Lecture 2: Layering & End-to-End

CSE 222A: Computer Communication Networks
Alex C. Snoeren

Thanks: Mike Freedman & Amin Vahdat
Lecture 2 Overview

- Short Internet history lesson
  - Growth & change

- Layering
  - Application interface
  - Transport services

- Discussion of End-to-End principle
1957: USSR launches Sputnik, first artificial earth satellite
  - U.S. responds by forming ARPA
1962: Licklider’s *Galactic Network*
1966: Roberts (MIT) *Towards a Cooperative Network of Time-Shared Computers*
1967: ACM SOSP *Multiple Computer Networks and Intercomputer Communication*
2008: Amazon EC2 (the “cloud”)
2009: Android phones hit the market at volume
Vannevar Bush established the U.S. military / university research partnership that later developed the ARPANET.

"Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and to coin one at random, "memex" will do. A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory.

"It consists of a desk, and while it can presumably be operated from a distance, it is primarily the piece of furniture at which he works. On the top are slanting translucent screens, on which material can be projected for convenient reading. There is a keyboard, and sets of buttons and levers. Otherwise it looks like an ordinary desk.

Vannevar Bush, *As We May Think*; Atlantic Monthly, July 1945.
J.C.R. Licklider

- Joseph Carl Robnett "Lick" Licklider developed the idea of a universal network, spread his vision throughout the IPTO, and inspired his successors to realize his dream by creation of the ARPANET.

“It seems reasonable to envision, for a time 10 or 15 years hence, a ‘thinking center’ that will incorporate the functions of present-day libraries together with anticipated advances in information storage and retrieval.”

“The picture readily enlarges itself into a network of such centers, connected to one another by wide-band communication lines and to individual users by leased-wire services. In such a system, the speed of the computers would be balanced, and the cost of the gigantic memories and the sophisticated programs would be divided by the number of users.”


Source: Livinginternet.com
1969 Internet Map

BBN TECHNOLOGIES

CSE 222A – Lecture 2: Layering & End-to-End
Chicago Data Center (2008)
Racks and racks and racks…
Internet Timeline

- 1971: Tomlinson develops email program, big hit
- 1972: Telnet
- 1973: FTP
- 1974: TCP
- 1978: TCP split into TCP and IP
- 1979: USENET established using UUCP between Duke and UNC by Tom Truscott, Jim Ellis, and Steve Bellovin
- 1984: 1000 hosts, DNS introduced
- 1988: Internet worm brings down 10% of Internet
- 1991: WAIS, Gopher, WWW released
Internet Growth Trends

- 1977: 111 hosts on Internet
- 1981: 213 hosts
- 1983: 562 hosts
- 1984: 1,000 hosts
- 1986: 5,000 hosts
- 1987: 10,000 hosts
- 1989: 100,000 hosts
- 1992: 1,000,000 hosts
- 2001: 150 – 175 million hosts
- 2002: over 200 million hosts
- 2011: over 2 billion users
  - Over 1 billion of them are on mobile devices (per KPCB)
End hosts are changing!

![Global Internet Device Sales Graph](image)

Source: Gartner, IDC, Strategy Analytics, company filings, BI Intelligence estimates
Internet History

- Goal: effective multiplexed use of existing networks
  - Minimal support from underlying networks
    - e.g., no support for multicast, real-time, fast failover, congestion control, etc.
  - Packet switching (fine-grained resource sharing)
    - AT&T said it could not be built
  - Routers connecting networks

- Recommended reading:
  - "Where Wizards Stay Up Late" by Hafner & Lyon
Internet History: Other Goals

- Survive hardware failure
- Support multiple types of applications
- Run on wide variety of networks
- Distributed management of resources
- Cost-effective
- Low cost host attachment
- Accounting
Survivability

- Internet approach
  - Cheap, commodity components
  - Stateless routers + self-healing
  - Keep routing simple (non-adaptive)
  - End to end recovery

- Telephone approach
  - Ultra reliable switches
  - Make the network very smart
OSI Model

Function
- Ultimate data destination
- Format conversion
- Interaction across presentation
- Reliable, ordered delivery
- Routing/Internetworking
- Data framing over links
- Bits on the wire

Application

Presentation

Session

Transport

Network

Data link

Physical

Example
- Web browser
- ASCII/XDR
- Restartable file transfer
- TCP
- IP
- Ethernet, ATM
- SONET, 100BT

CSE 222A – Lecture 2: Layering & End-to-End
Where does security go?

