I must create a system, or be enslav’d by another Man’s; I will not reason and compare: my business is to create.

--- William Blake

Acknowledgement: this lecture is partially based on the slides of Dr. Larry Peterson, Dr. James Kurose, and Dr. Keith Ross
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (Inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Building Blocks

- **Nodes**: PC, special-purpose hardware...
  - hosts
  - switches

- **Links**: coax cable, optical fiber...
  - point-to-point
  - multiple access
Switched Networks

- A network can be defined recursively as ...
  - two or more nodes connected by a link, or
  - two or more networks connected by a node
Addressing and Routing

- Address: byte-string that identifies a node
  - usually unique

- Routing: process of forwarding messages to the destination node based on its address

- Types of addresses
  - unicast: node-specific
  - broadcast: all nodes on the network
  - multicast: some subset of nodes on the network
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Switching strategies

- Circuit switching: carry bit streams
  - original telephone network

- Packet switching: store-and-forward messages
  - Internet
Circuit Switching

End-end resources reserved for “call”

- link bandwidth, switch capacity
- dedicated resources: no sharing
- circuit-like (guaranteed) performance
- call setup required
Circuit Switching: piece-wise resource allocation

Network resources (e.g., bandwidth) divided into “pieces”

- Pieces allocated to calls
- Resource piece *idle* if not used by owning call *(no sharing)*

- Dividing link bandwidth into “pieces”
  - frequency division
  - time division
Circuit Switching: FDM and TDM

FDM

(time)

4 users

(S)TDM

Example:
Numerical example

- How long does it take to send a file of 640,000 bits from host A to host B over a circuit-switched network?
  - All links are 1.536 Mbps
  - Each link uses TDM with 24 slots/sec
  - 500 msec to establish end-to-end circuit

Let’s work it out!
Packet Switching

each end-end data stream divided into *packets*

- user A, B packets *share* network resources
- each packet uses full link bandwidth
- resources used *as needed*
- store and forward: packets move one hop at a time
  - Node receives complete packet before forwarding

resource contention:

- aggregate resource demand can exceed amount available
- congestion: packets queue, wait for link use

Bandwidth division into "pieces"

Dedicated allocation

Resource reservation
Packet Switching: Statistical Multiplexing

Sequence of A & B packets does not have fixed pattern, shared on demand ➔ statistical multiplexing.

TDM (as in circuit switching): each host gets same slot in revolving TDM frame.
Packet-switching: store-and-forward

- Entire packet must arrive at router before it can be transmitted on next link: store and forward

- Takes L/R seconds to transmit packet of L bits on to link of R bps

- Entire packet must arrive at router before it can be transmitted on next link: store and forward

- delay = \( \frac{3L}{R} \) (assuming zero propagation delay)

**Example:**
- \( L = 7.5 \) Mbits
- \( R = 1.5 \) Mbps
- delay = 15 sec

more on delay shortly ...
Packet switching vs. circuit switching

Packet switching allows more users to use network!

- 1 Mb/s link

- each user:
  - 100 kb/s when “active”
  - active 10% of time

- circuit-switching:
  - 10 users

- packet switching:
  - with 35 users, probability of > 10 active less than 0.0004

Q: how did we get value 0.0004?
Packet switching vs. circuit switching

Is packet switching a “slam dunk winner?”

- Great for bursty data
  - resource sharing
  - simpler, no call setup

- Excessive congestion: packet delay and loss
  - protocols needed for reliable data transfer, congestion control

- Q: How to provide circuit-like behavior?
  - bandwidth guarantees needed for audio/video apps
  - still an unsolved problem

Q: human analogies of reserved resources (circuit switching) versus on-demand allocation (packet-switching)?
Multiplexing

- (Synchronous) Time-Division Multiplexing (<S>TDM)
- Frequency-Division Multiplexing (FDM)
Statistical Multiplexing

- On-demand time-division
- Schedule link on a per- *packet* basis
- Packets from different sources interleaved on link

