ELEC / COMP 177 – Fall 2012

Computer Networking → Exam Review

Some slides from Kurose and Ross, *Computer Networking*, 5th Edition

Final Exam

- **Tuesday, December 11th – 8am-11am**
- Short answer format
- Comprehensive
  - No paper resources (books, notes, …)
  - No electronic resources (computer, phone, …)
  - No human resources (except for you!)
- Time limited – 3 hours max
- Just you, your pencil, and paper
  - *You can bring a calculator if you want to convert from binary<->decimal*

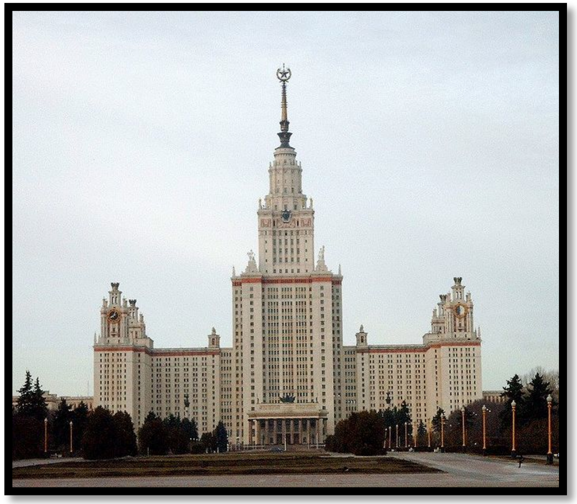
Final Exam

- What could be on the final exam?
  - Course lectures
  - Homework
  - Mid-term
  - Labs (but most of that is in *lab practical exam*)
- What won't be on the final exam?
  - Socket programming
  - Questions about obscure packet header details

Final Exam Review

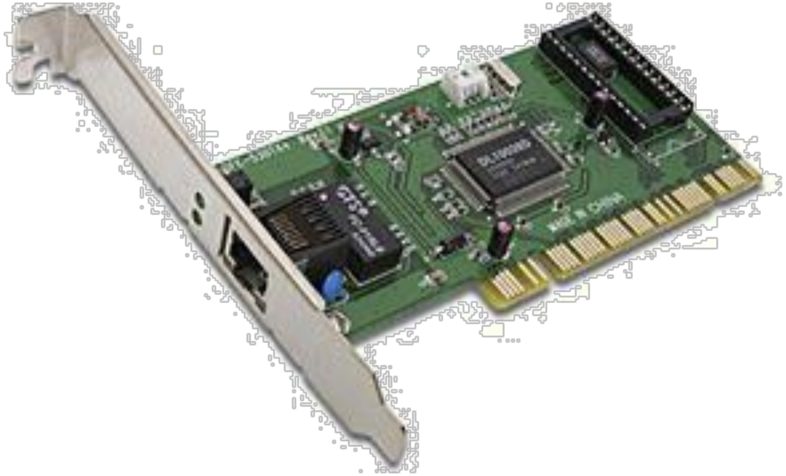
The Problem

- I take my laptop…
- Walk into CTC 115…
- Plug the Ethernet cable into the wall…
- Launch Safari…
- Load the webpage http://www.msu.ru/en/
- **What happens?**



Network Interface Card

- After plugging in cable, NIC detects link is active
- NIC *auto-negotiates* with device at other end (the switch)
  - Link speed?
  - Half duplex? Full duplex?
  - **Not** Ethernet frames (lower level)
- NIC notifies operating system (via device driver) that link is up



Operating System

- Notified by driver (part of OS) that network link is active
- OS attempts to configure network
  - Static configuration? *(User input)*
  - Dynamic configuration? *(DHCP)*
- Let's assume DHCP

DHCP

- DHCP packets are Ethernet←IP←UDP←DHCP
- **Four stages for a new host**
  - Discover
    - Client broadcasts to all hosts on subnet
  - Offer
    - DHCP server(s) respond directly to client with lease offer
  - Request
    - Client broadcasts acceptance of best offer to all hosts
  - Acknowledge
    - DHCP server responds directly to client with acceptance and other configuration information

Ethernet Switch

- Assume DHCP server is on LAN (connected to Ethernet switch)…
- **How does a switch learn the location of computers on the network? (what *field*)**
  - Source address in packet header
- **What is stored in the forwarding table?**
  - MAC address, output port
- **What does a switch do when it receives a packet with a broadcast destination address? A normal dest. MAC addr.?**

Post-DHCP

- What does the laptop OS now know?
  - IP address – 10.10.207.20
  - Netmask (for size of local subnet) – 255.255.254.0
  - IP address of default gateway – 10.10.207.254
  - IP address of DNS server - 10.10.4.2.226 and .227 (primary and secondary servers)
  - Other institution-specific information
    - Address book? Time server?

Ethernet

- **An aside – why do I need IP at all?**
- **Why can't we use Ethernet for global communication?**
  - Broadcasts to find location of computers – too much bandwidth to do worldwide
  - Loops – Ethernet uses spanning tree to prevent loops
    - Can't have a single "root" of the Internet!
  - **Address contains no information about location on network**
    - Would need to have a forwarding table with one entry for every PC on the Internet we want to communicate with
    - i.e. a single worldwide "phonebook" with no shortcuts!

Launch Web Browser

- User enters http://www.msu.ru/en/ into address bar and hits enter
- **What happens?**
  - Web browser checks cache. Present?
  - *Not present; need to fetch HTML page*

Web Browser -> Socket

- Web browser starts opening a TCP socket to www.msu.ru on port 80 to fetch /en/ via HTTP
- `getaddrinfo()` call into operating system
  - Arguments?
    - Hostname ("www.msu.ru")
    - Service type ("http")
    - Transport protocol (TCP, i.e. "SOCK_STREAM")
    - Optional configuration details
  - Function will produce an IP address and port number
    - **How?**

Resolving Host Name

- Laptop OS uses **DNS** to translate www.msu.ru into an IP address
  - This program is called the DNS *resolver*
- Check my DNS cache…
  - *Nope, empty!*
- DNS packets are Ethernet←IP←UDP←DNS
- Request is sent from laptop to Pacific DNS server
  - We learned the IP of this Pacific server via DHCP
  - Might be on local subnet (switches only)
  - Might be elsewhere (reach via **default gateway**)
    - *This is the case*

ARP

- **How do I know the MAC address of my default gateway?**
- Check my ARP table (cache)…
  - Nope, empty!
- ARP for it by IP address
  - ARP packets are Ethernet←ARP
  - Broadcast request to subnet: Who has this IP?
  - Default gateway should send reply directly to me
- **Cache** IP to MAC mapping in ARP table

ARP Cache after ARP

```
#> arp -a
...
...
(10.10.207.254) at 00:05:dc:53:3c:0a on eth0
...
```

