Networks Introduction
CSE 333 Fall 2023

Instructor: Chris Thachuk

Teaching Assistants:

Ann Baturytski
Yuquan Deng
Noa Ferman
James Froelich
Hannah Jiang
Yegor Kuznetsov

Humza Lala
Alan Li
Leanna Mi Nguyen
Chanh Truong
Jennifer Xu
Lecture Outline

- Introduction to Networks
  - Layers upon layers upon layers...
Today’s Goals

- Networking is a very common programming feature
  - You will likely have to create a program that will read/write over the network at some point in your career

- We want to give you a basic, high-level understanding of how networks work before you use them
  - Lecture will be more “story-like;” we will purposefully skip over most of the details, but hopefully you will learn something new about the Internet today!
  - Take CSE 461 if you want to know more about the implementations of networks (the course is pretty cool 😊)

- Let’s also examine “the network” as a system
Networks From 10,000 ft

clients

servers
“Network” Latency is Highly Variable

- Jeff Dean’s “Numbers Everyone Should Know” (LADIS ‘09)

![Numbers Everyone Should Know Table]

<table>
<thead>
<tr>
<th>Operation</th>
<th>Time (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache reference</td>
<td>0.5</td>
</tr>
<tr>
<td>Branch mispredict</td>
<td>5</td>
</tr>
<tr>
<td>L2 cache reference</td>
<td>7</td>
</tr>
<tr>
<td>Mutex lock/unlock</td>
<td>100</td>
</tr>
<tr>
<td>Main memory reference</td>
<td>100</td>
</tr>
<tr>
<td>Compress 1K bytes with Zippy</td>
<td>10,000</td>
</tr>
<tr>
<td>Send 2K bytes over 1 Gbps network</td>
<td>20,000</td>
</tr>
<tr>
<td>Read 1 MB sequentially from memory</td>
<td>250,000</td>
</tr>
<tr>
<td>Round trip within same datacenter</td>
<td>500,000</td>
</tr>
<tr>
<td>Disk seek</td>
<td>10,000,000</td>
</tr>
<tr>
<td>Read 1 MB sequentially from network</td>
<td>10,000,000</td>
</tr>
<tr>
<td>Read 1 MB sequentially from disk</td>
<td>30,000,000</td>
</tr>
<tr>
<td>Send packet CA-&gt;Netherlands-&gt;CA</td>
<td>150,000,000</td>
</tr>
</tbody>
</table>
The Physical Layer

- Individual bits are modulated onto a wire or transmitted over radio
  - Physical layer specifies how bits are encoded at a signal level
  - Many choices, e.g., encode “1” as +1v, “0” as -0v; or “0”=+1v, “1”=-1v, ...
Materials Matter – Latency

- Fiber optic cables are lower-latency and higher-bandwidth than traditional copper wiring
  - Much of the internet’s “long haul” data is transmitted on these
  - (signal attenuation is much better too)

- Is it faster to send 1 person from UW to …
  - Downtown Seattle?
  - Ballard?

(not just distance, but also speed limit & number of lanes, mode of transportation, route, traffic, etc.)
The Data Link Layer

- Multiple computers on a LAN contend for the network medium
  - Media access control (MAC) specifies how computers cooperate
  - Link layer also specifies how bits are “packetized” and network interface controllers (NICs) are addressed

MAC addresses:
- 00:1d:4f:47:0d:48
- 4c:44:1e:8f:12:0e
- 7a:37:8e:fc:1a:ea
- de:ad:be:ef:ca:fe
- 01:23:32:10:ab:ba

Diagram:
- Ethernet header: destination address, source address, data
- Physical layer:
  - Data link
  - Ethernet
  - NIC
  - Computer
The Network Layer (IP)

- Internet Protocol (IP) routes packets across multiple networks
  - Every computer has a unique IP address
  - Individual networks are connected by routers that span networks

**IPv4 addresses:**

1. 128.95.10.55
2. 128.95.10.72
3. 128.95.10.95

**Networks and Devices:****

- **Network 1:**
  - 128.95.10.1
  - 128.95.10.55
  - 128.95.10.72
  - 128.95.10.95

- **Network 2:**
  - 128.95.4.1
  - 128.95.4.3
  - 128.95.4.10
  - 128.95.4.12
The Network Layer (IP)

- There are protocols to:
  - Let a host map an IP to MAC address on the same network
  - Let a router learn about other routers to get IP packets one step closer to their destination
The Network Layer (IP)

- **Packet encapsulation:**
  - An IP packet is encapsulated as the payload of an Ethernet frame
  - As IP packets traverse networks, routers pull out the IP packet from an Ethernet frame and plunk it into a new one on the next network
Distance Matters – Latency

- Distances within a single datacenter are smaller than distances across continents

