CSE 461: Final Review
Winter 2021
Administrivia

- March 12, Friday, Tomorrow
  - Quiz 5, after the lecture
  - Project 3, due 11pm
  - Assignment 5, due 11:59pm

- March 18, next Thursday
  - Final Exam
    - 8:30 - 10:30 am

- Please fill out the course evaluation form:
  https://uw.iiasystem.org/survey/236245
Final Review Section

● Today: A brief review of lecture materials
  ○ Concepts, Protocols, Algorithms, ...

● What YOU should do after this section and before the exam:
  ○ Go through the lecture slides
  ○ Think about the problems that each protocol/algorithm tries to solve
    ■ Pros and cons of current approaches?
    ■ Any other possible solutions?
    ■ What has not been solved yet?
Before Midterm

- Physical Layer
- Link Layer
Physical Layer

- Coding: Clock Recovery - 4B/5B
- Modulation: Amplitude/Frequency/Phase Shift Keying
- Latency = Transmission Delay + Propagation Delay
- RTT: Time it takes between sending a request and getting a reply
- Bandwidth-Delay Product = R (bits/sec) x D (sec) = BD (bits)
  - Amount of data can be in transit in network
- Media: Wire/Fiber/Wireless
- Channel Limits: Shannon Capacity
Link Layer

- Framing: Byte Stuffing / Bit Stuffing
- Error Detection: Parity Bit, Internet Checksum, CRC
- Error Correction: Hamming Code
- Hamming Distance
  - Detect $d$ errors if $HD = d+1$
  - Correct $d$ errors if $HD = 2d+1$
- Retransmission: ARQ
  - Proper Timeouts
Link Layer

- Multiplexing: Time Division (TDM) / Frequency Division (FDM)
- Multiple Access: ALOHA, CSMA/CD, BEB
- Wireless: Hidden/Exposed Terminal Problem, RTS/CTS
- Switching: Backward Learning
  - Forwarding Loop? -> Spanning Tree Algorithm!
- Software Defined Networking (SDN)
  - Rising of Datacenter Networks
  - Separation of Control Plane and Data Plane
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

Datagram Model
- Connectionless service
- Packets contain destination address
- Routers looks up address in its forwarding table to determine next hop
- Example: IP

Virtual Circuits
- Connection-oriented service
- Connection establishment → data transfer → connection teardown
- Packets contain label for circuit
- Router looks up circuit in forwarding table to determine next hop
- Example: MPLS

Both of them use **Store-and-Forward packet switching**
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

![Diagram of the internet stack and IP header]
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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>

192.24.6.0  \(\rightarrow\) D  
192.24.14.32 \(\rightarrow\) B  
192.24.54.0 \(\rightarrow\) 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...

<table>
<thead>
<tr>
<th>Internal IP:port</th>
<th>External IP : port</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.12 : 5523</td>
<td>44.25.80.3 : 1500</td>
</tr>
<tr>
<td>192.168.1.13 : 1234</td>
<td>44.25.80.3 : 1501</td>
</tr>
<tr>
<td>192.168.2.20 : 1234</td>
<td>44.25.80.3 : 1502</td>
</tr>
</tbody>
</table>

**Internal source**

Src = 192.168.1.12:5523
Dst = 123.1.1.10:5000

**NAT**

**External destination**

IP=X, port=Y
Src = 44.25.80.3:1500
Dst = 123.1.1.10:5000
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.
Sometimes doesn’t perform very well: count-to-infinity scenario

Link State Routing (widely used)
Phase 1. **Topology Dissemination**: Nodes flood topology
Phase 2. **Route Computation**: running 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 (ISP & Customer)**
  - ISP announce everything it can reach to its customer
    - AS1 to AS2: you can send packet to AS4 through me
  - Customer ISP only announce its customers to ISP
    - 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
    - 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

<table>
<thead>
<tr>
<th>TCP (Streams)</th>
<th>UDP (Datagrams)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connections</td>
<td>DataTypes</td>
</tr>
<tr>
<td>Bytes are delivered once, reliably, and in order</td>
<td>Messages may be lost, reordered, duplicated</td>
</tr>
<tr>
<td>Arbitrary length content</td>
<td>Limited message size</td>
</tr>
<tr>
<td>Flow control matches sender to receiver</td>
<td>Can send regardless of receiver state</td>
</tr>
<tr>
<td>Congestion control matches sender to network</td>
<td>Can send regardless of network state</td>
</tr>
</tbody>
</table>
TCP Connection Establishment

Three-way handshake

Active party (client)
- CLOSED
- SYN_SENT
- ESTABLISHED

Passive party (server)
- CLOSED
- LISTEN
- SYN_RCVD
- ESTABLISHED

1. SYN (SEQ=x)
2. SYN (SEQ=y, ACK=x+1)
3. (SEQ=x+1, ACK=y+1)

Time

CSE 461 University of Washington
TCP Connection Release

States...