↑ MAC address of gateway router

DNS @ Pacific (10.10.4.226)

DNS REQUEST

```
Ethernet II, Src: 7c:6d:62:8c:c2:df,
   Dst: 00:05:dc:53:3c:0a
Internet Protocol
   Src: 10.10.207.20, Dst:
   10.10.4.226
    Version: 4
    Header length: 20 bytes
    Total Length: 56
    Time to live: 255
    Protocol: UDP (17)
User Datagram Protocol
   Src Port: 54941, Dst Port: domain
   (53)
    Length: 36
    Checksum: 0xe83f
Domain Name System (query)
    Transaction ID: 0xad5b
    Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.msu.ru: type A, class IN
```

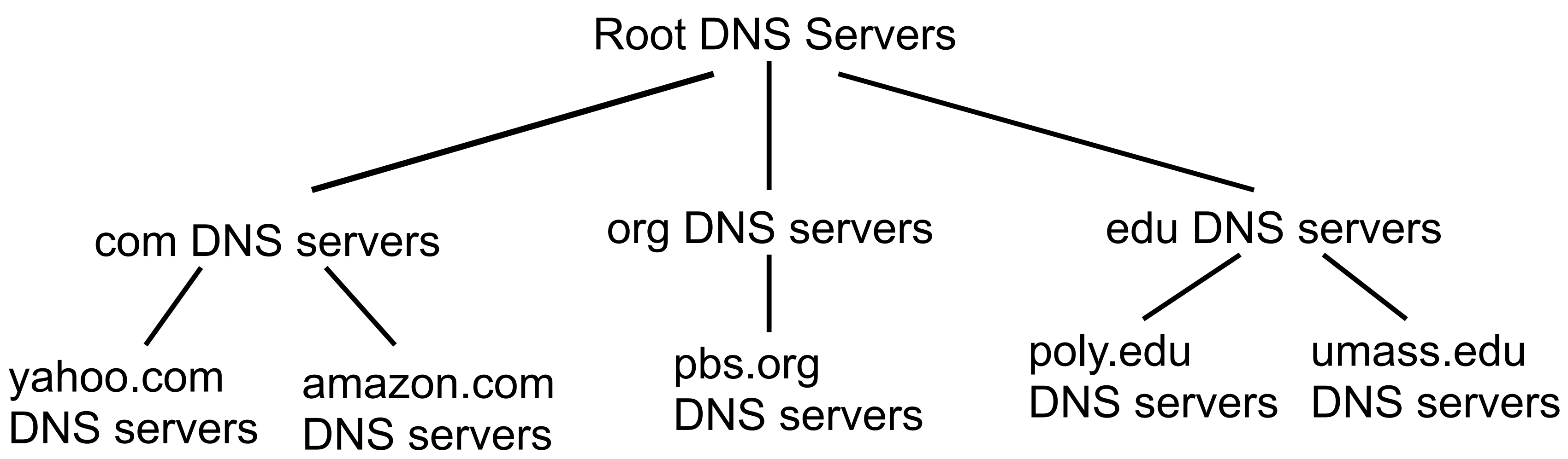
DNS RESPONSE

Waiting on response…

More DNS

- **How does the Pacific DNS server translate the hostname into IP address?**
- It might have the address cached from a prior lookup
  - *I hear www.facebook.com is a popular request…*
- Or it has to contact a root nameserver and iteratively/recursively traverse the hierarchy

DNS Hierarchy



- **What do the root nameservers store?**
  - Mapping between top-level domains (TLDs) (.com, .org, .edu, etc...) and servers for each TLD

More DNS

- Root nameservers – 13 total
  - *But in reality hundreds of physical machines on all continents*
  - *Some use **anycast** routing to find the closest host*
  - Labeled with hostnames A-M
    - *a.root-servers.net*
    - *b.root-servers.net*
    - *c.root-servers.net*
    - …

More DNS

- **How does the Pacific DNS server know the IP address of the 13 root nameservers?**
  - **It can't do a DNS lookup (circular dependency)**
- Root DNS IPs change **infrequently**
- When you obtain the DNS server software, it comes with a text file (named.cache) with the mapping
  - *As long as 1 mapping is still valid, that is sufficient to bring the server online and update the file*
  - Example: http://www.internic.net/zones/named.root

