Announcements

- Please try logging in into the development servers and trying out the Visual Studio.NET tutorials
  - Lab 1 will be handed out on Thursday
- You should have received e-mail about MSDNAA accounts
  - Allows you to download/install Visual Studio.NET (Professional)
    - Kind of big: ~600MB compressed, uncompressed 1.2GB
  - You can develop/debug/test labs on your local machine if you also have IIS + ASP.NET
    - IIS comes as part of Windows XP (Professional)
    - ASP.NET is downloadable
  - Unfortunately, I do not have resources to troubleshoot this setup, so you are on your own
    - Your labs must work on the development servers I have set up

Communication in Distributed Applications

Realization of the OSI Model in the Internet
Data Link Layer

- Responsible for sending data frames on raw physical medium
  - Handling single-hop errors
- Example: Ethernet LANs (IEEE 802.3 standard)
- A broadcast network, so need to control access to the medium
  - How to ensure that two nodes are not transmitting at the same time?
  - How to achieve good channel utilization?
- CSMA/CD (Carrier Sense Multiple Access/Collision Detect) protocol
  - Wait for the channel to be idle before sending (carrier sense)
  - Detect any collisions early (by comparing received and sent signals)
    - If collision, abort transmission: saves channel capacity
    - Retry after a random period of time with exponential back-off

Network Layer

- Responsible for transmission of data packets across multiple hops
  - Also, controlling congestion on links
- Two key elements
  - Host addressing: A network-wide unique (and uniform) name
  - Routing: A mechanism for delivering a packet to the corresponding host
- Host addressing in the Internet Protocol (IP addresses)
  - IPv4: 32-bit addresses, written as four octets
    - E.g., 216.165.111.8
  - IPv6: 64-bit addresses, written as eight groups of four hex digits
    - E.g., 8000:123:4567:89AB:CDEF
    - Leading zeros omitted for brevity
  - Sometimes interpreted as a pair: (Network number, Host number)
    - (IP address of a host) & NETMASK = Network Number
    - E.g., (216.165.111.8) & (255.255.255.0) = (216.165.111.0)

IP Datagrams

- Packet header contains the source and destination IP addresses
  - Length of header minimum: 5 words
  - Higher level protocol encapsulated in packet (6: TCP, 11: UDP)
  - Used to limit number of hops traversed by a packet

Routing of IP Packets

Gateway: router1
128.122.140.144

Sent directly to host on same subnet (with same network number)
Propagation of Route Information

Keeping Routing Tables Manageable

- Agreed-upon classes of IP addresses
  - A 0.0.0.0 to 127.255.255.255 (126)
  - B 128.0.0.0 to 191.255.255.255 (65,000+)
  - C 192.0.0.0 to 223.255.255.255 (2M+)
  - D 224.0.0.0 to 239.255.255.255
  - E 240.0.0.0 to 247.255.255.255
- Since 1992: CIDR (Classless Inter-Domain Routing)
  - More flexible network prefixes: /13 (2048 Class C’s) to /27 (32 addresses)
- IP addresses allocated to ISPs, which allocate them to organizations (in turn to departments, groups, and finally to individual machines)
  - Hierarchical structure permits aggregation
  - E.g., Routers distant from NYU don’t need to be aware of how exactly the 128.122.*.* space is divided up among NYU routers
  - Traffic arrives at an NYU gateway router, which knows more details

Routing within a Subnet: Data Link Layer Addressing

- Data link layer hardware does not understand IP addresses
  - E.g., the Ethernet NIC on your PC
- Questions:
  - How to translate an IP address to a corresponding data-link layer address?
  - How to route IP packets using these addresses?
- At the data link layer, each network card has a unique address
  - For Ethernet cards, referred to as the MAC or HW address
    - 48-bits (e.g., 00:80:C8:B9:6B:94)
    - Manufacturers request a block of addresses from a central authority
  - Boards only pick up packets marked with their own HW address
  - Recall that Ethernet is a shared medium
  - Boards operating in “promiscuous” mode can pick up all packets
  - Ethernet hubs send all packets to all hosts
  - Ethernet switches are aware of HW addresses, only route relevant packets

