History / OSI Model

Network Programming

Early Computers

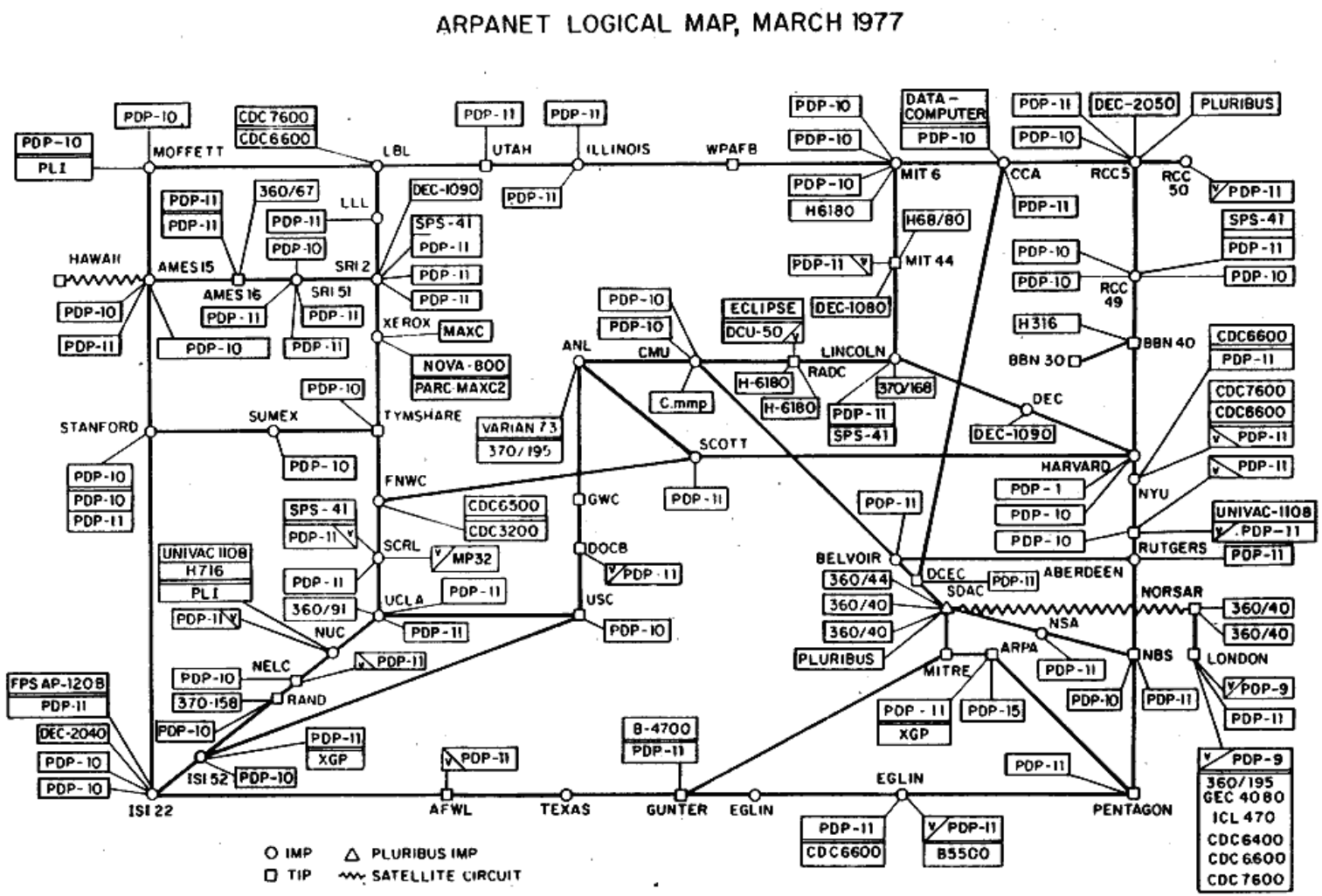
- No network
  - Large, centralized machine
  - Batch processing

- **"I think there is a world market for maybe five computers." – TJ Watson, IBM president, 1943**

Ancient History

- 1969: Advanced Research Projects Agency (ARPA) funded creation of ARPANET
- Four initial sites: UCLA, Stanford, UCSB, and U. of Utah
- 1983: TCP/IP became standard method of data transmission
- 1990: ARPANET decommissioned