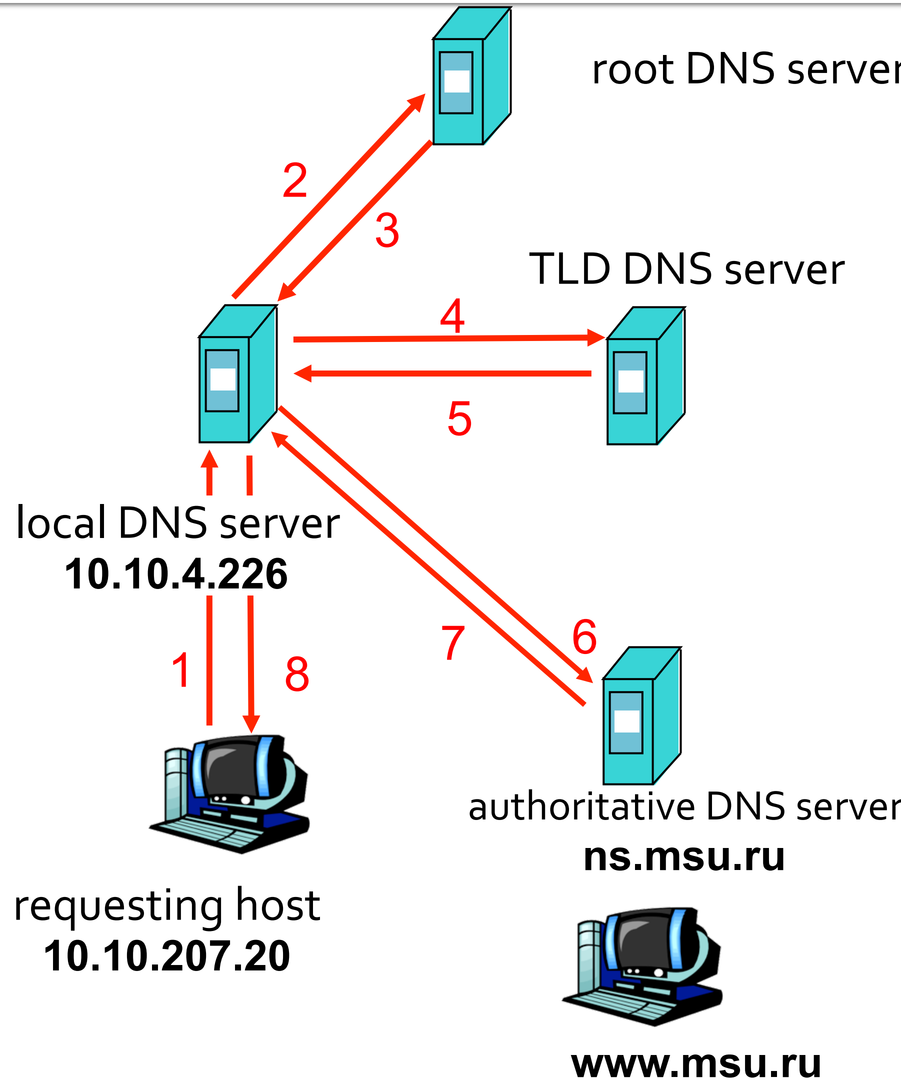
named.cache file

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       last update:    Jun 17, 2010
;       related version of root zone:   2010061700
;
; formerly NS.INTERNIC.NET
;
.                         3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.       3600000      A     198.41.0.4
A.ROOT-SERVERS.NET.       3600000      AAAA  2001:503:BA3E::2:30
;
; FORMERLY NS1.ISI.EDU
;
.                         3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.       3600000      A     192.228.79.201
;
; FORMERLY C.PSI.NET
;
.                         3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.       3600000      A     192.33.4.12
;
; FORMERLY TERP.UMD.EDU
;
```

etc… etc… etc…

More DNS

- Pacific DNS traverses the hierarchy
- Each partial result returns an IP address that gets us closer
- Final result!
  - Save in Pacific DNS cache
  - Send response to laptop



DNS Name Resolution

- Two modes – typical request mixes both!
- **Recursive**
  - The server you contact provides the final answer
  - *Behind the scenes, it may make several consecutive requests*
- **Iterative**
  - The server you contact directs you to a different server to get (closer to) the final answer

DNS and UDP

- DNS uses UDP by default
  - It *can* use TCP, but it's rare
  - **Isn't this unreliable?**
- Why use UDP
  - Faster (in three ways!)
    - No need to establish a connection (RTT/latency overhead)
    - Lower per-packet byte overhead in UDP header
    - Less packet processing by hosts
  - Reliability not needed
    - DNS will just re-request if no response received (2-5 seconds)

DNS @ Pacific (10.10.4.226)

EARLIER DNS REQUEST

```
Ethernet II, Src: 7c:6d:62:8c:c2:df,
   Dst: 00:05:dc:53:3c:0a
Internet Protocol
   Src: 10.10.207.20, Dst:
   10.10.4.226
    Version: 4
    Header length: 20 bytes
    Total Length: 56
    Time to live: 255
    Protocol: UDP (17)
User Datagram Protocol
   Src Port: 54941, Dst Port: domain
   (53)
    Length: 36
    Checksum: 0xe83f
Domain Name System (query)
    Transaction ID: 0xad5b
    Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.msu.ru: type A, class IN
```

DNS RESPONSE

```
Ethernet II, Src: 00:05:dc:53:3c:0a, Dst: 7c:6d:62:8c:c2:df
Internet Protocol
    Src: 10.10.4.226, Dst: 10.10.207.20
    Version: 4
    Header length: 20 bytes
    Total Length: 219
    Time to live: 252
    Protocol: UDP (17)
User Datagram Protocol
    Src Port: domain (53), Dst Port: 54941
    Length: 199
    Checksum: 0x33e0
Domain Name System (response)
    Transaction ID: 0xad5b
    Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 4
    Additional RRs: 3
    Queries
        www.msu.ru: type A, class IN
    Answers
        www.msu.ru: type A, class IN, addr 193.232.113.151
    Authoritative nameservers
        msu.ru: type NS, class IN, ns ns3.nic.fr
        msu.ru: type NS, class IN, ns ns1.orc.ru
        msu.ru: type NS, class IN, ns ns.msu.ru
        msu.ru: type NS, class IN, ns ns.msu.net
    Additional records
        ns.msu.net: type A, class IN, addr 212.16.0.1
        ns3.nic.fr: type A, class IN, addr 192.134.0.49
        ns3.nic.fr: type AAAA, class IN, addr
     2001:660:3006:1::1:1
```

Web Browser - Sockets

- Back to the laptop (which is still waiting for the DNS resolution to finish)
- *Waiting…*
- *Waiting…*
- Reply is received from Pacific DNS server with IP address – finally!
  - Let's **cache** it locally so we don't have to do this very often…
- Create the socket and try to connect

Web Browser – TCP

- Opening a TCP connection between laptop and destination web server
  - Source IP (of laptop) – learned from DHCP
  - Source port – **where does this come from?**
  - Destination IP (of server) – obtained from DNS
  - Destination port – **where does this come from?**
- TCP 3-way Handshake
  - Client sends **SYN** (“synchronize the sequence #s”)
  - Server responds with **SYN-ACK**
  - Client sends **ACK** (with optional first piece of data)

Web Server

- Assumptions:
  - A web server exists at MSU
  - The server has the same IP the laptop obtained by our DNS lookup
    - If the server moves, DNS must be updated!
  - The web server is listening on port 80
  - The firewall lets our communication through
    - It either specifically *permits* our traffic, or it does not specifically *deny* our traffic

Web Browser – HTTP

- Web browser sends data over TCP socket
- **What data?**