- Buffer packets that are *contending* for the link
- Buffer (queue) overflow is called *congestion*
Connectivity & resource sharing:
Internet as an example

- network edge:
  applications and hosts

- network core:
  - routers
  - network of networks

- access networks, physical media: communication links
The network edge

- **end systems (hosts):**
  - run application programs
  - e.g. Web, email
  - at “edge of network”

- **client/server model**
  - client host requests, receives service from always-on server
  - e.g. Web browser/server; email client/server

- **peer-peer model**
  - minimal (or no) use of dedicated servers
  - e.g. Skype, BitTorrent, KaZaA
The Network Core

- mesh of interconnected routers

- Answers the fundamental question: how is data transferred through net?
  - circuit switching: dedicated circuit per call: telephone net
  - packet-switching: data sent thru net in discrete “chunks”
**Internet structure: network of networks**

- roughly hierarchical

- at center: “tier-1” ISPs (e.g., MCI, Sprint, AT&T, Cable and Wireless), national/international coverage
  - treat each other as equals

Tier-1 providers interconnect (peer) privately

Tier-1 providers also interconnect at public network access points (NAPs)
Tier-1 ISP: Sprint as an example

Sprint US backbone network

POP: point-of-presence

- to/from backbone
- peering
- to/from customers

- Seattle
- Tacoma
- Stockton
- San Jose
- Anaheim
- Fort Worth
- Orlando
- Kansas City
- Cheyenne
- New York
- Pennsauken
- Relay
- Wash. DC
- Tacoma
- Atlanta
- Orlando

Link types:
- DS3 (45 Mbps)
- OC3 (155 Mbps)
- OC12 (622 Mbps)
- OC48 (2.4 Gbps)
Internet structure: network of networks

- “Tier-2” ISPs: smaller (often regional) ISPs
  - Connect to one or more tier-1 ISPs, possibly other tier-2 ISPs

Tier-2 ISP pays tier-1 ISP for connectivity to rest of Internet
- tier-2 ISP is customer of tier-1 provider

Tier-2 ISPs also peer privately with each other, interconnect at NAP
Internet structure: network of networks

- “Tier-3” ISPs and local ISPs
  - last hop (“access”) network (closest to end systems)
Internet structure: network of networks

- a packet passes through many networks!
Access networks

Q: How to connect end systems to edge router?

- residential access nets
- institutional access networks (school, company)
- mobile access networks

Keep in mind:

- bandwidth (bits per second) of access network?
- shared or dedicated?
Residential access: point to point access

- **Dialup via modem**
  - up to 56Kbps direct access to router (often less)
  - Can’t surf and phone at same time: can’t be “always on”

- **ADSL**: asymmetric digital subscriber line
  - up to 1 Mbps upstream (today typically < 256 kbps)
  - up to 8 Mbps downstream (today typically < 1 Mbps)
  - **FDM**: 50 kHz - 1 MHz for downstream
    - 4 kHz - 50 kHz for upstream
    - 0 kHz - 4 kHz for ordinary telephone
  - can surf and phone at the same time
Residential access: cable modems

- **HFC: hybrid fiber coax**
  - asymmetric: 2 Mbps upstream
  - up to 30Mbps downstream

- **network** of cable and fiber attaches homes to ISP router
  - homes share access to router

- deployment: available via cable TV companies
Residential access: cable modems

Diagram: http://www.cabledatacomnews.com/cmic/diagram.html
Cable Network Architecture: Overview

Typically 500 to 5,000 homes
Cable Network Architecture: Overview

server(s)
cable headend
cable distribution network
home
Cable Network Architecture: Overview

- Cable headend
- Cable distribution network (simplified)
- Home Environment
  - Set-Top Box
  - TV
  - Cable Modem
  - PC
Cable Network Architecture: Overview

FDM:

Channels

1 2 3 4 5 6 7 8 9

Cable headend

Cable distribution network

Home
Company access: local area networks

- company/univ local area network (LAN) connects end system to edge router

- Ethernet:
  - shared or dedicated link connects end system and router
  - 10 Mbs, 100Mbps, Gigabit Ethernet