ARPANET March, 1977

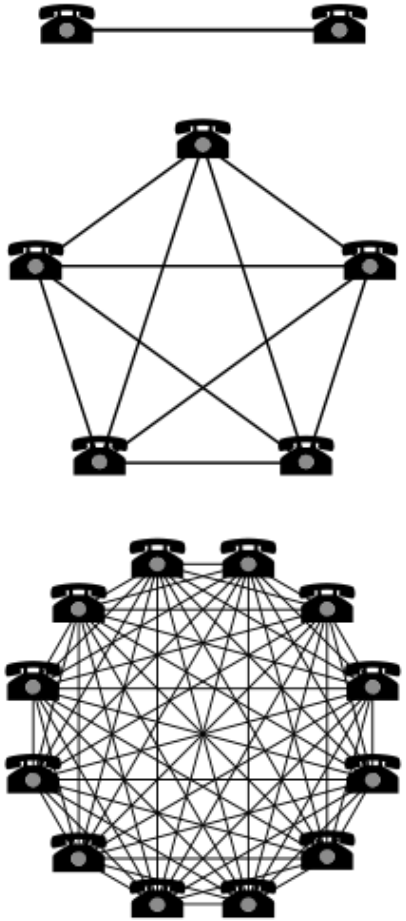


(PLEASE NOTE THAT WHILE THIS MAP SHOWS THE HOST POPULATION OF THE NETWORK ACCORDING TO THE BEST INFORMATION OBTAINABLE, NO CLAIM CAN BE MADE FOR ITS ACCURACY)

NAMES SHOWN ARE IMP NAMES, NOT (NECESSARILY) HOST NAMES

Metcalfe's Law

- Value of network proportional to square of connected users
- Bob Metcalfe created Ethernet in 1973
- Good for ideas, bad for data transmission
- Congestion!



Other Historic Events

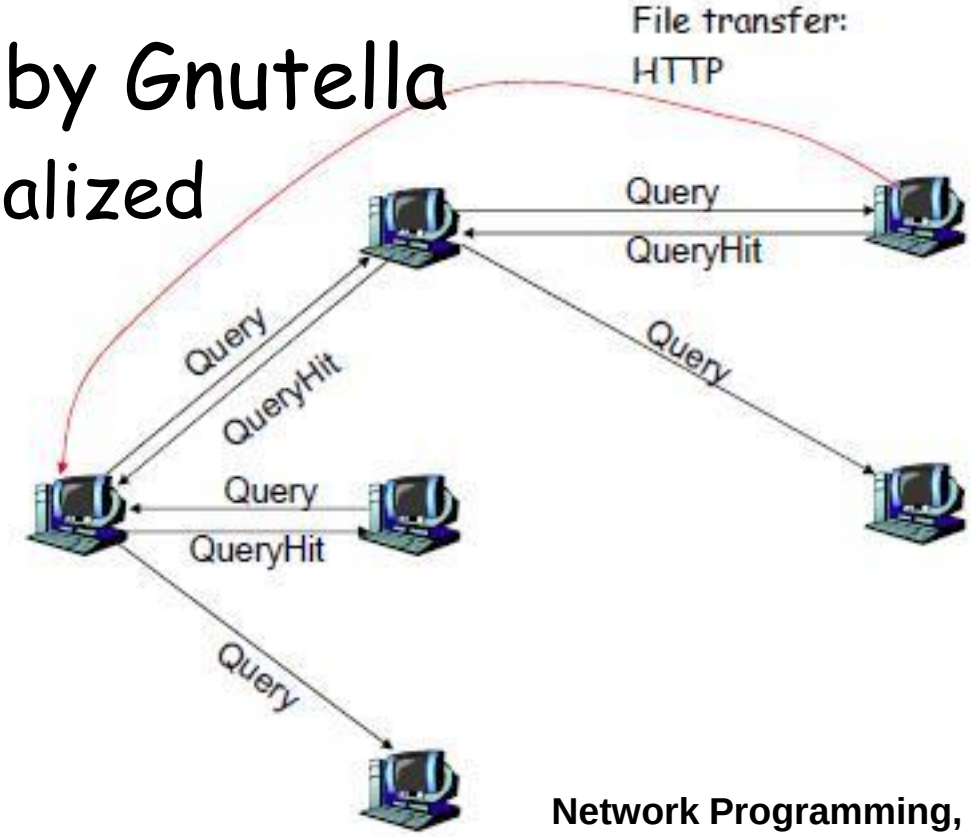
- 1968: BBN creates first Interface Message Processor (IMP) (i.e., router)
- 1969: First RFC written
- 1974: Early TCP version (Kahn & Cerf)
- 1977/8: TCP split into TCP and IP
- 1983: 4.2BSD introduces sockets
- 1993: NCSA releases Mosaic browser
- 2007: iPhone released

