TCP/UDP Basics
Transport Service Overview

- Provide service to application layer by using the service provided by network layer
- Hide physical network
  - Hide processing complexity
  - Hide different network technologies and architectures
- Provide reliable, host-to-host transport
Transport layer design issues

- Addressing
- Connection Establishment
- Connection Release
- Flow Control
- Error Detection and Crash Recovery
Agenda

- **TCP: Transmission control protocol (RFC 793)**
  - Addressing
  - Connection Establishment
  - Connection Release
  - Flow Control
  - Error Handling
  - Interface and State Machine
  - TCP application examples

- **UDP: User datagram protocol (RFC 768)**
- **TCP vs. UDP**
TCP -- Addressing

- There are many network applications running on a host. When a packet arrive at network layer, how to know which application to send to?
  - Port: there are $2^{16} = 65536$ ports (0-65535) on one machine
  - One port is linked to only one application
  - One application may use many ports for different purposes
    - e.g. FTP: 20, 21

- How a client knows which service uses which port?
  - Permanent, well-known: often used service
    - 0-1023: well-known ports
    - 1024-49151: registered ports
    - 49152-65535: private ports
  - Process server proxy and create service on-the-fly: temporary service
  - Name server: for file service
TCP Addressing Header Fields

32 Bits

Source port | Destination port
TCP Connection Establishment – design issue

- Connection establishment becomes tricky when the network lose, delay and duplicate packets
  - Bank example

- How to differentiate a new packet from a delayed, duplicated packet
  - Sequence number
    - Sequence number increase for each packet
    - Sequence number space issue:
      - Sequence number wrap back
      - A packet should avoid using a sequence number that another packet is using
  - A duplicated or delayed packet should die after a while
    - IP layer already handles this issue by ‘Time To Live’ header field
TCP Connection Related Header Fields

32 Bits

<table>
<thead>
<tr>
<th>Source port</th>
<th>Destination port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td>Acknowledgement number</td>
<td></td>
</tr>
</tbody>
</table>

TCP header length

URG ACK PSH RST SYN FIN
TCP Connection Establishment – solution three way handshake

(a) TCP connection establishment in the normal case.
(b) Call collision.
TCP Connection Release – design options and issue (1)

- Two release method: asymmetric and symmetric

- Asymmetric release issue: possibility of losing data
TCP Connection Release – design options and issue (2)

- Two army issue
  - No protocol can work
  - One side should always take more risk
TCP Connection Release – solution
Three way handshake + timeout
TCP Flow Control – design issue

- Speed of data sending is critical

  - Too fast:
    - network congestion or
    - receiving side overload

  - Too slow – type example
    - waste of network resource or receiving memory
TCP Flow Control – solution (1)

- Windows maintained by both sending and receiving hosts
- Receiving side window size is decided by the available capability of receiving host’s
- Sender maintains two windows
  - receiver window (got from receiving host), congestion window (to calculate)
TCP Flow Control – congestion window size calculation

- slow start, reaching threshold, linear increment till timeout, recalculate threshold, slow start... till reaching receiver window size or timeout...
TCP Error Handling

- Host crash and recovery
- Data error during transmission
TCP Error Handling – TCP checksum

<table>
<thead>
<tr>
<th>Source address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination address</td>
</tr>
<tr>
<td>0000000000</td>
</tr>
</tbody>
</table>

Checksum Calculated Over Pseudo Header and TCP Segment
TCP service primitives

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOCKET</td>
<td>Create a new communication end point</td>
</tr>
<tr>
<td>BIND</td>
<td>Attach a local address to a socket</td>
</tr>
<tr>
<td>LISTEN</td>
<td>Announce willingness to accept connections; give queue size</td>
</tr>
<tr>
<td>ACCEPT</td>
<td>Block the caller until a connection attempt arrives</td>
</tr>
<tr>
<td>CONNECT</td>
<td>Actively attempt to establish a connection</td>
</tr>
<tr>
<td>SEND</td>
<td>Send some data over the connection</td>
</tr>
<tr>
<td>RECEIVE</td>
<td>Receive some data from the connection</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Release the connection</td>
</tr>
</tbody>
</table>
TCP application examples

- When to use TCP:
  - When an application needs a reliable transport

- Examples
  - File Transfer Protocol: FTP (21)
  - Secure Shell: SSH (22)
  - Teletype Network: TELNET (23)
  - Simple Mail Transfer Protocol: SMTP (25)
  - Hypertext Transfer Protocol: HTTP (80)
Agenda

- TCP: Transmission control protocol (RFC 793)
- UDP: User datagram protocol (RFC 768)
  - UDP header
  - UDP properties
  - UDP application examples
- TCP vs. UDP
## UDP Header

<table>
<thead>
<tr>
<th>Source port</th>
<th>Destination port</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP length</td>
<td>UDP checksum</td>
</tr>
</tbody>
</table>

- **UDP Destination Port**: identifies destination process
- **UDP Source Port**: optional – identifies source process for replies, or zero
- **Message Length**: length of datagram in bytes, including header and data
- **Checksum**: optional -- 16-bit checksum over header and data, or zero
UDP Properties

- UDP provides an unreliable datagram service
  - Packets may be lost or delivered out of order
  - Message split into datagrams, user sends datagrams as packets on network layer
  - No buffer at either sending or receiving side
  - Unreliable but fast
  - Full duplex
  - Application must deal with lost packets
UDP Application Examples

- When to use UDP
  - Reduce the requirement of computer resources
  - The checking scheme has provided completely by the application program
  - When using the Multicast or Broadcast to transfer
  - The transmission of Real-time packets

- Examples
  - Trivial File Transfer Protocol, TFTP
  - Simple Network Management Protocol, SNMP
  - Dynamic Host Configuration Protocol, DHCP
  - Domain Name System, DNS
  - Routing Information Protocol, RIP
  - Real-Time Transport Protocol, RTP
## TCP vs. UDP

<table>
<thead>
<tr>
<th>TCP</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>connection-oriented</td>
<td>connectionless</td>
</tr>
<tr>
<td>confirmed service</td>
<td>unconfirmed service</td>
</tr>
<tr>
<td>high overhead</td>
<td>low overhead</td>
</tr>
<tr>
<td>(header 20 bytes)</td>
<td>(header 8 bytes)</td>
</tr>
<tr>
<td>flow control</td>
<td>no flow control</td>
</tr>
</tbody>
</table>
References