CSE 461 - Lecture 1

Course Mechanics

- Introduce staff
- Work expected:
  - Lectures, sections, reading, homeworks, projects, exams
- The text
- Grading
  - Late policy: global maximization of utility
  - Weighting: approximately 55% work that is turned in, 45% exams
- Projects
  - UDP, TCP, Custom request/response, HTTP proxy, Tor61, Bitcoins
  - Language: your choice
    - Java has been most popular, then Python, then C/C++
    - None of them is always best/simplest
  - Teams

Course Outcomes

After taking this course you should:

- Understand the core concepts of the Internet
- Understand what the key challenges are and approaches to overcoming them
- Be able to implement relatively sophisticated distributed applications
- Be able to evaluate an existing protocol and suggest improvements
- Implement protocols, on top of UDP and TCP, and understand why you’d rather not do that again

OSI Seven Layer Model

<table>
<thead>
<tr>
<th>Layer</th>
<th>Function</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Application</td>
<td>App specific</td>
<td>FTP, HTTP, SNMP, RTSP</td>
</tr>
<tr>
<td>6. Presentation</td>
<td>Data format conversion</td>
<td></td>
</tr>
<tr>
<td>5. Session</td>
<td>Multi-connection control</td>
<td>RTCP, SOCKS, RPC</td>
</tr>
<tr>
<td>4. Transport</td>
<td>Process ↔ Process</td>
<td>UDP, TCP, ssh</td>
</tr>
<tr>
<td>3. Network</td>
<td>Host ↔ Host</td>
<td>IPv4, IPv6</td>
</tr>
<tr>
<td>2. Data link</td>
<td>Encoding; logical link control; media access control</td>
<td>Ethernet (802.3), Wireless (802.11)</td>
</tr>
<tr>
<td>1. Physical</td>
<td>Analog ↔ Digital</td>
<td></td>
</tr>
</tbody>
</table>
**Wires**

- Transmission medium, channel
  - Wire, RF, IR, …
- Characteristics
  - bit rate
  - propagation delay
  - error rate / pattern

**Internet Topology**

- LAN
- switch
- router
- gateway

**UDP and TCP as Wires**

- Bit rate, propagation delay, error rate/pattern
- Naming endpoints: IP:port (e.g., 128.208.3.88:80)
  - The domain name service (DNS) provides translation between string names and IP addresses (e.g., between www.cs.washington.edu and 128.208.3.88)
- TCP
  - connection-based
  - reliable byte stream
    - What does “reliable” mean?
- UDP
  - datagram service
  - unreliable

**Ports / Sockets**

- Ports are defined by (and carried by) the transport protocol
- Sockets are provided by the OS
- *Bind* a socket to a port
**Using IP-based transports**

- Binding a socket
  - [Determining local IP]
    - getaddrinfo
    - localhost
  - Choosing a port
  - Determining remote IP:port
    - “Discovery”

**Using UDP**

- Write/send packets on the socket
  - Specify a destination IP:port
  - “Best-effort” delivery
- Read/receive packets from the socket
  - Data: what the sender sent
  - “Metadata: e.g., source IP:port

**Using TCP**

- Server:
  - Creates a socket and binds it to a port
  - listen: Indicates to OS that it wants to use it as a server socket – to wait for incoming connections
  - accept: Wait for an incoming connection
    - accept returns a new socket, bound to a new port
    - that socket is a connection that one client
      - read/receive / write/send
  - Some mechanism is needed so that the ability of the server to establish additional incoming connections doesn't depend on the behavior of the client that just connected
    - Cannot depend on a read from the socket connected to that client
- Client:
  - Discover server's IP:port (?)
  - Create a socket
    - bind: Port number usually doesn't matter
  - connect to server