The network core

- mesh of interconnected routers
- packet-switching: hosts break application-layer messages into packets
  - forward packets from one router to the next, across links on path from source to destination
  - each packet transmitted at full link capacity
Packet-switching: store-and-forward

- takes $L/R$ seconds to transmit (push out) $L$-bit packet into link at $R$ bps
- **store and forward**: entire packet must arrive at router before it can be transmitted on next link
- end-end delay = $2L/R$ (assuming zero propagation delay)

**one-hop numerical example:**
- $L = 7.5$ Mbits
- $R = 1.5$ Mbps
- one-hop transmission delay = 5 sec

more on delay shortly …
Packet Switching: queueing delay, loss

queueing and loss:

- If arrival rate (in bits) to link exceeds transmission rate of link for a period of time:
  - packets will queue, wait to be transmitted on link
  - packets can be dropped (lost) if memory (buffer) fills up
Two key network-core functions

**routing**: determines source-destination route taken by packets
- **routing algorithms**

**forwarding**: move packets from router’s input to appropriate router output

---

<table>
<thead>
<tr>
<th>header value</th>
<th>output link</th>
</tr>
</thead>
<tbody>
<tr>
<td>0100</td>
<td>3</td>
</tr>
<tr>
<td>0101</td>
<td>2</td>
</tr>
<tr>
<td>0111</td>
<td>2</td>
</tr>
<tr>
<td>1001</td>
<td>1</td>
</tr>
</tbody>
</table>
Chapter 1: roadmap

1.1 what is the Internet?
1.2 network edge
   - end systems, access networks, links
1.3 network core
   - packet switching, circuit switching, network structure
1.4 delay, loss, throughput in networks
1.5 protocol layers, service models
1.6 networks under attack: security
1.7 history
How do loss and delay occur?

packets *queue* in router buffers

- packet arrival rate to link (temporarily) exceeds output link capacity
- packets queue, wait for turn

packet being transmitted *(delay)*

free (available) buffers: arriving packets dropped *(loss)* if no free buffers
Four sources of packet delay

\[ d_{\text{nodal}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}} \]

\( d_{\text{proc}} \): nodal processing
- check bit errors
- determine output link
- typically < msec

\( d_{\text{queue}} \): queueing delay
- time waiting at output link for transmission
- depends on congestion level of router
Four sources of packet delay

\[ d_{\text{nodal}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}} \]

\( d_{\text{trans}} \): transmission delay:
- \( L \): packet length (bits)
- \( R \): link bandwidth (bps)
- \( d_{\text{trans}} = \frac{L}{R} \)

\( d_{\text{prop}} \): propagation delay:
- \( d \): length of physical link
- \( s \): propagation speed in medium (~2x10^8 m/sec)
- \( d_{\text{prop}} = \frac{d}{s} \)

* Check out the Java applet for an interactive animation on trans vs. prop delay
Caravan analogy

- cars “propagate” at 100 km/hr
- toll booth takes 12 sec to service car (bit transmission time)
- car~bit; caravan ~ packet
- Q: How long until caravan is lined up before 2nd toll booth?

- time to “push” entire caravan through toll booth onto highway = 12*10 = 120 sec
- time for last car to propagate from 1st to 2nd toll both: 100km/(100km/hr) = 1 hr
- A: 62 minutes
Caravan analogy (more)

- suppose cars now “propagate” at 1000 km/hr
- and suppose toll booth now takes one min to service a car
- **Q:** Will cars arrive to 2nd booth before all cars serviced at first booth?
  - **A:** Yes! after 7 min, 1st car arrives at second booth; three cars still at 1st booth.
Queueing delay (revisited)

- $R$: link bandwidth (bps)
- $L$: packet length (bits)
- $a$: average packet arrival rate
- $La/R \sim 0$: avg. queueing delay small
- $La/R \rightarrow 1$: avg. queueing delay large
- $La/R > 1$: more “work” arriving than can be serviced, average delay infinite!