- LANs: chapter 2
Wireless access networks

- shared *wireless* access network connects end system to router
  - via base station aka “access point”

- wireless LANs:
  - 802.11b/g (WiFi): 11 or 54 Mbps

- wider-area wireless access
  - provided by telcom carriers
  - 3G: ~ 384 kbps
  - GPRS in Europe/US
Home networks

Typical home network components:

- ADSL or cable modem
- router/firewall/NAT
- Ethernet
- wireless access point

![Diagram of home network components](image-url)
Physical Media

- **bit**: propagates between transmitter-receiver pairs
- **physical link**: what lies between transmitter & receiver

- **guided media**:
  - signals propagate in solid media: copper, coax, fiber

- **unguided media**:
  - signals propagate freely, e.g., radio
Physical Media: twisted pair (TP)

- two insulated copper wires
  - Category 3: traditional phone wires, 10 Mbps Ethernet
  - Category 5: 100Mbps Ethernet
Physical Media: coax, fiber

Coaxial cable:
- two concentric copper conductors
- Bidirectional

  baseband:
  - single channel on cable
  - legacy Ethernet

  broadband:
  - multiple channels on cable
  - HFC

Fiber optic cable:
- glass fiber carrying light pulses, each pulse a bit

  high-speed operation:
  - high-speed point-to-point transmission (e.g., 10’s-100’s Gbps)

  low error rate:
  - repeaters spaced far apart
  - immune to electromagnetic noise
Physical media: radio

- signal carried in electromagnetic spectrum
- no physical “wire”
- Bidirectional (with possible asymmetry in reliability)
- propagation environment effects:
  - reflection (=> multi-path)
  - obstruction by objects
  - interference

Radio link types:

- Terrestrial microwave
  - LAN (e.g., Wifi)
    - 11Mbps, 54 Mbps
  - wide-area (e.g., cellular)
    - e.g. 3G: hundreds of kbps
- Satellite
  - Kbps to 45Mbps channel (or multiple smaller channels)
  - Up to 280 msec end-end delay
  - geosynchronous vs. low altitude
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (Inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Inter-Process Communication

- Turn host-to-host connectivity into process-to-process communication.
- Fill gap between what applications expect and what the underlying technology provides.
IPC Abstractions

- Request/Reply
  - distributed file systems
  - digital libraries (web)

- Stream-Based
  - video: sequence of frames
    - \(1/4\) NTSC = 352x240 pixels
      - \((352 \times 240 \times 24)/8=247.5\)KB
    - 30 fps \(\approx 7500\)KBps = 60Mbps
  - video applications
    - on-demand video
    - video conferencing
Reliability: what may go wrong in the Network?

- Bit-level errors (electrical interference)
- Packet-level errors (congestion and thus packet drop)
- Link and node failures
- Packets are delayed
- Packets are delivered out-of-order
- Third parties eavesdrop
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Architecture?

Networks are complex!

- 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?
Layering

- Use abstractions to hide complexity
- Abstraction naturally lead to layering
- Alternative abstractions at each layer

<table>
<thead>
<tr>
<th>Application programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request/reply channel</td>
</tr>
<tr>
<td>Host-to-host connectivity</td>
</tr>
<tr>
<td>Hardware</td>
</tr>
</tbody>
</table>
Protocols

- Building blocks of a network architecture

- Term “protocol” is overloaded
  - specification of peer-to-peer interface
  - module that implements this interface
What’s a protocol?

**human protocols:**

- “what’s the time?”
- “I have a question”
- introductions

... specific msgs sent

... specific actions taken when msgs received, or other events

**network protocols:**

- machines rather than humans
- all communication activity in Internet governed by protocols

protocols define format, order of msgs sent and received among network entities, and actions taken on msg transmission, receipt
What’s a protocol?