```
GET /en/ HTTP/1.1\r\n
Host: www.msu.ru\r\n
User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X
10_6_4; en-us) AppleWebKit/533.18.1 (KHTML, like
Gecko) Version/5.0.2 Safari/533.18.5\r\n
Accept: application/xml,application/xhtml+xml,text/
html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5\r\n
Accept-Language: en-us\r\n
Accept-Encoding: gzip, deflate\r\n
Cookie: <snipped...>
Connection: keep-alive\r\n
\r\n
```

Web Server – HTTP

- Web server returns data over TCP socket
- **What data?**

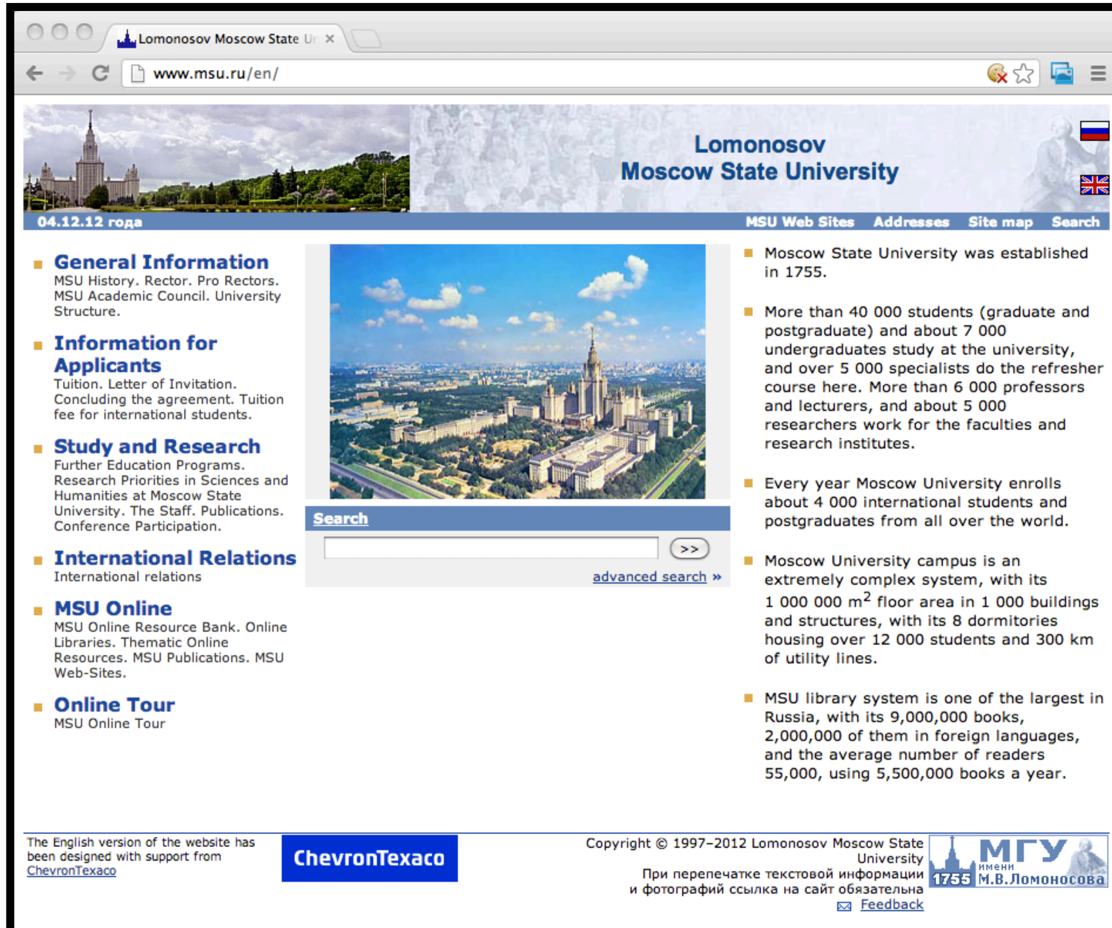
```
HTTP/1.1 200 OK\r\n
Date: Thu, 04 Nov 2010 16:32:35 GMT\r\n
Server: Apache/1.3.39\r\n
Cache-Control: max-age=1814400\r\n
Expires: Thu, 04 Nov 2010 17:32:35GMT\r\n
Last-Modified: Thu, 04 Nov 2010 15:32:35 GMT\r\n
Keep-Alive: timeout=15, max=100\r\n
Connection: Keep-Alive\r\n
Transfer-Encoding: chunked\r\n
Content-Type: text/html; charset=windows-1251\r\n
\r\n
<Data here in chunks>
```

Web Browser – Multiple Requests

- Web browser parses HTML page
  - What images need downloaded?
  - What CSS files?
  - What JavaScript files?
  - What Flash objects?
- Suddenly, we have a list of perhaps 50+ objects that all need downloaded
  - Request objects over open connection
  - Open extra TCP sockets to server for parallelism

Wrapping Up

- Downloaded all the content?
  - Close TCP connections
  - Display to user



Not Yet Discussed – The Routers

- Took **for granted** that my IP packets were able to traverse the globe to MSU
- **What were the routers doing behind the scenes?**
- Setting up their forwarding tables!
  - RIP
  - OSPF
  - BGP

Forwarding Tables

RIP, OSPF, BGP

Recap – Routing

- In addition to forwarding packets, routers are busy (*asynchronously*) calculating **least-cost** routes to destinations
  - Goal: Have the forwarding table ready by the time your packet arrives with a specific destination
- **What happens if the forwarding table isn't ready, and there is no entry for your destination?**
