UDP Protocol Details

Vermont Technical College
Peter C. Chapin

IP is Unreliable

- The underlying IP protocol is unreliable...
  1. Packets might not get delivered.
  2. Packets might get delivered in the wrong order.
  3. Packets might get delivered multiple times.
- This simplifies network infrastructure.
  - Routers can throw away packets if necessary.
  - Route can change on the fly.
  - Unnecessary retransmissions are not a problem.

I Want Reliability

- Most applications require reliable data transfer.
  - Can't accept lost data.
  - Can't accept rearranged data.
  - Can't accept unexpected data duplication.
  - Examples:
    - Large file transfer
    - Executable content
    - Email
    - Web pages
    - etc, etc...

Reliable Not Always Needed

- However, some applications can live with unreliability.
  - Streaming media
    - Timeliness is more important than completeness.
    - Lost data degrades quality of stream, but not usefulness
  - “Simple” applications
    - Client request fits in one packet
    - Server reply fits in one packet
      - Reply plays the role of an acknowledgment
      - Only one packet means no data reordering issues.

What About Duplicates?

- What if client request is duplicated?
  - Due to network.
  - Due to retransmission when server reply is lost.
- ***Defn**: A service is idempotent if the reply is the same whenever the request is identical and when the service has no side effects.*
  - Thus servicing duplicate requests has no bad effects
    - Aside from possible waste of server computational resources.

Square Root Server

- Imagine a “Square Root Server”
  - Client sends the server a floating point number.
  - Server returns the square root of that number.
- Does this meet the criteria?
  - Client request fits in one packet.
  - Server reply fits in one packet.
  - Service is idempotent.
- *The unreliability of IP is not a problem!*

IP Doesn't Know About Processes

- IP protocol delivers packets to machines (really interfaces).
  - Once the packet is delivered IP is “done.”
  - What happens to that data is no longer IP's concern.
- TCP has “ports” to help distinguish between processes on the same machine.
- What is needed..
  - A protocol like IP but that has ports.

User Datagram Protocol

- UDP (User Datagram Protocol)
  - No more reliable than IP
  - Uses 16 bit port numbers like TCP
    - But UDP ports are in a different port space. TCP port 1234 need not have any relation to UDP port 1234.
      - Although it is common to use the same number for a particular service.
  - A thin wrapper around IP.

Why?

- Why not just use TCP for everything?
  - TCP has significant overhead...
    - 3 segments to establish a connection
    - 4 segments (worst case) to tear down a connection
    - 20+ octet TCP segment header.
    - Acknowledgment segments.
  - A lot of extra stuff for the Square Root Server.
- UDP eliminates this overhead.
  - **No connections!** (saves seven segments)
  - Smaller header
  - No acknowledgments

UDP Header

```
 0      7 8     15 16    23 24    31
+--------+--------+--------+--------+
|     Source      |   Destination   |
|      Port       |      Port       |
+--------+--------+--------+--------+
|                 |                 |
|     Length      |    Checksum     |
+--------+--------+--------+--------+
|
|          data octets ...
+---------------- ...
```

UDP is described in RFC-768

Notice that the header is only 8 octets (down from TCP's 20 octets)

Length field includes the header (and is thus never less than 8)

Connectionless

- UDP does not use connections
  - Each *datagram* is sent independently.
    - Launched by sender without prior arrangement.
    - Arrives at receiver “by surprise.”
  - Reply datagrams...
    - Are sent to the client by “reversing” the address in the incoming datagram.
      - Incoming source IP address is outgoing destination IP address.
      - Incoming source port is outgoing destination port.
    - Outgoing datagram typically uses a different source port
      - So each “connection” is unique.
      - Assumes client is smart enough to use this new port if necessary (for additional communication when/if that happens).

Example

- [show diagram of packet exchange]