Lecture 6: Transport Layer Protocols

CSE 123: Computer Networks
Aaron Schulman
Overview

- Process naming/demultiplexing
- User Datagram Protocol (UDP)
- Transport Control Protocol (TCP)
  - Three-way handshake
  - Flow control
Today: Transport Layer

Application Layer

Transport Layer

Network Layer

Link Layer

host

host

router

router

CSE 123 – Lecture 6: Transport Protocols
Transport Layer Tasks

- Define and provide specific delivery semantics
  - To which end point (which application/process)?
  - When?
  - How?
  - If?

- Multiplexing different processes on the same links

- Reliability
Naming Processes/Services

- Process here is an abstract term for your Web browser (HTTP), Email servers (SMTP), hostname translation (DNS)

- How do we identify for remote communication?
  - Process id or memory address are OS-specific and transient

- So TCP and UDP use **ports**
  - 16-bit integers representing mailboxes that processes “rent”
  - Identify process uniquely as (IP address, protocol, port)
We still have the problem of allocating port numbers
- What port should a Web server use on host X?
- To what port should you send to contact that Web server?

Servers typically bind to well-known port numbers
- e.g., HTTP 80, SMTP 25, DNS 53, … look in /etc/services
- Ports below 1024 traditionally reserved for well-known services

Clients use OS-assigned temporary (ephemeral) ports
- Above 1024, recycled by OS when client finished
Establishing well-known ports

Network Working Group  
Request for Comments #322  
NIC #9609  

Well Known Socket Numbers

V. Cerf  
J. Postel  
UCLA-NMC  
26 March 72

The Network Measurement Group would like to establish a network standard socket number for a Process Discard service (not all HOSTs need cooperate.) To do this, it is necessary to know which sockets at each HOST have already been allocated to standard network functions.

At all HOSTs which permit login for services, socket 1 is the Network Logger on which ICP may be performed.

We would like to catalog other sockets which are supposed to be well-known, so we would appreciate having a note or phone call from each HOST Technical Liaison describing the function and socket numbers of network service programs at each HOST. The catalog will be published and we would recommend that it be maintained at NIC.

Dr. Vint Cerf or Jon Postel  
3804 Boelter Hall  
UCLA Computer Science Department  
Los Angeles, California 90024

(213) 825-4864  Vint
(213) 825-4733  Jon
(213) 825-2368  Secretary
User Datagram Protocol (UDP)

- Provides *unreliable message delivery* between processes
  - Source port filled in by OS as message is sent
  - Destination port identifies UDP delivery queue at endpoint
- Connectionless (no state about who talks to whom)

<table>
<thead>
<tr>
<th>SrcPort</th>
<th>DstPort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td>Length</td>
</tr>
</tbody>
</table>

Data
UDP Delivery

Packets arrive

Ports → DeMux

Message Queues

Application process

Application process

Application process

Kernel boundary
UDP Checksum

- UDP includes optional protection against errors
  - Checksum intended as an end-to-end check on delivery
  - So it covers data, UDP header, and IP pseudoheader (history)
Applications for UDP

- Streaming media (e.g., live video)
- DNS (Domain Name Service)
- NTP (Network Time Protocol) (synchronizing clocks)
- FPS multi-player video games (e.g., Call of Duty)
- Why might UDP be appropriate for these?
Transmission Control Protocol

- Reliable bi-directional **bytestream** between processes
  - Uses a sliding window protocol for efficient transfer

- Connection-oriented
  - Conversation between two endpoints with beginning and end

- Flow control (last lecture)
  - Prevents sender from over-running receiver buffers
  - (tell sender how much buffer is left at receiver)

- Congestion control (later in term)
  - Prevents sender from over-running network capacity
TCP Delivery

Application process

Write bytes

TCP
Send buffer

Receive buffer

Transmit segments

Segment Segment ... Segment

Application process

Read bytes
TCP Header Format

- Ports plus IP addresses identify a connection (4-tuple)

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcPort</td>
<td>0-4</td>
</tr>
<tr>
<td>DstPort</td>
<td>5-15</td>
</tr>
<tr>
<td>SequenceNum</td>
<td>16-20</td>
</tr>
<tr>
<td>Acknowledgment</td>
<td>21-24</td>
</tr>
<tr>
<td>HdrLen</td>
<td>25-26</td>
</tr>
<tr>
<td>Flags</td>
<td>27-28</td>
</tr>
<tr>
<td>AdvertisedWindow</td>
<td>29-31</td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>UrgPtr</td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
</tr>
</tbody>
</table>
```
TCP Header Format

- Sequence, Ack numbers used for the sliding window
  - How big a window? Flow control/congestion control determine

Flow control – Receiver tells the transmitter how big its remaining window is
TCP Header Format

- Flags may be ACK, SYN, FIN, URG, PSH, RST

<table>
<thead>
<tr>
<th>Flags</th>
<th>Options (variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcPort</td>
<td>DstPort</td>
</tr>
<tr>
<td>SequenceNum</td>
<td>Acknowledgment</td>
</tr>
<tr>
<td>AdvertisedWindow</td>
<td>UrgPtr</td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
</tr>
</tbody>
</table>

CSE 123 – Lecture 6: Transport Protocols
For next time…

- More TCP details
- Read Ch 5.2 in P&D