Instructors & GTAs

- Instructors:
  Marc Abrams <abrams@cs.vt.edu>
  Srinidhi Varadarajan <srinidhi@cs.vt.edu>

- GTAs:
  Batul Mirza <bmirza@csgrad.cs.vt.edu>
  Abhijit Khobare <akhobare@csgrad.cs.vt.edu>
  GTAs available ...
  - Monday-Friday
  - 4:00-6:00 pm
  - In 128 McBryde
Teaching Method

- Team teaching:
  - Course topics split between Profs.
  - 1 Professor gives same lecture at both 9:00 & 1:00
  - Same projects, homeworks in both classes
Textbook

Peterson & Davie

Computer Networks: A Systems Approach

©2000
CS/ECPE 5516: Assignments

- ~5 Homeworks
- Term Project
  - Gives "hands-on" protocol experience
    Part 1: Implement reliable data-link protocol (due after spring break)
    Part 2: Add routing (due on last class meeting)
  - Done in teams
  - Teams have programming & non-programming roles
- Exams
  - Midterm exam
  - Final exam
Class Communication

- **Class WWW site:**
  - http://courses.cs.vt.edu/~cs5516/
  - These lecture notes available on Web

- **Class Listserv**
  - cs5516@courses.cs.vt.edu
  - You’re automatically on listerv
CS/ECPE 5516: Objectives (1)

- Understand network architectures, algorithms, and protocols

- Understand the underlying principles of computer networks
  - Link and multiaccess protocols
  - Packet and cell switching
  - Internetworking and routing
  - End-to-end protocols
  - End-to-end data issues
  - Congestion control
  - High-speed networking
CS/ECPE 5516: Objectives (2)

Understand widely-used and emerging networking technologies and protocols

- Ethernet (IEEE 802.3)
- Token Rings (FDDI and IEEE 802.5)
- Wireless LANs (IEEE 802.11)
- Asynchronous Transfer Mode (ATM)
- TCP/IP protocol suite + IP version 6
CS/ECPE 5516: Topics

Chapter 1: Foundation
Chapter 2: Direct Link Networks
Chapter 3: Packet & cell switching
Chapter 4: Internetworking and routing
Chapter 5: End-to-end protocols
Chapter 6: Congestion control
Chapter 7: End-to-end data issues
Chapter 8: Security
Graduate Networking Courses
Graduate Networking Courses

CS/ECPE 5516 (NET-1)
- Networks
- Protocols
- Algorithms
Graduate Networking Courses

- CS/ECPE 5516 (NET-1)
  - Networks
  - Protocols
  - Algorithms

- ECPE 5984 (NET-2)
  - Network analysis
  - Network design
  - Network management
Graduate Networking Courses

- **CS/ECPE 5516 (NET-1)**
  - Networks
  - Protocols
  - Algorithms

- **ECPE 5984 (NET-2)**
  - Network analysis
  - Network design
  - Network management

- **ECPE 6504 (NET-3)**
  - Advanced topic(s)
    - Emerging technology
    - Applied and/or theoretical
Prerequisites for This Course (1)

A high-level language programming course

- A lower-division undergraduate programming course is adequate

- You must be able to...
  - Read & understand algorithms in code
  - Appreciate issues related to software implementation
  - Write C code, or team with other students who are C programmers for projects
Prerequisites for This Course (2)

A computer organization course

- Need to understand
  - Basic principles of computer structures
  - Data formats
  - Hardware-software interaction

- ECPE 2504, CS/ECPE 4504, or the GPIT foundation course, "Fundamentals of Computer Engineering" satisfy this prerequisite
Additional Prerequisite for NET-2, NET-3

- Calculus-based probability & stat course
  - Need to understand
    - Continuous & discrete probability distributions
    - Probability density & distribution functions
    - Mean, variance, standard deviation, and moments of probability distributions
  - STAT 4714 or equivalent

- And, NET-1 -> NET-2 and NET-2 -> NET-3
History & motivation

Network architecture (section 2.2)
- Layered models
- Definitions and abstractions
- OSI Reference Model

Network design issues (section 2.1)
- Definitions
- Components
- Message, packet, and cell switching
- Resource sharing
- Functionality
- Performance
History & Motivation
What's a *Network*?