  - Packet is dropped – you lose

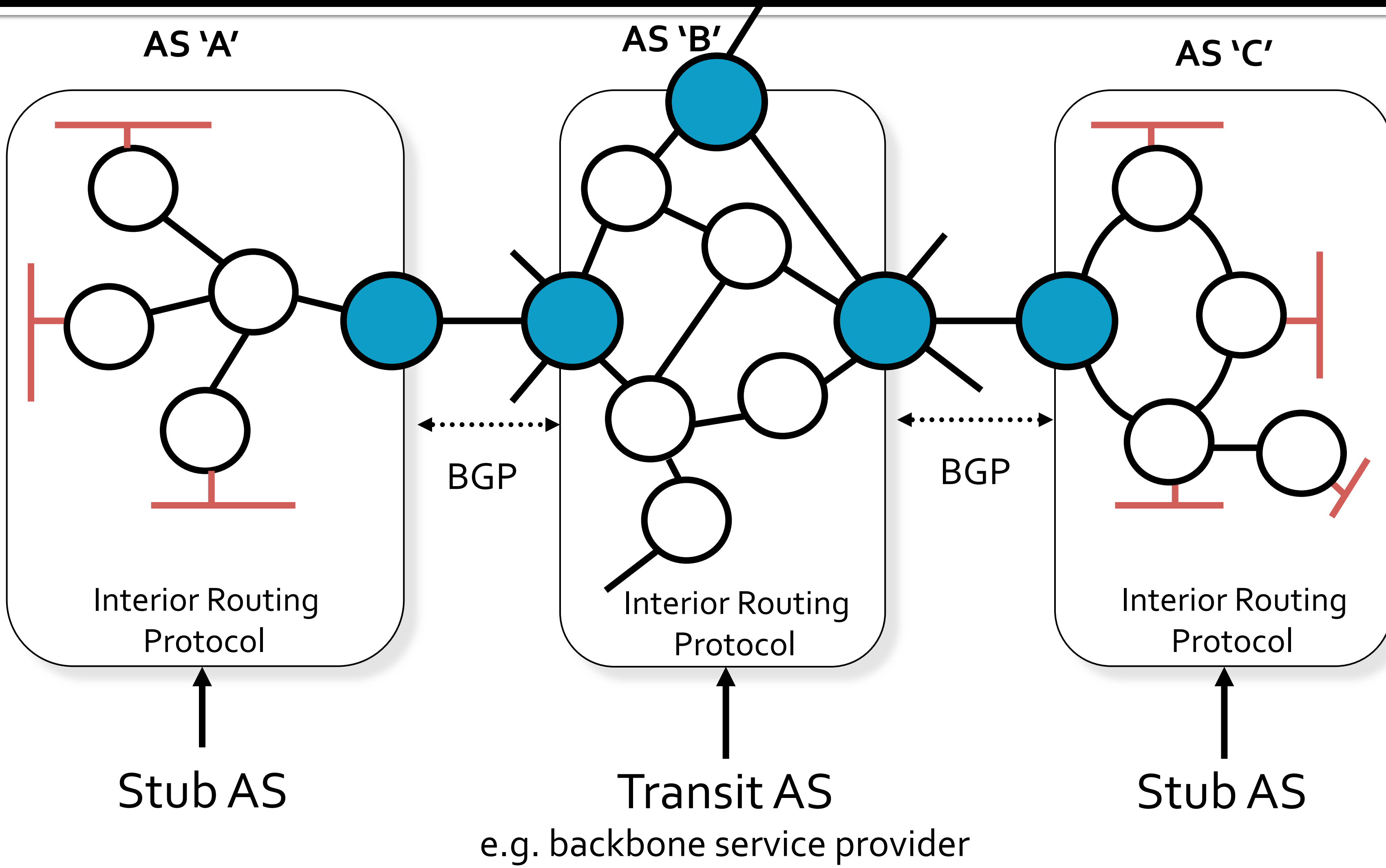
Recap – IP Routers

- **What do routers forward packets based on?**
  - Destination IP address
- **What is in the router's forwarding table?**
  - Prefixes, e.g. `138.16.9/24`
  - Next hop IP + exit port
- **What happens if more than one prefix matches the destination IP address?**
  - Longest prefix match determines winner
- **What fields do routers change in packet headers?**
  - Dest/src MAC address (*for new LAN*), TTL (*decrement by 1*), Checksum (*recalc*), Ethernet CRC (*recalc*)

Autonomous Systems

- **What is an autonomous system?**
  - Grouping of routers owned/administered by a single entity
- **Can a company only have a single autonomous system?**
  - No, can have multiple AS's
- **Why might I have more than one?**
  - Better routing for geography (i.e. North America versus Europe versus Asia)
  - Other reasons…

Autonomous Systems



BGP Basics

- BGP routers advertise prefixes (aka subnets)
  - An ***advertisement*** is a *promise:* If you give me packets destined for IP addresses in this range, I will move them closer to their destination
- BGP routers exchange advertisements to learn ***reachability***
  - What subnets can I reach by going through a particular AS?
- BGP routers exchange information to learn about all prefixes on the Internet
  - A*ggregation* can be used to reduce the number of separate entries in their tables

BGP Basics

- BGP advertises complete ***paths*** – a list of autonomous systems
  - "The network 171.64/16 can be reached via the path {AS1, AS5, AS13}"
  - Makes no use of distance vectors or link states, just ***reachability*** at a high and abstract level
- Path selection
  - Paths with loops are detected locally and ignored
  - Local policies pick the preferred path among options
  - When a link/router fails, the path is "withdrawn"

Traceroute