Translating Between IP and HW Addresses

ARP: Address Resolution Protocol
Transport and Higher-Level Layers

- So far: Discussed sending an IP datagram from one host to another
  - Standalone packet
- Applications require higher-level abstractions
  - Services: Some way of identifying different programs on the recipient host that will deal with the packet
    - Addressing/naming handled using port numbers
    - Networking code responsible for demultiplexing
    - Realized as the User Datagram Protocol (UDP)
  - Connections
    - A continuous stream of packets
    - In-order, exactly-once delivery semantics, plus flow and congestion control
    - Realized as the Transmission Control Protocol (TCP)

Ports

- End-point of a communication operation
  - A 16-bit number
  - Alternatives: String (name of the application), URL
  - Tags packets as belonging to different services/streams
    - Note that there is no assumption that these packets will be picked up and/or serviced appropriately
    - Need network-aware programs that can do this – Rest of the Course
- Port numbers < 1024 are reserved for privileged programs (convention)
- Ports of publicly accessible services need to be widely advertised
  - On Unix, the /etc/services file

<table>
<thead>
<tr>
<th>Service</th>
<th>Port/Protocol</th>
<th>Service</th>
<th>Port/Protocol</th>
<th>Service</th>
<th>Port/Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftp</td>
<td>20/tcp, 21/tcp</td>
<td>domain</td>
<td>53/udp, 53/tcp</td>
<td>ms-sql</td>
<td>1433/tcp, udp</td>
</tr>
<tr>
<td>ssh</td>
<td>22/tcp</td>
<td>http</td>
<td>80/tcp</td>
<td>nfsd</td>
<td>2049/udp</td>
</tr>
<tr>
<td>smtp</td>
<td>25/tcp</td>
<td>pop3</td>
<td>110/tcp</td>
<td>rdp</td>
<td>3389/tcp</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

UDP

- A connectionless transport protocol
  - Send IP datagrams without establishing a connection
  - No guarantees of delivery (in- or out-of-order)
  - Used by applications whose interactions involve one request, one response
    - E.g., Domain Name Service (DNS)
- Packet format
  - 32 bits
  - IP header
    - Source port
    - Destination port
    - UDP length
    - UDP checksum
  - Payload

TCP

- A connection-based transport protocol
  - Connection identified by (src:sport, dst:dport) pair
  - Provides abstraction of a reliable byte stream
- Three major components
  - Connection setup
    - Permits ends of connection to synchronize on sequence numbers
    - Basis for reliable, in-order delivery
  - Window management
    - Allows receiver to control rate at which sender can send data
    - Ensures sufficient space at receiver to store packets → reliable delivery
  - Congestion management
    - Allows sender to detect and cope with congestion in the network
    - Prevents fill-up of buffer space at sender → reliable delivery
      - Improved throughput
TCP – Details

Connection Setup

Agreeing on SEQ # allows hosts to discover out-of-order, dropped packets (and request retransmission)

Window Management

SYN(SEQ=x)

SYN(SEQ=y, ACK=x+1)

(SEQ=x+1, ACK=y+1)

Sender

Receiver

2K

2K, SEQ=0

3K

2K

2K, SEQ=2048

Block sender

ACK=2048, WIN=2048

ACK=4096, WIN=0

1K, SEQ=4096

9/8/2003

TCP – Congestion Management

Why worry?

• Congestion may result in packet getting dropped
• Sender times out (waiting for an acknowledgement) and retransmits
• Retransmitted packet is also dropped, sender repeats
• Now assume, everybody is doing the same thing …

Solution

• Sender rate controlled by an additional parameter:
  \[ \text{Rate} = \min(\text{receiver window}, \text{congestion window}) \]
• Congestion window is dynamically adjusted based on time to receive acknowledgements

Helper Applications – Domain Name Service

• Applications prefer to work with symbolic host names
  – E.g., netserver1.pds.cs.nyu.edu, localhost
• The Domain Name Service (DNS) translates these into IP addresses
  – Sometimes, the reverse translation is also useful

Logical View

Physical View

Physical View

Network Programming

• Builds on top of networking protocols (primarily: UDP, TCP, HTTP)
  – Lowest-level API just provides user-level abstractions for TCP and UDP
• Sockets: Application-level end-point of communication
  – Operations often described by drawing analogy of a telephone
    • Call (Connection) setup
    • Conversation (Sending and receiving data packets)
    • Hangup (Disconnection)
• Next lecture: Sockets API