* Check out the Java applet for an interactive animation on queuing and loss
“Real” Internet delays and routes

- what do “real” Internet delay & loss look like?
- traceroute program: provides delay measurement from source to router along end-end Internet path towards destination. For all $i$:
  - sends three packets that will reach router $i$ on path towards destination
  - router $i$ will return packets to sender
  - sender times interval between transmission and reply.
"Real" Internet delays, routes

traceroute: gaia.cs.umass.edu to www.eurecom.fr

3 delay measurements from gaia.cs.umass.edu to cs-gw.cs.umass.edu

trans-oceanic link

Do some traceroutes from exotic countries at www.traceroute.org
Packet loss

- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving to full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all

* Check out the Java applet for an interactive animation on queuing and loss
**Throughput**

- **throughput**: rate (bits/time unit) at which bits transferred between sender/receiver
  - **instantaneous**: rate at given point in time
  - **average**: rate over longer period of time

server sends bits (fluid) into pipe

pipe that can carry fluid at rate $R_S$ bits/sec

pipe that can carry fluid at rate $R_C$ bits/sec
Throughput (more)

- $R_s < R_c$ What is average end-end throughput?

- $R_s > R_c$ What is average end-end throughput?

*bottleneck link*

link on end-end path that constrains end-end throughput
Throughput: Internet scenario

- per-connection end-end throughput: \( \min(R_c, R_s, R/10) \)
- in practice: \( R_c \) or \( R_s \) is often bottleneck

10 connections (fairly) share backbone bottleneck link \( R \) bits/sec
Chapter 1: roadmap

1.1 what is the Internet?

1.2 network edge

- end systems, access networks, links

1.3 network core

- packet switching, circuit switching, network structure

1.4 delay, loss, throughput in networks

1.5 protocol layers, service models

1.6 networks under attack: security

1.7 history
Protocol “layers”

Networks are complex, with many “pieces”:

- hosts
- routers
- links of various media
- applications
- protocols
- hardware, software

Question:

is there any hope of organizing structure of network?

…. or at least our discussion of networks?
Organization of air travel

- ticket (purchase)
- baggage (check)
- gates (load)
- runway takeoff
- airplane routing

- ticket (complain)
- baggage (claim)
- gates (unload)
- runway landing
- airplane routing

❖ a series of steps
# Layering of airline functionality

<table>
<thead>
<tr>
<th>Layer</th>
<th>Action</th>
<th>Layer</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ticket (purchase)</td>
<td>ticket (complain)</td>
<td>ticket</td>
<td></td>
</tr>
<tr>
<td>baggage (check)</td>
<td>baggage (claim)</td>
<td>baggage (claim)</td>
<td></td>
</tr>
<tr>
<td>gates (load)</td>
<td>gates (unload)</td>
<td>gate</td>
<td></td>
</tr>
<tr>
<td>runway (takeoff)</td>
<td>runway (land)</td>
<td>takeoff/landing</td>
<td></td>
</tr>
<tr>
<td>airplane routing</td>
<td>airplane routing</td>
<td>airplane routing</td>
<td>airplane routing</td>
</tr>
</tbody>
</table>

**Layers:** each layer implements a service

- via its own internal-layer actions
- relying on services provided by layer below
Why layering?

dealing with complex systems:

- explicit structure allows identification, relationship of complex system’s pieces
  - layered *reference model* for discussion

- modularization eases maintenance, updating of system
  - change of implementation of layer’s service transparent to rest of system
  - e.g., change in gate procedure doesn’t affect rest of system

- layering considered harmful?
Internet protocol stack

- **application**: supporting network applications
  - FTP, SMTP, HTTP
- **transport**: process-process data transfer
  - TCP, UDP
- **network**: routing of datagrams from source to destination
  - IP, routing protocols
- **link**: data transfer between neighboring network elements
  - Ethernet, 802.111 (WiFi), PPP
- **physical**: bits “on the wire”
ISO/OSI reference model