- Provides connectivity between computers
- Based on *protocols*:
  Rules for computers to exchange data
Landmarks in Networking

- First description of Arpanet: early 1960’s, for defense
- First packet-switched network: Europe
- First Internetwork: US
- First LAN: Bell Labs building
- First OS to include networking: UNIX + UUCP
- Cataclysmic event leading to invention of Ethernet: insomnia
Networking is Hot!!!
Networking is Hot!!!

Voice over IP
Networking is Hot!!!
Networking is Hot!!!

- Voice over IP
- 3rd Generation Cellular + WAP Protocols
- QoS for Network Video
Networking is Hot!!!

- Exponential growth in fiber bandwidth
- Third Generation Cellular + WAP Protocols
- Voice over IP
- QoS for Network Video
Networking is Hot!!

- Exponential growth in fiber bandwidth
- Internet via Satellite
- 3rd generation Cellular + WAP Protocols
- Voice over IP
- QoS for Network Video

Exponential growth in fiber bandwidth

Internet via Satellite

3rd generation Cellular + WAP Protocols

Voice over IP

QoS for Network Video
Networking is Hot!!!
Networking is Hot!!!

- Exponential growth in fiber bandwidth
- Internet via Satellite
- 3rd Generation Cellular + WAP Protocols
- Voice over IP
- QoS for Network Video
- Internet phone

... and many others ...

Internet Timeline

http://www.isoc.org/internet-history/brief.html
Growth in # Internet Hosts

http://www.isoc.org/guest/zakon/Internet/History/HIT.html
Growth in # Domains

http://www.isoc.org/guest/zakon/Internet/History/HIT.html
Growth in # WWW Sites

http://www.isoc.org/guest/zakon/Internet/History/HIT.html
NET.WORK.VIRGINIA

- ATM network with Internet access
- Over 400 sites with OC3, DS3, or DS1 service
- Service through Sprint and Vision Alliance (consortium led by Bell Atlantic)

Backbone/Internet Gateway
2.2: Network Architecture
Network Architecture

- Think of network architecture in two terms...
  - Box-diagrams (shows organizations)
  - Processes (shows operation)

- Diagrams use **layered model**
  (Assists in coping with complexity)

- Process description uses **cooperating processes**
A Simple Layered Model

- Decomposes system into simpler, manageable components
- Provides a modular design
A Simple Layered Model

- Decomposes system into simpler, manageable components
- Provides a modular design

Application Programs
A Simple Layered Model

- Decomposes system into simpler, manageable components
- Provides a modular design

Application Programs
Process-to-Process Channels
A Simple Layered Model

- Decomposes system into simpler, manageable components
- Provides a modular design

Application Programs

Process-to-Process Channels

Host-to-Host Connectivity
A Simple Layered Model

- Decomposes system into simpler, manageable components
- Provides a modular design

Application Programs
Process-to-Process Channels
Host-to-Host Connectivity
Networking Hardware
Multiple Abstractions for One Layer

- Application Programs
  - Request/Reply Channel
  - Message Stream Channel
- Host-to-Host Connectivity
- Networking Hardware

■ Process-to-process channel
  - Request/reply interaction
  - Stream of messages
Functions Are Not Always “Layer-able”

- Application Programs
- Process-to-Process Channels
- Host-to-Host Connectivity
- Networking Hardware

- Some functions may need to interact with multiple layers
Layered Models ... Generalized (1)

■ Layer $N$
  - Provides services to layers $N+1$ and above
  - Uses services offered by layers $N-1$ and below
  - May interact with peer layer $N$ entities via protocols

■ Distinction between *service*, *interface*, and *implementation*
Layered Models ... Generalized (2)

services provided to upper layers

service interface

Layer N

Node A

peer-to-peer interface

protocol

Node B

services provided by lower layers
Layered Models ... Generalized (3)

- **Protocols** are rules for *cooperation* between peers
  - Peer-to-peer interfaces, e.g. Protocol X defines the interfaces
  - “Protocol” sometimes used to refer to the layer itself, e.g. the entity that realizes Protocol X

- **Service access points** (SAPs) between layers adhere to interface definition
  - Service or layer-to-layer interface
Interfaces and Protocols

■ Three components of an interface
  ● Visible objects, operations, parameters
  ● Rules governing operation sequences
  ● Encoding and formatting conventions required for operations & parameters
  ● Example:
    HTTP’s peer interface = docs + GET/PUT

■ Protocols are usually restricted to peer layers, whereas interfaces are used in 2 places…
Layered Models ... Generalized (2)

Two kinds of interfaces!