Clients & Servers

- Servers typically provide access to a resource or service
  - Typically up 24/7, aim for "5 nines" uptime
- Clients on the other hand, often connect to servers to utilize services
  - On average, simpler than servers

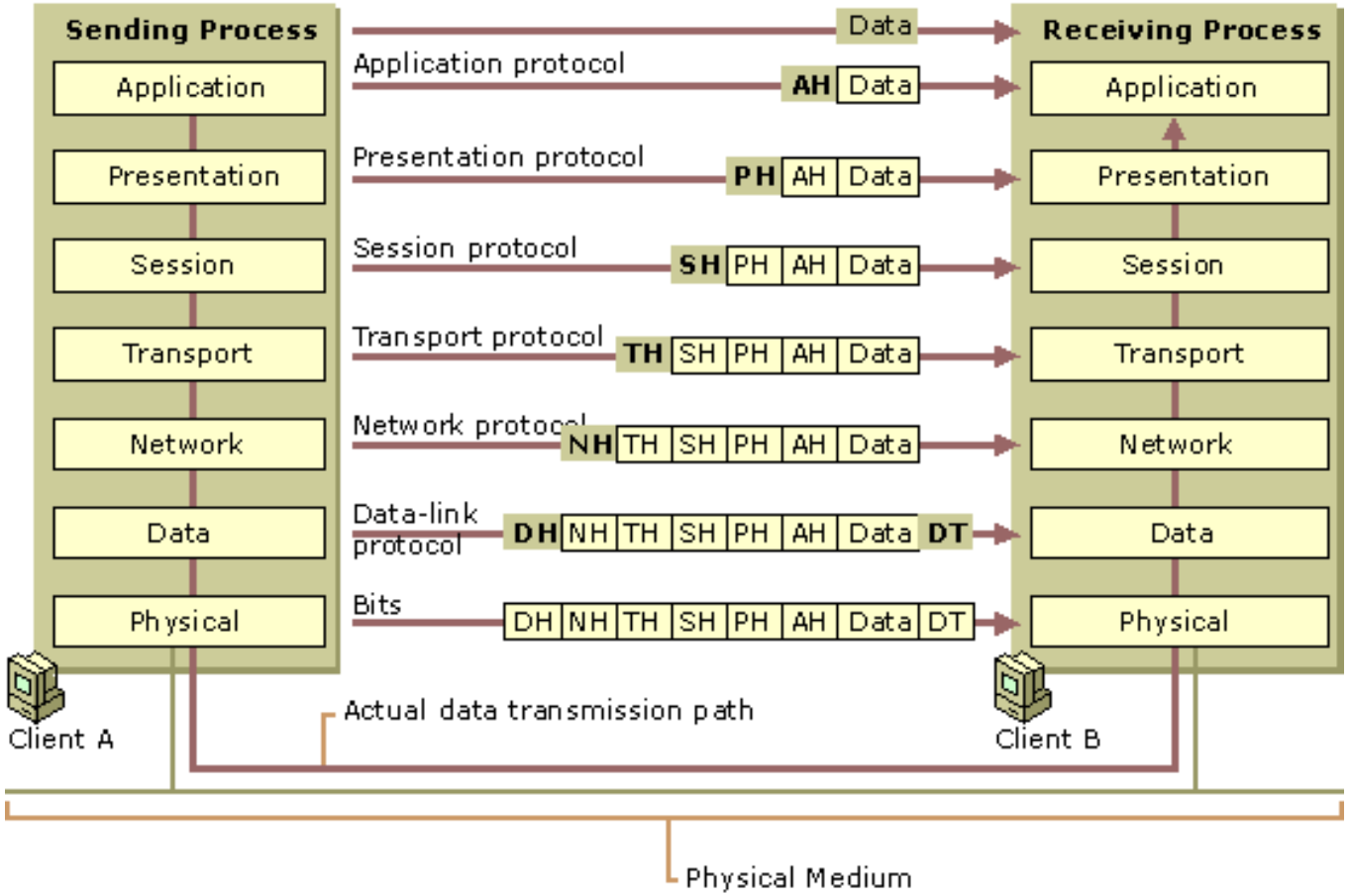
Napster

- 1999 P2P music sharing, Shawn Fanning
- Shut down 2001
  - Possible because of central server!
- Followed by Gnutella
  - Decentralized



OSI Model

- Open Systems Interconnection, seven layer model



"Layering Considered Harmful"