- **presentation**: allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- **session**: synchronization, checkpointing, recovery of data exchange
- Internet stack “missing” these layers!
  - these services, *if needed*, must be implemented in application
  - needed?
Encapsulation

message
segment
datagram
frame

source

application
transport
network
link
physical

destination

application
transport
network
link
physical

link
physical

network
link
physical

switch

router

Introduction 1-26
The hourglass shape of the Internet protocol stack

Applications
- Application-layer protocols
  - HTTP
  - SMTP
  - RTP
- Transport layer
  - TCP
  - UDP
- Network layer
  - IPv4
- Link layer
  - PPP
  - Ethernet
- Physical layer
  - Optical Fiber
  - Twisted Pair
  - Coaxial Cable
Chapter 1: roadmap

1.1 what is the Internet?
1.2 network edge
   ▪ end systems, access networks, links
1.3 network core
   ▪ packet switching, circuit switching, network structure
1.4 delay, loss, throughput in networks
1.5 protocol layers, service models
1.6 networks under attack: security
1.7 history
Network security

- field of network security:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks

- Internet not originally designed with (much) security in mind
  - original vision: “a group of mutually trusting users attached to a transparent network” 😊
  - Internet protocol designers playing “catch-up”
  - security considerations in all layers!
Bad guys: put malware into hosts via Internet

- malware can get in host from:
  - **virus**: self-replicating infection by receiving/executing object (e.g., e-mail attachment)
  - **worm**: self-replicating infection by passively receiving object that gets itself executed

- **spyware malware** can record keystrokes, web sites visited, upload info to collection site

- infected host can be enrolled in **botnet**, used for spam, DDoS attacks
Bad guys: attack server, network infrastructure

Denial of Service (DoS): attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

1. select target
2. break into hosts around the network (see botnet)
3. send packets to target from compromised hosts
Bad guys can sniff packets

**packet “sniffing”:**

- broadcast media (shared ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by

- wireshark software used for end-of-chapter labs is a (free) packet-sniffer
Bad guys can use fake addresses

**IP spoofing:** send packet with false source address

… lots more on security (throughout, Chapter 8)
Chapter 1: roadmap

1.1 what is the Internet?
1.2 network edge
   ▪ end systems, access networks, links
1.3 network core
   ▪ packet switching, circuit switching, network structure
1.4 delay, loss, throughput in networks
1.5 protocol layers, service models
1.6 networks under attack: security
1.7 history
Internet history

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational

- 1972:
  - ARPAnet public demo
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes
Internet history

1972-1980: Internetworking, new and proprietary nets

- 1970: ALOHAnet satellite network in Hawaii
- 1974: Cerf and Kahn - architecture for interconnecting networks
- 1976: Ethernet at Xerox PARC
- late 70’s: proprietary architectures: DECnet, SNA, XNA
- late 70’s: switching fixed length packets (ATM precursor)
- 1979: ARPAnet has 200 nodes

Cerf and Kahn’s internetworking principles:
- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control

define today’s Internet architecture
1980-1990: new protocols, a proliferation of networks

- 1983: deployment of TCP/IP
- 1982: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1988: TCP congestion control
- new national networks: Csnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks
Internet history

1990, 2000’s: commercialization, the Web, new apps

- early 1990’s: ARPAnet decommissioned
- early 1990s: Web
  - hypertext [Bush 1945, Nelson 1960’s]
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
- late 1990’s – 2000’s:
  - more killer apps: instant messaging, P2P file sharing
  - network security to forefront
  - est. 50 million host, 100 million+ users
  - backbone links running at Gbps

late 1990’s – 2000’s:

- commercialization of the Web
Internet history

2005-present

- ~750 million hosts
  - Smartphones and tablets
- Aggressive deployment of broadband access
- Increasing ubiquity of high-speed wireless access
- Emergence of online social networks:
  - Facebook: soon one billion users
- Service providers (Google, Microsoft) create their own networks
  - Bypass Internet, providing “instantaneous” access to search, email, etc.
- E-commerce, universities, enterprises running their services in “cloud” (eg, Amazon EC2)
Introduction: summary

covered a “ton” of material!

- Internet overview
- what’s a protocol?
- network edge, core, access network
  - packet-switching versus circuit-switching
  - Internet structure
- performance: loss, delay, throughput
- layering, service models
- security
- history

you now have:

- context, overview, “feel” of networking
- more depth, detail to follow!