```
dhcp-10-6-162-134:~ shafer$ traceroute -a -q 1 www.msu.ru
traceroute to www.msu.ru (93.180.0.18), 64 hops max, 52 byte packets
 1  [AS65534] 10.6.163.254 (10.6.163.254)  1.677 ms
 2  [AS1] 10.0.0.141 (10.0.0.141)  1.116 ms
 3  [AS1] 10.0.0.90 (10.0.0.90)  1.053 ms
 4  [AS0] 138.9.253.252 (138.9.253.252)  5.200 ms
 5  [AS0] 74.202.6.5 (74.202.6.5)  8.137 ms
 6  [AS4323] pao1-pr1-xe-1-2-0-0.us.twtelecom.net (66.192.242.70)  13.241 ms
 7  [AS3356] te-9-4.car1.sanjose2.level3.net (4.59.0.229)  92.772 ms
 8  [AS3356] vlan70.csw2.sanjose1.level3.net (4.69.152.126)  8.440 ms
 9  [AS3356] ae-71-71.ebr1.sanjose1.level3.net (4.69.153.5)  11.130 ms
10  [AS3356] ae-2-2.ebr2.newyork1.level3.net (4.69.135.186)  80.992 ms
11  [AS3356] ae-82-82.csw3.newyork1.level3.net (4.69.148.42)  77.316 ms
12  [AS3356] ae-61-61.ebr1.newyork1.level3.net (4.69.134.65)  74.584 ms
13  [AS3356] ae-41-41.ebr2.london1.level3.net (4.69.137.65)  147.127 ms
14  [AS3356] ae-48-48.ebr2.amsterdam1.level3.net (4.69.143.81)  151.779 ms
15  [AS3356] ae-1-100.ebr1.amsterdam1.level3.net (4.69.141.169)  152.848 ms
16  [AS3356] ae-48-48.ebr2.dusseldorf1.level3.net (4.69.143.210)  156.349 ms
17  [AS3356] 4.69.200.174 (4.69.200.174)  168.386 ms
18  [AS3356] ae-1-100.ebr1.berlin1.level3.net (4.69.148.205)  167.652 ms
19  [AS3356] ae-4-9.bar1.stockholm1.level3.net (4.69.200.253)  192.668 ms
20  [AS3356] 213.242.110.198 (213.242.110.198)  176.501 ms
21  [AS3267] b57-1-gw.spb.runnet.ru (194.85.40.129)  198.827 ms
22  [AS3267] m9-1-gw.msk.runnet.ru (194.85.40.133)  204.276 ms
23  [AS3267] msu.msk.runnet.ru (194.190.254.118)  202.454 ms
24  [AS2848] 93.180.0.158 (93.180.0.158)  201.358 ms
25  [AS2848] 93.180.0.170 (93.180.0.170)  200.257 ms
26  [AS2848] www.msu.ru (93.180.0.18)  204.045 ms !Z
```

AS Numbers in Traceroute

AS	Name
0	Reserved (local use)
18663	University of the Pacific *(Traceroute didn't resolve this due to missing information in address registry…)*
4323	Time Warner Telecom
3356	Level 3 Communications
3267	Runnet - State Institute of Information Technologies & Telecommunications (SIIT&T "Informika")
2848	Moscow State University

First AS

- First AS is Pacific's (AS18663)
- Do a lookup on the AS
  - http://www.ripe.net/data-tools/stats/ris/routing-information-service
  - https://www.dan.me.uk/bgplookup
  - http://www.peeringdb.com/
    - Among other places…
- Pacific's gateway(s) to the Internet advertise a BGP prefix (aka subnet)
  - **`138.9.0.0/16`**

First AS

- An advertisement is a *promise:*
  - If you give me packets destined for IP addresses in this range, I will move them closer to their destination.
  - In this case, we *are* the destination!
  - This advertisement *originates* from our AS

Second AS

- Pacific buys Internet service from Time Warner (AS4323) , which has border routers that speak BGP
  - Pacific's routers talk to their routers, and they learn of our advertisement for 138.9.0.0/16
  - Now, Time Warner knows how to reach Pacific's IPs
  - We also learn of their advertisements!
    - Both for prefixes *originating* at those ISPs, and prefixes *reachable* through those ISPs

Announcements

- **When Time Warner give our routers their BGP announcements, do we get lots of tiny entries like 138.9.0.0/16?**
  - Maybe
  - But, routes can be aggregated together and expressed with smaller prefixes, e.g. `138.0.0.0/8`
    - Reduces communication time plus router CPU and memory requirements

Second AS (continued)

- Pacific had only 1 announcement
- Time Warner *originates* ~159 announcements (as of Nov 2012)
  - Some are large, e.g. `173.226.0.0/15`
  - Some are small, e.g. `159.157.233.0/24`
- Time Warner also provides transit to their *downstream* customers' prefixes, including Pacific's prefix
  - Total of ~6395 announcements (as of Nov 2012)
  - We get this full list, as does ever other (BGP-speaking) AS connected to Time Warner

Third AS

- Time Warner (AS4323) can move this packet to San Jose, where it enters the Equinix Internet Exchange (See https://www.peeringdb.com)
  - Private location to peer ("exchange traffic") with dozens of other companies
  - Akamai, Apple, Amazon, Facebook, Google, Microsoft, many ISPs, etc…
- Time Warner connects with Level 3 (AS3356)
  - *Do they pay, or is this free?*
  - Same sharing of BGP announcements occurs here

Last AS

- **The same thing is happening over in Eurasia**
- Last AS of our path is Moscow State University (AS2848)
- MSU's gateway(s) to the Internet advertise a BGP prefix for `93.180.0.0/18` (along with 3 others that *originate* in this AS)
  - That encompasses the destination IP of `93.180.0.18`

Next-to-last AS

- Moscow State University connects to Runnet (AS3267)