- Too much overhead w/ respect to all the layers
- Simplify by collapsing layers 5-7 into single "Application Layer"
  - Sometimes treat layers 1-2 as a single "Link Layer" as well
- "It is always possible to agglutinate multiple separate problems into a single complex interdependent solution. In most cases this is a bad idea." –RFC 1925

Halloween Document

- Internal Microsoft documents
- Microsoft was very anti-Linux!
- Spread FUD followed by "embrace, extend, extinguish"
- Microsoft has adopted newer & less aggressive stance towards Linux / OSS

Unix Tools

Network Programming

arp

- Address Resolution Protocol
  - Maps IPs to MAC addresses
- Example scenario:
  - *sender* broadcasts request for *dest* on LAN
  - *dest* replies to *sender* with its MAC address
- Ties layers 2 and 3 together
- Typically cached for efficiency
  - arp spoofing possible…

ping

- From the man page: "The ping utility uses the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway."

- Essentially, is the host up?
- Response indicates status of route/host

traceroute

- Each packet gets initial TTL
  - Decremented at every hop
- Essentially "pings" destination with increasing TTL values
- When TTL reaches zero, send back Time Exceeded response
  - Increment TTL, fire off another "ping"
- On MS Windows, called *tracert*

ifconfig

- Configure an interface
  - Or, by default, output interface configuration
- Find your IP, subnet mask, MTU, etc.
- Behaves slightly differently between Linux and others
  - Linux provides RX/TX statics
- Try pinging broadcast address from ifconfig

netstat

- Lists state of all active sockets
  - -a flag to see much more information
- Also shows routing tables if requested
- Can access much of the underlying networking data structures
  - Caveat: non-standard output, flags

lsof

- List Open Files
  - Normal files, directories, streams, etc.
  - Who opened that file?
- Must use –i flag to see open connections
  - Similar in function to netstat
- Options by default OR'd together
  - AND them with the –a option

dtrace

- Tracing framework developed by Sun, ported to Mac, finally Linux?
- View CPU usage, memory, etc.
- Dynamic, i.e., always present but minimal overheads when not in use
  - Tools must be built with this type of instrumentation ahead of time
- Ex:
  - sudo dtrace -n 'syscall:::entry {\
    @num[probefunc] = count(); }'

route

- Show current state of internal routing tables
- Able to modify local routing decisions through this command
- We likely won't be mucking with routing tables in this class

dig / nslookup

- Do name translation (hostname -> IP)
- Query DNS servers (or local files)
- Ask for more than just hostname mappings
  - MX (mail server), NS (nameserver), others…
- Reverse lookup (IP -> hostname) as well!

tcpdump

- Command line packet analyzer
- Quickly view traffic
  - Filter on interface, port, etc.
  - Also by expressions, hosts…
- If you're logged in to a remote machine, don't forget to exclude *your* traffic!
- See the man page for tons of details!

Wireshark

- You may need to download this one…
- Graphical counterpart to tcpdump
- Clickable packet data
  - Goes both ways!
- Filterable as well
  - Build bigger & more specific filters to track down traffic of interest!

netcat

Network Programming

Networking Swiss Army Knife

- TCP / UDP
- Can use any port or local IP address
- Can read or write data
- Client or server modes available!

Client mode

- Default operation (no flags)
- Ex: nc www.cs.rpi.edu 80
- Uses stdin

Server mode

- Requires –l (listen) option
- Ex: nc –l 1234
- Opens port 1234, ready to accept

DNS automatically

- Netcat by default will use DNS for resolving hostnames
- Bypass this by supplying -n option

Netcat Uses

- Scripting
- File transfer
  - nc –l 1234 > foo.txt
- Relays! (see cheat sheet)
- Find (unintended) open ports on your own system

Port scanning

- nc -z localhost 1-100
  - -z option doesn't try to fully connect, just scan for listening ports
- We can grab headers to hunt for vulnerable software by sending bogus input after connection established
  - echo "QUIT" | nc localhost 1-100

Leave Backdoors

- I don't know why you would ever want to do this so don't but here you go!
- nc -l -p 6996 -e /bin/bash
  - Many implementations have removed the -e
- There are many ways around this and you can google for them ☺

Proxying

- nc -l 12345 | nc www.google.com 80 is one directional
- We need redirection and pipes (or FIFOs)
  - mkfifo backpipe
  - nc -l 12345 0<backpipe | nc www.google.com 80 1>backpipe

Many Implementations

- Netcat
- Socat – more complex than base netcat
- Cryptcat – has additional encryption features
- Ncat from nmap, others…