Layer $N$

Node A

service interface

Node B

peer-to-peer interface

protocol
ISO Model

- ISO = International Organization for Standards
- 1980’s: 7-layer
  Open Systems Interconnect (OSI) model
- OSI
  - *protocols* fell out of favor
  - *7-layer model* remains today
- But 7 layer model is a *fiction*:
  Most protocols suites (TCP/IP, ATM) don’t *strictly* obey model
OSI Terminology for Layering

Layer N+1

Layer N

SAP  Service Access Point (where N+1 accesses N)
IDU  Interface Data Unit (passed from N+1 to N)
SDU  Service Data Unit (data from N+1)
ICI  Interface Control Information (service type, etc.)
PDU  Protocol Data Unit (exchanged by peer N entities)
OSI Reference Model
Deviation from Strict Layering

- Example: Fiber Distributed Data Interface (FDDI)

<table>
<thead>
<tr>
<th>Data Link</th>
<th>LLC</th>
<th>SMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physical</td>
<td>MAC</td>
<td>PHY</td>
</tr>
<tr>
<td></td>
<td>PMD</td>
<td></td>
</tr>
</tbody>
</table>

- LLC     Logical Link Control
- MAC     Media Access Control
- PHY     Physical
- PMD     Physical Media Dependent
- SMT     Station Management
Layered Model Example

- Typical protocol “stack” in a UNIX-based TCP/IP environment

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>X, FTP, Telnet</td>
</tr>
<tr>
<td></td>
<td>HTTP, SMTP, NFS</td>
</tr>
<tr>
<td>Presentation</td>
<td>XDR</td>
</tr>
<tr>
<td>Session</td>
<td>RPC</td>
</tr>
<tr>
<td>Transport</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Network</td>
<td>IP</td>
</tr>
<tr>
<td>Data Link</td>
<td>Ethernet, FDDI</td>
</tr>
<tr>
<td>Physical</td>
<td></td>
</tr>
</tbody>
</table>

TCP/IP stack in a UNIX-based environment.
Internet protocols ("TCP/IP") really uses a four-layer architecture
Advantages of Layering (1)

- Data hiding and encapsulation -- data structures, algorithms, etc. in layer aren’t visible to other layers

- Decomposition -- complex systems can be decomposed into simpler pieces

- System can evolve -- you can reimplement a layer (as long as service & interface don’t change)
Advantages of Layering (2)

- Can offer alternate services at layer $N+1$ that share services of layer $N$:

<table>
<thead>
<tr>
<th>UDP</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layer 2</td>
<td></td>
</tr>
</tbody>
</table>

- Alternate implementations of layer $N$ can co-exist:

<table>
<thead>
<tr>
<th>Layer N+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet</td>
</tr>
</tbody>
</table>

- Layer/sublayer can be simplified/omitted if services aren’t needed

- Can test layers separately (Otherwise protocol testing is intractable!)
Disadvantages of Layering

- Some functions (like FDDI station management) really need to access and operate at multiple layers.
- Poorly conceived layers can lead to awkward and complex interfaces.
- There may be performance penalties due to extra overhead of layers, for example memory-to-memory copies.
- Design of (an older) layer $N+1$ may be sub-optimal given the properties of (a new) layer $N$. 
Physical Layer

- Transmits bit sequence over physical media
- Synchronous or asynchronous
- Defines physical interface, signaling, cabling, connectors, etc.
- May depend on data link protocol
  - IEEE 802.3 (Ethernet):
  - 10Base5 (thick wire)
  - 10Base2 (thin wire)
  - 10BaseT (twisted pair)
Data Link Layer