- Even within a datacenter, distances can sometimes matter

123Net Data Center, Wikimedia
Topology Matters – Latency, Reliability

- Some places are surprisingly well- or poorly-connected to “backbone” infrastructure like fiber optic cables
- Unintuitive topology can create interesting failures
  - e.g., 2006 7.0-magnitude Hengchun Earthquake disrupted communications to Singapore, Philippines, Thailand, China, etc. for a month
Reflect and Discuss

✧ Does this system of submarine cable connections seem ‘optimal’ to you?
✧ If not, who influences the decision-making process and what might their motivations be?
  ▪ Explore the map here: https://www.submarinecablemap.com/
Submarine Cable Network Today

- ~436 fiber optic cables currently in use
  - Supports 99% of transoceanic communication
  - Primarily laid during early 2000’s “fiber boom”, but still occasional new cables and decommissioned cables

- Owners
  - Telecom carriers
  - Content providers

- Users
  - You and many others...

- Explore the network and its history:
  [http://www.surfacing.in](http://www.surfacing.in)
The Transport Layer

- Provides an interface to treat the network as a *data stream*
- Provides different protocols to interface between source and destination:
  - *e.g.*, Transmission Control Protocol (TCP), User Datagram Protocol (UDP)
  - These protocols still work with packets, but manages their order, reliability, multiple applications using the network...

Note that we have the *abstraction* of a direct connection.
The Transport Layer (TCP)

- Transmission Control Protocol (TCP):
  - Provides applications with reliable, ordered, congestion-controlled byte streams
    - Sends stream data as multiple IP packets (differentiated by sequence numbers) and retransmits them as necessary
    - When receiving, puts packets back in order and detects missing packets
  - A single host (IP address) can have up to $2^{16} = 65,535$ “ports”
    - Kind of like an apartment number at a postal address (your applications are the residents who get mail sent to an apt. #)
The Transport Layer (TCP)

- Packet encapsulation – one more nested layer!

Diagram:
- Ethernet header
  - IP header
    - TCP header
      - TCP chunk 1
        - src, dst, port
        - TCP payload
      - IP payload
    - ethernet payload
  - IP payload
    - TCP header
      - TCP chunk 2
        - src, dst, port
        - TCP payload
      - IP payload
    - ethernet payload
The Transport Layer (TCP)

- Applications use OS services to establish TCP streams:
  - The “Berkeley sockets” API
    - A set of OS system calls (part of POSIX for Linux)
  - Clients `connect()` to a server IP address + application port number
  - Servers `listen()` for and `accept()` client connections
  - Clients and servers `read()` and `write()` data to each other

```
transport -------- data link -------- physical
    ^      |        |               |
    |      v        v               v
    |      network  network  network
    |               |        |
    v               v        v
    physical       physical physical
```

Same as for file I/O
The Transport Layer (UDP)

- User Datagram Protocol (UDP):
  - Provides applications with *unreliable* packet delivery
  - UDP is a really thin, simple layer on top of IP
    - Datagrams still are fragmented into multiple IP packets

Diagram of network layers: transport, network, data link, physical.
The Transport Layer

TCP:

UDP:
The (Mostly Missing) Layers 5 & 6

- **Layer 5: Session Layer**
  - Supposedly handles establishing and terminating application sessions
  - Remote Procedure Call (RPC) kind of fits in here

- **Layer 6: Presentation Layer**
  - Supposedly maps application-specific data units into a more network-neutral representation
  - Encryption (SSL) kind of fits in here
The Application Layer

- Application protocols
  - The format and meaning of messages between application entities
    - *e.g.*, HTTP is an application-level protocol that dictates how web browsers and web servers communicate
      - HTTP is implemented *on top of* TCP streams
The Application Layer

- Packet encapsulation:

- Ethernet header
- IP header
- TCP header
- HTTP header

- Destination address
- Source address
- Data

- HTTP payload (e.g., chunk of HTML page)
The Application Layer

- Packet encapsulation:

| ethernet header | IP header | TCP header | HTTP header | HTTP payload (e.g., chunk of HTML page) |
The Application Layer

- Popular application-level protocols:
  - **DNS:** translates a domain name (e.g., www.google.com) into one or more IP addresses (e.g., 74.125.197.106)
    - Domain Name System
    - An hierarchy of DNS servers cooperate to do this
  - **HTTP:** web protocols
    - Hypertext Transfer Protocol
  - **SMTP, IMAP, POP:** mail delivery and access protocols
  - **SSH:** secure remote login protocol
    - Secure Shell
  - **bittorrent:** peer-to-peer, swarming file sharing protocol
netcat demo (if time)

- netcat (`nc`) is “a computer networking utility for reading from and writing to network connections using TCP or UDP”

- **Listen on port:** `nc -l <port>`
- **Connect:** `nc <IPaddr> <port>`
  - Local host: `127.0.0.1`