**Active party**
- ESTABLISHED
- FIN_WAIT_1
- FIN_WAIT_2
- TIME_WAIT (timeout)
- CLOSED

**Passive party**
- ESTABLISHED
- CLOSE_WAIT
- LAST_ACK
- CLOSED

**Messages**
1. FIN (SEQ=x)
   - (SEQ=y, ACK=x+1)
2. FIN (SEQ=y, ACK=x+1)
   - (SEQ=x+1, ACK=y+1)
Flow Control - Sliding Window Protocol

- Instead of stop-and-wait, sends $W$ packets per 1 RTT
  - To fill network path, $W=2B$
- 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
Max-Min Fair Allocation

- Start with all flows at rate 0
- Increase the flows until there is a new bottleneck in the network
- Hold fixed the rate of the flows that are bottlenecked
- 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 - Additive Increase Multiplicativc Decrease

- Additive Increase
- Multiplicative Decrease

Host 1 or 2’s Rate

Host 1

Multiplicative Decrease

Additive Increase

Time
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 $\frac{1}{2}$, no slow-start
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
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.

- **Random Early Detection (RED)**
  - Drop at random, depending on queue size.
  - As queue approaches full, increase likelihood of packet drop.
  - Example: 1 queue slot left, 10 packets expected, 90% chance of drop.
Application Layer

- DNS
- HTTP
- Web Caching / CDN
- Security
DNS

- Terminology
  - Names - higher-level identifiers for resources
  - Addresses - lower-level locators for resources
  - Resolution - mapping a name to an address
  - Zones - contiguous portions of the namespace
  - Nameserver - server to contact for information about a particular zone
- Maps between host names and address
- Recursive vs iterative query
- Caching
- Built on top of UDP
- Security (DNS Spoofing)
HTTP - HyperText Transfer Protocol

Basis for fetching Web pages

Steps to fetch a web HTTP with URL:

- Resolve the server IP
- Setup TCP connection
- Send/Receive HTTP request over TCP
- Fetch embedded resources
- Teardown TCP connection
HTTP

Components of a web page

● HTML - Markup language for web content
● DOM - Base primitive for web browsers interacting with HTML
● Dynamic contents (client or server)

HTTP Protocol

● Commands in request- GET/POST/...
● Codes in response - 2xx=Success/4xx=Client Error/...
HTTP

REST - Representational State Transfer

- HTTP for general network services
- For HTTP-based APIs -- RESTful APIs
- Tenants
  - Uniform Interface
  - Client/Server
  - Stateless
  - Cachable
  - Layered
HTTP

How to decrease Page Load Time (PLT)?

- Parallel connections and persistent connections (HTTP1.1)
- HTTP caching and proxies
- Change HTTP protocol
- Move content closer to client (CDNs)
HTTP

HTTP2/SPDY

- Prioritized Stream Multiplexing
  - multiple concurrent HTTP connections in a single TCP flow
- Header Compression
- Server push
  - Push resource with HTML response

HTTP3/QUIC

- UDP
  - Congestion
  - Encryption
- TLS
- IP Mobility support
HTTP Request over TCP+TLS (with 0-RTT)

Client

- TCP SYN
- TCP SYN + ACK
- TCP ACK
- TLS ClientHello
- HTTP Request
- TLS ServerHello
- HTTP Response
- TLS Finished

Server

HTTP Request over QUIC (with 0-RTT)

Client

- QUIC
- HTTP Request
- QUIC
- HTTP Response
- QUIC

Server
CDNs

- Content Delivery Networks
- Place popular content near clients
  - Use DNS to place replicas across the Internet for use by all nearby clients
  - Reduces server, network load
  - Improves user experience
Caching/Proxies

Web Caching
- Local copy on browser
- Revalidate copy with remote server
  - Timestamp
  - Server header

Web Proxies
- Placed near pool of clients
  - Caching
  - Security checking
  - Organization policies
Security (details in slides of last section)

- Encryption, for confidentiality
  - Symmetric encryption (one key)
  - Public key encryption (public/private key pair)
- Message Authentication, for integrity and authenticity
  - MAC: Message Authentication Code
  - Digital Signature
- Web Security
  - HTTPS -> HTTP over SSL/TLS