Responsible for error-free transmission of packets between “adjacent” or directly-connected nodes

<table>
<thead>
<tr>
<th>LLC</th>
<th>MAC</th>
</tr>
</thead>
</table>

- **Sublayer: Media access control (MAC)**
  - Allows multiple nodes to share common transmission media
  - Supports addressing of nodes

- **Sublayer: Logical link control (LLC)**
  - Detect, correct errors, ...
Network Layer

- Gets packet through network from source node to destination node
  - Routing over multiple hops (vs. DLL - 1 hop only)
  - Flow control or congestion control
  - Internetworking to allow transmission between different types of networks

- In WAN or internetwork, network layer requires cooperation among peers at intermediate nodes

- Network layer function is minimal in a LAN
Transport Layer (1)

- Provides network-independent, end-to-end message transfer between pairs of ports or sockets

- Ports are destination points for communication that are defined by software
  - Ports are identified by a transport address that identifies (host computer address, port number)
  - Port = application (process) address
  - Established services, like FTP and HTTP, have “well-known” default port numbers (HTTP = port 80)
Transport Layer (2)
Transport Layer (2)

Process A

Network

Process B

Ports (Sockets)
Transport Layer (3)

Transport layers typically provide one of two basic types of service:

- Virtual circuit or connection-oriented service
  - Transmission Control Protocol (TCP)

- Datagram or connection-less service
  - User Datagram Protocol (UDP)
Transport Layer: Virtual Circuits

- **Circuit** = electrical connection from A to B
- **Virtual circuits** = logical connection

Network equivalent of dialing phone number

Connections maintained for many packet or message transmissions until released

- Network layer may still use dynamic routing

**Functions**

- Translate transport address to network address
- Segment messages into packets for transmission
- Pass packets to network layer for delivery
- Reassemble packets at receiving end
Transport Layer: Datagrams

- Datagram communication is connection-less (vs. virtual circuits)
- New connection is established and released for each packet or message transmitted
  - Packet itself establishes and releases connection
- Functions
  - Translate transport address to network address
  - Pass messages to network layer for delivery
  - Each message sent as 1 packet
  - Upper layer responsible for re-ordering and error detection
Session Layer

- Provides name space to tie together different virtual connections (e.g., audio + video)
- Might handle access rights (e.g., use network printer)
- Might multiplex multiple sessions over 1 connection (provided by lower layer)
- Might handle transaction commit & rollback
Preserve semantics (meanings or values) while resolving syntactic (representation) differences

- Ex: 123.5 E+10 might be represented by different bits on two computers

In open systems, heterogeneous computers result in heterogeneous representations

- Characters: ASCII, EBCDIC, Unicode
- Integers: lengths, 1’s versus 2’s complement
- Reals: fixed or floats, different float points
- Byte order: 01234567... or 67452301
- Structured data

Network standard data representation used in Internet to solve problem (XDR)
Presentation Layer (2)

- Might do encryption
- Might do compression
Application Layer

- Network applications make up the application layer
- Protocol specific to each particular application
- Certain applications, like HTTP, NFS, FTP, and Telnet have been standardized
- Standards do not provide a fixed model for applications, but models do exist
  - Client-server versus peer-to-peer
  - Remote procedure call (RPC) versus message passing
2.2: Network Requirements
Network Requirements

*Depend on perspective …*

- Network users
  - Services that a user’s applications need, e.g., latency (delay) and loss rate

- Network designers
  - Cost-effective design e.g., network resources are efficiently utilized and fairly allocated

- Network service providers
  - System that is easy to administer and manage e.g., faults can be easily isolated and it is easy to account for use

*Considerations that drive network design …*
Connectivity (1)

- Network building blocks
  - *Nodes* -- Workstations, PCs
  - *Direct links* -- twisted pair, coaxial cable, optical fiber, radio frequency link, ...
    - Point-to-point
    - Multiple access (multiaccess)
Connectivity (1)

Network building blocks

- Nodes -- Workstations, PCs
- Direct links -- twisted pair, coaxial cable, optical fiber, radio frequency link, ...
  - Point-to-point
  - Multiple access (multiaccess)

Point-to-Point
Connectivity (1)