  - Runnet announces prefix `93.180.0.0/18` (along with 291 others reachable *downstream*, and 13 that *originate* in this AS)
  - Runnet now knows how to reach IPs that belong to MSU
- Runnet obtains transit through Level3, so our link is complete!

What's Missing?

- The forwarding table!
  - We keep forgetting to generate the forwarding table!
- Need more information
  - BGP tells us links between autonomous systems
  - Other protocols (RIP, OSPF) tell us paths within autonomous systems

Interior Routing Protocol

- Let's say (hypothetically) that Time Warner runs RIP, a distance-vector protocol
  - *We used RIP in lab instead of manually entering routes*
- **Does each router have a complete view of the network inside the AS?**
  - No, router knows physically-connected neighbors and link costs to neighbors
  - Iterative process of computation, exchange of info with neighbors
- **What algorithm is used to develop routes?**
  - Bellman-Ford (using distance vectors)

Interior Routing Protocol

- Let's say (hypothetically) that Time Warner runs OSPF, a link-state protocol
- **Does each router have a complete view of the network inside the AS?**
  - Yes, all routers know complete topology and link cost
- **What algorithm is used to develop routes?**
  - Dijkstra's

End Result is the Same

- Each router inside the AS updates its own forwarding table to direct BGP prefixes to the appropriate gateway router to the next AS
  - Rules might be very simple, i.e. just forward everything not destined to this AS to the same gateway router
  - Or rules might be complicated…
- **End result is a forwarding table for the router**
  - Prefix (for LPM)
  - Next-hop IP
  - Exit port

Transport Layer

TCP and UDP

User Datagram Protocol (UDP) Characteristics

- UDP is a connectionless datagram service.
  - There is no connection establishment: packets may show up at any time.
- UDP packets are self-contained.
- UDP is unreliable:
  - No acknowledgements to indicate delivery of data.
  - Checksums cover the header, and only optionally cover the data.
  - Contains no mechanism to detect missing or mis-sequenced packets.
  - No mechanism for automatic retransmission.
  - No mechanism for flow control, and so can over-run the receiver.

UDP

- Why is there a UDP?
  - No connection establishment (which can add delay)
  - Simple: no connection state at sender, receiver
  - Small segment header (lower byte overhead)
  - Avoids redundancy with application-provided features (such as retransmission)
  - No congestion control: UDP can blast away as fast as desired

TCP Characteristics

- TCP is connection-oriented.
  - 3-way handshake used for connection setup (SYN, SYN-ACK, ACK)
- TCP provides a bi-directional stream-of-bytes service
- TCP is reliable:
  - Acknowledgements indicate delivery of data.
  - Checksums are used to detect corrupted data.
  - Sequence numbers detect missing, or mis-sequenced data.
  - Corrupted data is retransmitted after a timeout.
  - Mis-sequenced data is re-sequenced.
  - (Window-based) Flow control prevents over-run of receiver
- TCP uses congestion control to share network capacity among users

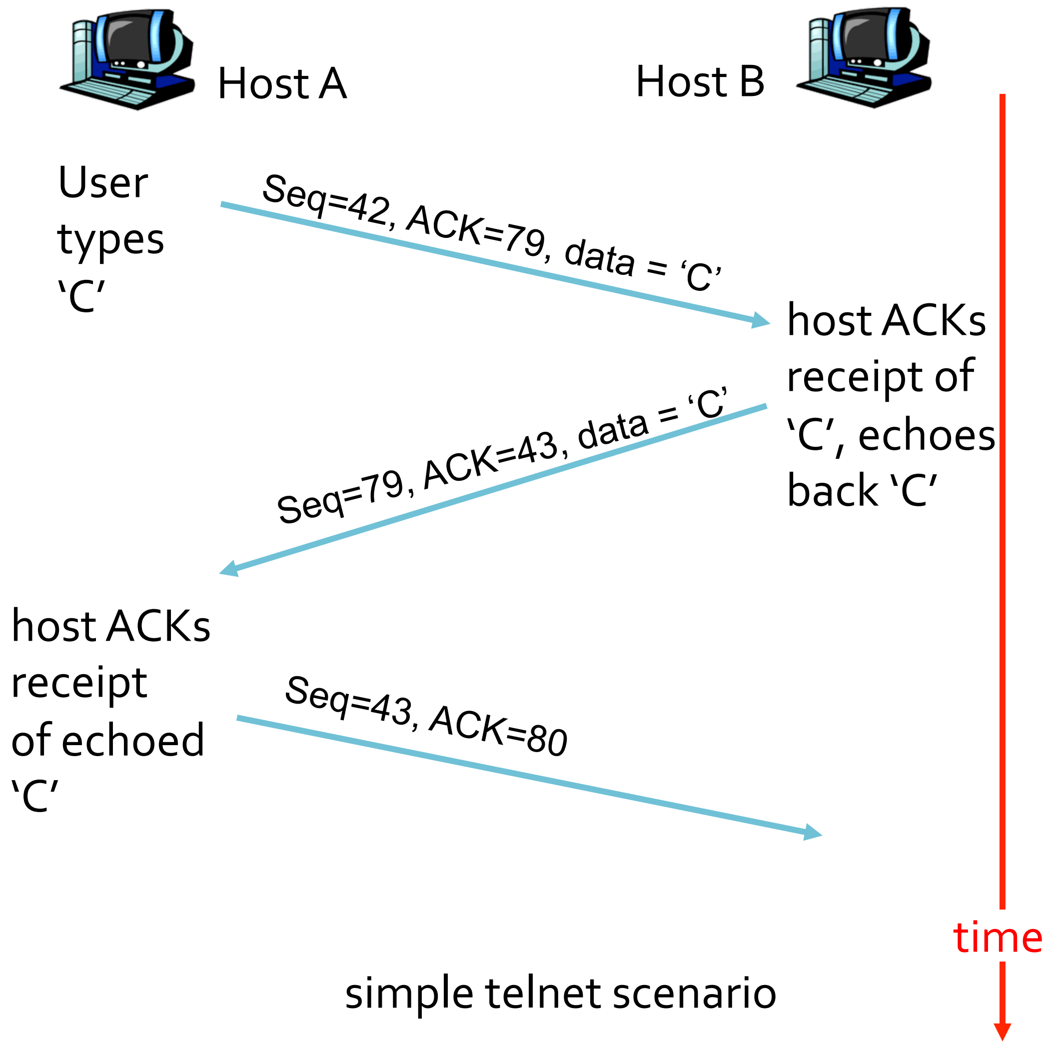
TCP seq. #'s and ACKs

Seq. #'s:

- byte stream "number" of first byte in segment's data

ACKs:

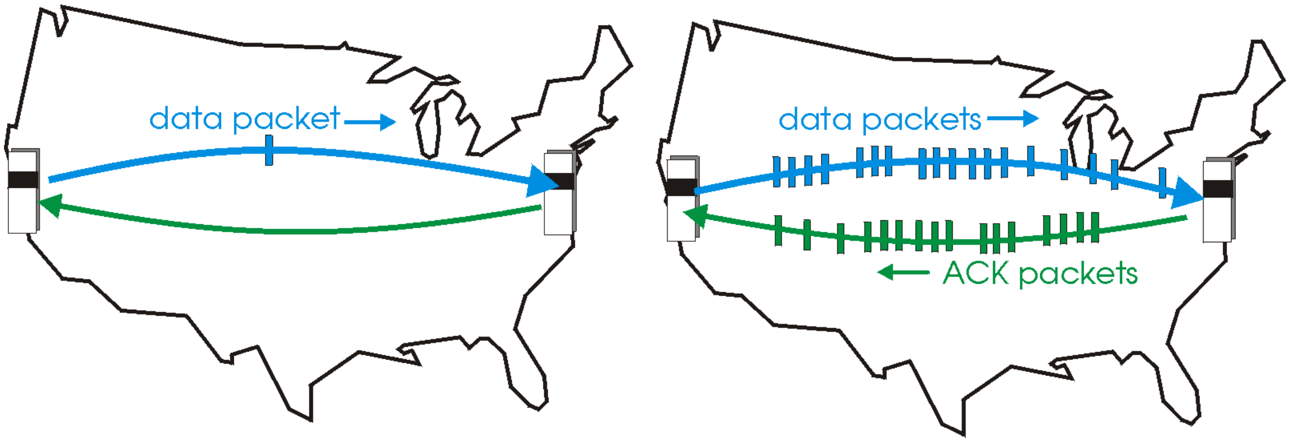
- seq # of next byte expected from other side
- cumulative ACK



simple telnet scenario

Pipelined Protocols

- **How to combat latency in a naïve stop-and-wait design?**
- Pipelining: sender allows multiple, "in-flight", yet-to-be-acknowledged packets
  - Range of sequence numbers must be increased
  - Buffering at sender and/or receiver



(a) a stop-and-wait protocol in operation (b) a pipelined protocol in operation

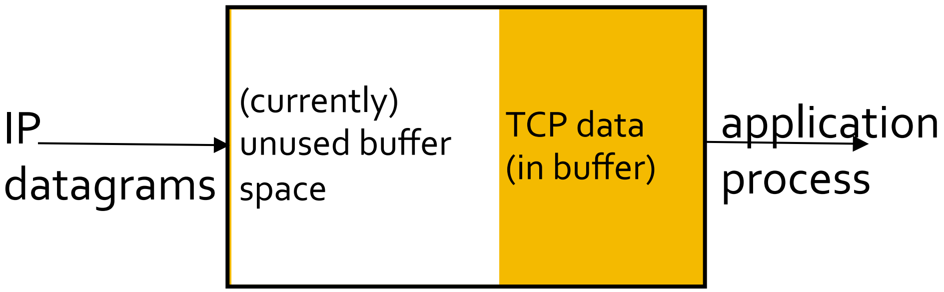
TCP Round Trip Time and Timeout

- **How to set TCP timeout value?**
- Should be longer than RTT (round-trip-time)
  - But RTT varies…
- If it is too short
  - Premature timeout
  - Unnecessary retransmissions…
- If it is too long
  - Slow reaction to segment loss

- **How can we estimate RTT?**
- Measure time from segment transmission until ACK receipt
  - Ignore retransmissions
- This sampled time will vary
  - We want a "smoother" estimated RTT
  - Weighted moving average – influence of past samples decrease quickly

TCP Flow Control

- Receive side of TCP connection has a receive buffer:



- App process may be slow at reading from buffer

Flow control
Prevents sender from overflowing receiver's buffer by transmitting too much, too fast

- *Speed-matching service:* matching send rate to receiving application's drain rate
- Receiver communicates size of unused buffer space back to sender in ACKs
  - Sender limits in-flight data to less than unused buffer