A human protocol and a computer network protocol:

Q: Other human protocols?
Interfaces

- Each protocol object has two different interfaces
  - peer-to-peer interface: messages exchanged with peer
  - service interface: operations on this protocol
Recipe Machinery

- **Protocol Graph**
  - most peer-to-peer communication is indirect
  - peer-to-peer is direct only at hardware level
Protocol machinery (contd.)

- Multiplexing and Demultiplexing (demux key)
- Encapsulation (header/body)
Internet Architecture

- Defined by Internet Engineering Task Force (IETF)
- Hourglass Design
- Application vs Application Protocol (FTP, HTTP)

Internet protocol graph

Alternative view of the Internet architecture
ISO Architecture

One or more nodes within the network
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Socket API: create a socket

- Creating a socket

  int socket(int domain, int type, int protocol)
  
  - domain = PF_INET, PF_UNIX
  - type = SOCK_STREAM, SOCK_DGRAM, SOCK_RAW
  - protocol: not used in most cases
Socket API: open a socket

- Active Open (on client)
  
  ```c
  int connect(int socket, struct sockaddr *addr,
              int addr_len)
  ```

- Passive Open (on server)
  
  ```c
  int bind(int socket, struct sockaddr *addr, int addr_len)
  int listen(int socket, int backlog)
  int accept(int socket, struct sockaddr *addr, int addr_len)
  ```
Socket API: send/receive messages

int send(int socket, char *msg, int mlen, int flags)

int recv(int socket, char *buf, int blen, int flags)
Protocol implementation issues

- **Process Model**
  - Key issue: to avoid context switches

- **Buffer Model**
  - Key issue: to avoid data copies
Process model

(a) process-per-protocol: context switch

(b) process-per-message: procedure call
Buffer model

Between application and network subsystems

With in network subsystem
A reference

Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
Performance Metrics

- **Bandwidth**
  - data transmitted per time unit
  - link vs. end-to-end
  - Notation: $KB = 2^{10}$ bytes, $Mbps = 10^6$ bits per second

  - Throughput: measured bandwidth

- **Latency (delay)**
  - time to send message from point A to point B
  - one-way vs. round-trip time (RTT)
  - components
    - Latency = Propagation + Transmit + Queue
    - Propagation = Distance / c
    - Transmit = Size / Bandwidth
User perceived performance (delay in service): Bandwidth vs. Latency (prop. + queuing)

- Relative importance depends on applications
  - 1-byte data: 1ms vs. 100ms dominates 1Mbps vs. 100Mbps
    - e.g., interactive remote command line
  - 25MB data: 1Mbps vs. 100Mbps dominates 1ms vs. 100ms
    - e.g., file download

- “Infinite” bandwidth (compared with data size)
  - RTT dominates (note: RTT only accounts for propagation delay in this case)
    - Throughput = TransferSize / TransferTime
    - TransferTime = RTT + 1/Bandwidth x TransferSize
  - E.g., “1-MB file to 1-Gbps link” as “1-KB packet to 1-Mbps link” in overall delay
Delay x Bandwidth Product

- Amount of data “in flight” or “in the pipe”
- Usually, we are interested in using RTT as “delay” instead of one-way delay
  - E.g., congestion control in TCP
- Example: 100ms x 45Mbps = 560KB
- Affects higher-layer design
  - E.g., congestion control in satellite networks where RTT is large compared with wired networks
Outline

- Requirements
  - Connectivity
  - Cost-effective resource sharing
  - Support for common services (inter-Process Communication)

- Network Architecture

- Implementation Issues

- Performance Metrics

- Discussion
How about emerging networks such as Sensornets?

- Limited bandwidth and highly dynamic link/network properties

- Indoor testbed at OSU; 3 feet node separation
- 300 data points for each distance, with each data point representing the status of 100 broadcast transmissions
Emerging networks: link properties, etc.