- Network building blocks
  - *Nodes* -- Workstations, PCs
  - *Direct links* -- twisted pair, coaxial cable, optical fiber, radio frequency link, ...
    - Point-to-point
    - Multiple access (multiaccess)

```
Point-to-Point
```

```
Multiple Access
```
Indirect connectivity

- Switched or routed networks allow indirectly connected nodes to communicate
- Switches, routers, hubs, etc. are specialized nodes in the network
- Switching network is the “cloud”
Indirect connectivity

- Switched or routed networks allow indirectly connected nodes to communicate
- Switches, routers, hubs, etc. are specialized nodes in the network
- Switching network is the “cloud”
Connectivity (3)

- An *internetwork* or *internet* is a network of networks
  - Need internetworking devices
An *internetwork* or *internet* is a network of networks

- Need internetworking devices
Message versus Packet Switching (1)

- Networks may be classified by how they segment data for transmission and switching
  - Message-switched versus packet-switched
  - Most networks use packet switching (or cell switching)

- Messages
  - Have some higher level meaning, e.g. as a request for service or a reply
  - Encoded as a string of bits
Message versus Packet Switching (2)

- Packets
  - Messages may be decomposed into one or more packets for transmission, reconstructed at receiver
  - Lower layer entities may further decompose packets, for example: Ethernet frames, ATM cells
Sessions

- Messages usually occur as part of a longer transaction called a session

- Session properties
  - Message or packet arrival process (rate, variability)
  - Session holding time
  - Message or packet length distribution
  - Acceptable delay
  - Required reliability and security
  - Acceptable ordering of messages or packets
Circuit vs. Store-and-Forward Switching

- Two forms of switching for the messages or packets in a session are widely used
  - *Circuit switching*
  - *Store-and-forward* or, simply, “packet switching”
Circuit Switching

- Session $s$ initiated with a request for a fixed transmission rate (bandwidth requirement) of $r_s$ bits/sec

- Path created through the network
  - Each link in path allocates capacity of $r_s$ bits/sec to $s$, e.g. using time-division multiplexing (TDM) or frequency division multiplexing (FDM)
  - Request is blocked if no path can be established

- Bandwidth dedicated to $s$ for the life of the session
Efficiency of Circuit Switching

- Most data traffic is “bursty,” so links are not well utilized

  ![Time chart showing bursty data traffic]

- Circuit switching not widely used in data networks (except, inefficiently, for access)
  - Links are expensive
  - Sessions require significant portion of link capacity
    (only a few sessions can be supported)
  - Traffic is bursty, so utilization is low
Store-and-Forward Switching (1)

- No transmission rate allocation is dedicated at set-up
  - Differs from circuit switching
- Data transmitted at full link capacity, but links can be shared by multiple sessions on a demand basis
Store-and-Forward Switching (2)

- Advantages:
  - Link fully utilized if there is any data to transmit
  - Delay can be significantly reduced
  - Utilization can be significantly increased

- Disadvantages:
  - Greater variance in delay due to queuing delays
  - Flow control needed to prevent buffer overflows
How is information switched?

- **Message Switching**: messages are sent intact without being broken into packets
- **Packet Switching**: messages are broken into packets for transmission
- **Cell Switching**: messages (or packets) are broken into fixed-size packets called *cells*
How are messages or packets routed through the network?

- **Virtual Circuit Routing**: a path is established and used for the duration of the session
  - Connection-oriented or virtual circuit service
- **Dynamic Routing**: each packet or message may traverse a different path through the network
  - Connection-less or datagram service
Networks may be classified by their geographic extent:

- **DANs, LANs, MANs, and WANs**
- Useful classification for lower level protocols
- Should be transparent to upper layer protocols

**DAN: Desk Area Network**

- Connects PC and peripherals
- USB is sort of an example
- Medium to high data rates
- Low-cost, high-volume, built-in interfaces
Geographic Extent (2)

- Local Area Networks (LANs)
  - Limited extent (10’s of meters to a few kilometers)
  - High data rates (megabits to gigabits per second)
  - Built-in interfaces in workstations, PCs
  - Low cost
  - Low delay
  - Examples: Ethernet, Token Ring, FDDI, ATM
Geographic Extent (3)