Principles of Congestion Control

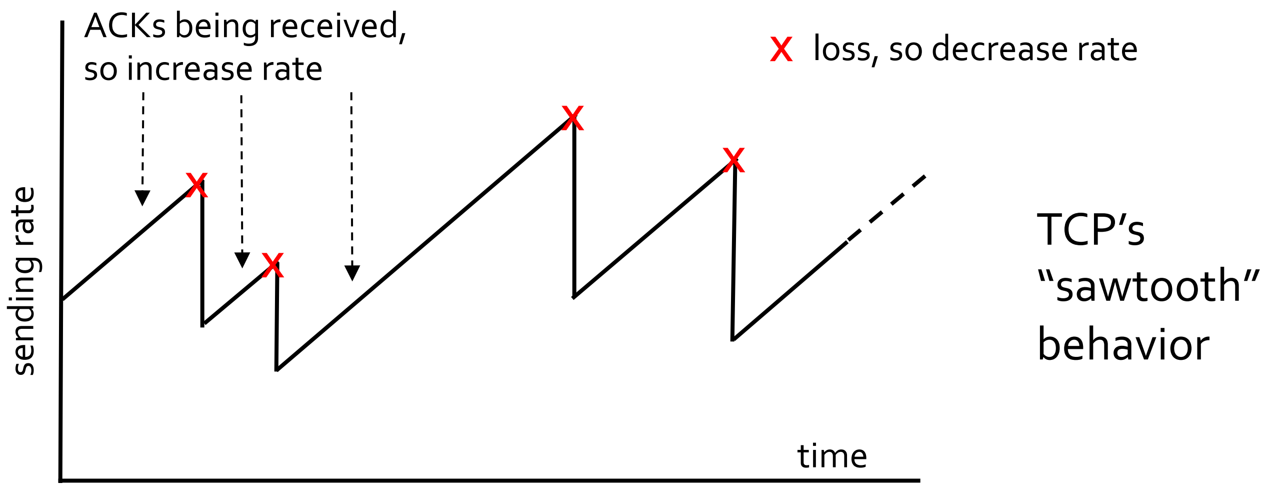
- What is congestion?
  - Informally: "too many sources sending too much data too fast for **network** to handle"
- Different from flow control!
  - *Which is too fast for the **receiver** to handle*
- Manifestations
  - Lost packets (buffer overflow at routers)
  - Long delays (queueing in router buffers)

TCP Congestion Control

- Goal: TCP sender should transmit **as fast as possible**, but without congesting network
- How do we find the rate just below congestion level?
  - Decentralized approach – each TCP sender sets its own rate, based on *implicit* feedback:
  - ACK indicates segment received (a good thing!)
    - Network not congested, so increase sending rate
  - Lost segment – assume loss is due to congested network, so decrease sending rate

TCP Congestion Control: Bandwidth Probing

- Probing for bandwidth
  - Increase transmission rate on receipt of ACK, until eventually loss occurs, then decrease transmission rate



Security vs Internet Design

Internet Design Fundamentals

- Packet-based (statistical multiplexing)
- Routing is hop-by-hop and destination-based
- Global addressing: IP addresses
- Simple to join (as infrastructure)
- Smart end hosts (end-to-end argument)
- Hierarchical naming service

Internet Design vs. Security

- **Packet-based (statistical multiplexing)**
  - Simple design
  - How to keep someone from hogging resources?
  - Difficult to put a bound on resource usage (no notion of flow, and source addresses can't be trusted)
  - Community is allergic to per-flow state
- Routing is hop-by-hop and destination-based
- Global addressing: IP addresses
- Simple to join (as infrastructure)
- Smart end hosts (end-to-end argument)
- Hierarchical naming service

Internet Design vs. Security

- Packet-based (statistical multiplexing)
- **Routing is hop-by-hop and destination-based**
  - Don't know where packets are coming from
    - Source address can be spoofed
  - Fixes are available but not widely deployed
- Global addressing: IP addresses
- Simple to join (as infrastructure)
- Smart end hosts (end-to-end argument)
- Hierarchical naming service

Internet Design vs. Security

- Packet-based (statistical multiplexing)
- Routing is hop-by-hop and destination-based
- **Global addressing (IP addresses)**
  - Everyone can talk to everyone – democracy!
  - Even people who don't necessarily want to be talked to ("every psychopath is your next door neighbor" – Dan Geer)
- Simple to join (as infrastructure)
- Smart end hosts (end-to-end argument)
- Hierarchical naming service

Internet Design vs. Security

- Packet Based (statistical multiplexing)
- Routing is hop-by-hop, destination-based
- Global Addressing (IP addresses)
- **Simple to join (as infrastructure)**
  - Untrusted infrastructure
    - Easy to grow organically, but…
  - My router has to trust what your router says
  - Misbehaving routers can violate data integrity and privacy
- Smart end-hosts (end-to-end argument)
- Hierarchical Naming Service

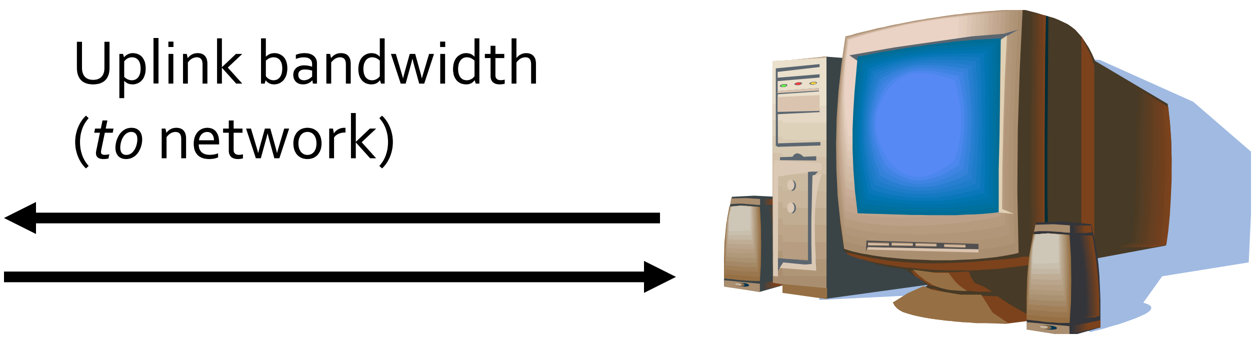
Internet Design vs. Security

- Packet-based (statistical multiplexing)
- Routing is hop-by-hop, destination-based
- Global addressing (IP addresses)
- Simple to join (as infrastructure)
- **Smart end-hosts (end-to-end argument)**
  - Decouple hosts and infrastructure = innovation at the edge!
  - **Giving power to least trusted actors**
    - Assume end hosts are "good"
    - How to guarantee good behavior? (like congestion control)
  - How to protect complex functionality at end-points?
- Hierarchical naming service

Internet Design vs. Security

- Packet-based (statistical multiplexing)
- Routing is hop-by-hop, destination-based
- Global addressing (IP addresses)
- Simple to join (as infrastructure)
- Smart end hosts (end-to-end argument)
- **Hierarchical naming service**
  - Lots of caching along the way
  - Need protection/trust at each point or response to name request can be modified

DoS: Via Resource Exhaustion



Many different resources can be exhausted!

Buffer Overflow Vulnerability

- **What is a buffer overflow attack?**
  - ```
    char buf1[8];
    char buf2[8];
    strcat(buf1, "excessive");
    ```
- End up overwriting two characters beyond buf1!
- What is beyond my buffer in memory?
  - Other variables and data? (probably buf2)
  - The stack? (further out)
  - **The return address to jump to after my function finishes?**
- If app is running as administrator, attacker now has full access!
- In network apps, incoming (**untrusted**) data from the network overflows a buffer
    ```

Other Topics

Other Topics

- Email: POP, IMAP, SMTP
  - *Review prior lecture*
- Future of the Internet: IPv6
  - *Review prior lecture*