- Jerry Zhao, and Ramesh Govindan, *Understanding Packet Delivery Performance In Dense Wireless Sensor Networks*, ACM SenSys’03


Emerging networks: architecture & protocol design

- Dynamic wireless channels, resource constraints, and application diversity challenges network design in terms of both architecture and algorithmic design
Emerging networks: system software design

- Event-driven system architecture for concurrency and energy efficiency
  

  on the design philosophy of TinyOS
Further readings

- Network architecture: Internet
  - D. Clark, *The design philosophy of the DARPA Internet protocols*, ACM SIGCOMM’88
Ongoing research programs

- USA: National Science Foundation (NSF)
  - Global Environment for Networking Innovations (GENI)
  - Network Science and Engineering (NetSE)

- Europe: Future Internet Research and Experimentation (FIRE)
  - by European Commission

- Similar efforts are pursued in other countries
Summary

- Connectivity and cost-effective resource sharing
- Inter-Process Communication
- Network Architecture
- Implementation Issues
- Performance Metrics
- Discussion
  - Emerging technologies
First computer network application?

E-MAIL
Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent

- a.k.a. “mail reader”
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Mozilla Thunderbird
- outgoing, incoming messages stored on server
Electronic Mail: mail servers

Mail Servers

- **mailbox** contains incoming messages for user
- **message queue** of outgoing (to be sent) mail messages
- **SMTP protocol** between mail servers to send email messages
  - client: sending mail server
  - “server”: receiving mail server

![Diagram of mail servers and user agents using SMTP protocol]
Electronic Mail: SMTP [RFC 2821]

- uses TCP to reliably transfer email message from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
Scenario: Alice sends message to Bob

1) Alice uses UA to compose message and “to”
   bob@someschool.edu

2) Alice’s UA sends message to her mail server; message placed in message queue

3) Client side of SMTP opens TCP connection with Bob’s mail server

4) SMTP client sends Alice’s message over the TCP connection

5) Bob’s mail server places the message in Bob’s mailbox

6) Bob invokes his user agent to read message
Sample SMTP interaction: right after TCP connection is setup between client and server

SMTP Client (C): crepes.fr
SMTP Server (S): hamburger.edu

S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
Try SMTP interaction for yourself:

- `telnet servername 25`
  - E.g., `telnet www.yahoo.com 25`

- see 220 reply from server

- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands

above lets you send email without using email client (reader)
Mail access protocols

- SMTP: delivery/storage to receiver’s server
- Mail access protocol: retrieval from server
  - POP: Post Office Protocol [RFC 1939]
    - authorization (agent <-->server) and download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - more features (more complex)
    - manipulation of stored msgs on server
  - HTTP: gmail, Hotmail, Yahoo! Mail, etc.
Our focus:
to build networks that support application-level communication

And more complex scenarios …
Assignment - Chapter 1

Exercise#0:
- Exercises 3, 5, 13, and 26
  - Hint: 1) focus on the precise definition of bandwidth and delay; 2) for Ex. 28: image consists of pixels, with each pixel represented by certain # of bits (e.g., 8)
- Exercise 10
  - Hint: identify scenarios where STDM and FDM work well and do not work well respectively
- Exercises 36 and 37
  - for both exercises (i.e., ping and traceroute), use command “script” to record the screen printout and submit it
  - May use www.cs.wayne.edu and www.wayne.edu, if you cannot ping/traceroute

TinyExam#0
TinyExam questions

Type 1: Test your understanding of basic concepts, protocols, etc

- An example
  
  In implementing network protocols, what is the major drawback of the “process-per-protocol” model?

- How to do well?
  
  - Understand lectures, read relevant materials from the textbook
  
  - Ask questions, if any, well before the exam
  
  - Work on exercise questions
TinyExam questions

Type 2: Problem solving

- An example
  Calculate the total time required to transfer a 2000-KB file in the following cases, assuming a RTT of 100ms, a packet size of 1KB data, and an initial 2×RTT of “handshaking” before data is sent:
  (a) The bandwidth is 1.5Mbps, and data packets can be sent continuously.
  (b) The bandwidth is 1.5Mbps, but after we finish sending each data packet we must wait for one RTT before sending the next.

- How to do well
  - Work on exercise questions
  - Understand the fundamentals of relevant networking concepts, protocols, techniques, etc