■ Metropolitan Area Networks (MANs)
  - Medium extent (10’s of kilometers)
  - Medium data rates (kilobits to 100’s of megabits per second)
  - Special access equipment, often expensive
  - Example: FDDI, ATM

■ Wide Area Networks (WANs)
  - Large extent (global)
  - Low speed (kilobits to 100’s of megabits per second)
  - Special access equipment, usually expensive
  - High latency
  - Examples: T1, T3, SMDS, ATM
Economics dictates that network resources must be shared or *multiplexed* among multiple users

- Shared links
- Shared network nodes (switches, hubs, etc.)
Resource Sharing (2)

- Multiplexing schemes
  - Fixed
    - *Time-division multiplexing* (TDM) or *synchronous time-division multiplexing* (STDM)
    - *Frequency division multiplexing* (FDM)
  - On-demand
    - *Statistical multiplexing*, including asynchronous time-division multiplexing
Statistical Multiplexing

Packets from all traffic streams are merged into a single queue and transmitted on-demand

- Scheduling is typically first-come first-served (FCFS), but priority schemes are also used
- $T_{SM} = L/C$ seconds needed to transmit $L$-bit packet
- May also maintain a separate queue for each traffic stream and service in a “round-robin” manner (skipping over an empty queue with no loss of transmission capacity)
Synchronous Time-Division Multiplexing

- Time on the channel is divided into $m$ slots and each of $m$ traffic streams is given one slot -- unused slots are wasted
  - Create $m$ channels, each with capacity $C/m$
  - $L$-bit packet takes $T_{STDM} = Lm/C$ seconds to transmit if packets are long compared to the length of a slot
  - $L$-bit packet takes $T_{STDM} = L/C$ seconds to transmit if slots are of packet length, but must wait $(m-1)$ slots between transmissions
Frequency Division Multiplexing

- Channel bandwidth $W$ is subdivided into $m$ channels and each of $m$ traffic streams is given one channel
  - Create $m$ channels, each with bandwidth $W/m$, or capacity $C/m$ (ignoring guard bands between channels)
  - $L$-bit packet takes $T_{\text{FDM}}=Lm/C$ seconds to transmit
FDM, STDM vs. Statistical Multiplexing

- Statistical multiplexing has smaller average delay than either STDM or FDM
  - Channel capacity is wasted with STDM (wasted time slot) and FDM (wasted bandwidth) when a traffic stream is idle
  - Transmission time greater for STDM and FDM

- Advantages of STDM or FDM
  - Statistical multiplexing has lower average delay, but higher *variance* of delay
  - STDM and FDM eliminate the need to identify traffic stream associated with each packet
Functionality (1)

Network must support common services or process-to-process channels, for example
- Request/reply channel for file access, digital libraries, etc.
- Message stream channel for video and audio applications
Functionality (2)

What can corrupt this functionality? What can go wrong?

- Link or node failures
- Errors at the bit or packet level
- Arbitrary delays
- Buffer overflows -- lost packets
- Out of order delivery
- Security -- eavesdropping, spoofing, etc.
Functionality (3)

- The key problem is to bridge
  - What the application expects and
  - What the underlying technology can provide

- Carries over to a layered model -- Layer $N$ needs to provide
  - What Layer $N+1$ expects using
  - What Layer $N-1$ can provide
Distributed Algorithms (1)

- Peers must cooperate to perform network functions
- A distributed algorithm is decomposed into one or more local algorithms
- Each local algorithm proceeds based on the data received from other layers or peers, and the order in which the data is received

Diagram: 

Network --- Network --- Network --- Network
Data Link --- Data Link --- Data Link --- Data Link
Physical --- Physical --- Physical --- Physical
Distributed Algorithms (2)

- These algorithms are complex because underlying services may be unreliable

- Data may …
  - Never arrive (due to transmission error, overflow, etc.)
  - Arrive late (due to arbitrary network delay)
  - Arrive out of order (due to differing network paths)

- It may be impossible to ensure correct operation 100% of the time
  - Maximize probability of success
  - Detect errors
Maroon and Orange Armies (1)

