Sending Rate
0
10
20
30
40
50
60
70
80
90
100
Time
Link Capacity
Reno
CUBIC
BBR