What about reliability?
OSI Model Discussion

- OSI standardized before implemented
  - IETF philosophy: “We reject kings, presidents and voting. We believe in rough consensus and working code”
  - IETF requires two working/interoperable versions before considering a standard

- Modular design, but some boundaries are arbitrary
  - Why seven layers?
  - What exactly is the session layer?
  - Much basic network functionality at multiple layers
    - Reliability, flow control, security
Internet Architecture

- IP Hourglass:

  - Telecollaboration
  - NFS
  - HTTP
  - Email
  - Rlogin
  - RPC
  - TCP
  - UDP
  - IP
  - RSVPP
  - Ethernet
  - Modem
  - ATM
  - PPP
  - SONET
  - Packet radio
  - Air
  - 100BT

- Layering not strict
  - Can define new abstractions on any existing protocol
Layering in Applications

- **Bottlenecks**
  - Boundary crossings
  - Copies
  - Context switches

- **Layering nice way to logically consider protocols**
  - May not lead to fastest implementation
  - But! Processors are getting faster... people are getting more expensive
Socket Abstraction

- Best-effort packet delivery is a clumsy abstraction
  - Applications typically want higher-level abstractions
  - Messages, uncorrupted data, reliable in-order delivery

- Applications communicate using “sockets”
  - Stream socket: reliable stream of bytes (like a file)
  - Message socket: unreliable message delivery
Two Basic Transport Features

- **Demultiplexing**: port numbers
  - Service request for 128.2.194.242:80 (i.e., the Web server)

- **Error detection**: checksums

---

CSE 222A – Lecture 2: Layering & End-to-End
Two Main Transport Layers

- **User Datagram Protocol (UDP)**
  - Just provides demultiplexing and error detection
  - Header fields: port numbers, checksum, and length
  - Low overhead, good for query/response and multimedia

- **Transmission Control Protocol (TCP)**
  - Adds support for a “stream of bytes” abstraction
  - Retransmitting lost or corrupted data
  - Putting out-of-order data back in order
  - Preventing overflow of the receiver buffer
  - Adapting the sending rate to alleviate congestion
  - Higher overhead, good for most stateful applications
Sharing the ‘Net

- Best-effort network easily becomes overloaded
  - No mechanism to “block” excess calls
  - Instead excess packets are simply dropped

- Examples
  - Shared Ethernet medium: frame collisions
  - Ethernet switches and IP routers: full packet buffers

- Quickly leads to congestion collapse

Increase in load that results in a decrease in useful work done.
Adjusting to Congestion

- End hosts adapt their sending rates
  - In response to network conditions

- Learning that the network is congested
  - Shared Ethernet: carrier sense multiple access
    » Seeing your own frame collide with others
  - IP network: observing your end-to-end performance
    » Packet delay or loss over the end-to-end path

- Adapting to congestion
  - Slowing down the sending rate, for the greater good
  - But, host doesn’t know how bad things might be…
Ethernet Back-off Mechanism

- Carrier sense: wait for link to be idle
  - If idle, start sending; if not, wait until idle
- Collision detection: listen while transmitting
  - If collision: abort transmission, and send jam signal
- Exponential back-off: wait before retransmitting
  - Wait random time, exponentially larger on each retry
TCP Congestion Control

- Additive increase, multiplicative decrease
  - On packet loss, divide congestion window in half
  - On success for last window, increase window linearly
End-To-End Argument

Where to Place Functionality?

Application
TCP
IP
Router
Link Layer
End-to-End Argument

- Functionality should be implemented at a lower layer if and only if it can be correctly and completely implemented there
  - Should not be implemented at lower level if redundant with higher level
  - Performance optimizations are not a violation

- Early example
  - ARPANet provided reliable link transfers between switches
  - Packets could still get corrupted on host-to-switch link, or inside switches
  - Want to know if host *acted* on the request not whether it *received* it
Example: Reliable File Transfer

- From disk on file (web) server over network to client
  - Disk can introduce bit errors
  - Host I/O buses can introduce bit errors
  - Packets can get garbled, dropped, misordered at any node
- Solution: integrity check on file, not per packet or per hop
Hop by Hop for Performance

- Does not violate end to end argument to provide reliability at link layer, even if not required for correct operation

- For file transfer application, consider varying conditions:
  - Prob(corrupted/lost packet per link) = p
  - Prob(packet lost end to end), avg. 15 hops across Internet
    - p = 0.0001% => Prob(loss) = 0.0015%
    - p = 1% => Prob(loss) = 14%

- Chance of file corruption grows with size of file
  - Potentially retransmit entire file for one lost packet?
Application-Specific Semantics

- Example: move reliability into the network communication protocol (such as TCP)
  - Certain computational and bandwidth overheads to implementing reliable, in-order delivery in the network
  - Not all applications want to pay this overhead

- Real-time voice/audio
  - Better to drop a packet, rather than hold up later packets
  - *On-time* delivery more important than *reliability*

- Applications should be able to pick and choose the semantics they require from underlying system
  - *Active Networks*, *overlay networks*
Other Examples

- Distributed transactions
  - Exactly once vs. at most once vs. at least once

- Security
  - Security only as strong as weakest assumption/link

- Suppressing duplicate messages
  - Duplicate effort between network and application layer

- Misordered messages
Discussion

- When should the network support a function?
  - E.g., link-layer retransmission in wireless networks?
- Who’s interests are served by the e2e argument?
- How does a network operator influence the network without violating the e2e argument?
- Does the design of IP and TCP make it *hard* to violate the e2e argument?
  - E.g., middlebox functionality like NATs, firewalls, proxies
- Should the e2e argument apply to routing?
For Next Class…

- Read P&D Chapter 3

- Read and review Cerf & Kahn `74
  - Submit review in HotCRP

- Keep thinking about term project ideas/groups
  - Suggestions available by the end of this week