- Maroon Armies #1 and #2 must attack simultaneously to defeat the Orange Army.
- Maroon Army #1 wants to send a messenger (😊) to Maroon Army #2 to set a time for the attack.
Maroon and Orange Armies (2)

- The messenger must go through enemy territory (an unreliable communication channel)
- Problems ...
  - May be delayed -- until after the attack time
  - May be captured -- so that message is never delivered
Maroon and Orange Armies (3)

- Possible solution: require Maroon Army #2 to send another messenger to acknowledge that the first messenger arrived with the message
  - Acknowledgment messenger may be delayed or captured
  - Maroon Army #2 would think that the attack is on, but Maroon Army #1 cannot know if it is on or not

- There is no possible solution to the problem with probability 1 of success
Maroon and Orange Armies (4)

- The attack can be synchronized with high probability
  - For example, send many messengers to increase likelihood of one reaching Maroon Army #2
Performance

- Protocols and services define *functionality*, but not *performance*
  - *Bandwidth*, throughput, data rate, capacity, ...
  - *Latency*, delay, ...
  - Variability in latency and data rate important for some applications
  - Loss is sometimes a performance measure

- Performance is determined by
  - Underlying technologies
  - Protocol design
  - Protocol implementation
  - Use by the application or upper layer
Bandwidth

- Bandwidth is commonly used to indicate the amount of data that can be transferred in some unit of time

- Example: 10 megabits per second
  - $10^7$ bits per second
  - $10^{-7}$ seconds per bit (100 ns) -- the “bit width”

- Link versus end-to-end bandwidth may vary
Latency (1)

- Latency is delay, i.e. the time it takes for a message to get from one point to another
- Round-trip time (RTT) is the time it takes to get to one point and receive a return back
- End-to-end versus link delay

Components
- Processing overhead -- e.g., software overhead
- Transmission time -- depends on bandwidth and length of message
- Propagation delay -- time for a bit to travel from one end of a link to another
- Queueing delay -- time waiting for a shared link
Latency (2)

Example

- Processing overhead -- assume 1 $\mu$s
- Transmission time
  - Assume $L = 1,000$ bit message
  - Assume $C = 10$ Mbps link
  - Transmission time: $T = L/C = 100$ $\mu$s
- Propagation delay
  - Speed of light is $c = 2 \times 10^8$ m/s in optical fiber
  - Assume $D = 1$ km (1000 m)
  - Propagation delay = $D/c = 5$ $\mu$s
- Queueing delay -- assume 0
- Latency is $1 + 100 + 5 = 106$ $\mu$s (transmission time dominates in this example)
Latency (3)

■ Dominating factors
  ● Processing overhead can dominate for high data rate links over short distances with short messages
  ● Transmission time can dominate for slower links or longer messages
  ● Propagation delay is important with long links
  ● Queueing delay can dominate in a congested network
The delay × bandwidth product is an important factor in protocol design

- Determines the “size of the pipe”
- Made large by
  - High delay, e.g. long propagation time
  - High bandwidth, e.g. a fast link
- Large product means that a large amount of data must be sent to “fill the pipe” before the receiver can respond
You should now be able to ... (1)

- Define protocol, service access point, protocol data unit, service data unit
- Describe the structure and role of a layers in a network architecture
- Cite advantages and disadvantages of a layered model for a network architecture
- List the seven layers in OSI reference model and describe the basic functions of each layer
- Describe the three different perspectives on network design
You should now be able to … (2)

- Define the basic components of a network including links, nodes, and switches
- Describe the construction of an internet (with a lower case i)
- Distinguish between message, packet, and cell switching
- Distinguish between store-and-forward and circuit switching and cite advantages and disadvantages of each
- Define DAN, LAN, MAN, and WAN and describe their general characteristics
You should now be able to ... (3)

- Describe how STDM, FDM, and statistical multiplexing enable resource sharing and cite advantages and disadvantages of STDM and FDM versus statistical multiplexing
- Define bandwidth and latency
- Calculate bandwidth given the time needed to transmit one bit
- Define the components of latency and describe factors that can increase latency
- Calculate latency given information about the components