The Transport Layer

• **End-to-End Communication**
  – Enable processes to communicate

• **Transport Services**
  – Connection Oriented/ Connectionless
    • User Datagram protocol
    • Transmission control protocol
Transport Layer QoS

• Transport Quality of Service (QoS)
  – Connection establishment delay
  – Connection establishment failure probability
  – Throughput
  – Transit delay (Source to Destination)
  – Residual error ratio
    • Lost packets / total sent
Transport Layer (QoS)

- **Protection**
- **Priority**
  - Different transport connection Priorities
  - Resilience – Probability of TPL terminating a connection
## Transport Layer Primitives

<table>
<thead>
<tr>
<th>Primitives</th>
<th>TPDU Sent</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTEN</td>
<td>None</td>
<td>Block until some process tries to come</td>
</tr>
<tr>
<td>Connect</td>
<td>Connect request</td>
<td>Actively attempt to establish connection</td>
</tr>
<tr>
<td>Send</td>
<td>Data</td>
<td>Send Information</td>
</tr>
<tr>
<td>Receive</td>
<td>None</td>
<td>Block until a TPDU arrives</td>
</tr>
<tr>
<td>Disconnect</td>
<td>Disconnect request</td>
<td>One Side wants to release connection</td>
</tr>
</tbody>
</table>
Connection Management:

Addressing: Well known TSAPs for servers

TSAP – Transport Service Access Point
TCP Connection Establishment

• A Directory server on *host2* attaches to *TSAPy* on host
  – Waits for an incoming call (Listen)

• An application process at *host1* wants some directory assistance

• (Source *TSAPx* and Dest *TSAPy*)
TCP Connection Establishment (contd)

- TP entity (host1) sets up network connection between host1 and host2.
  - TP entity – asks for connection between TSAP x on host1 and TSAP y on host2.
    - TP entity on host2 check whether TSAP y on host2 is willing to accept a connection
    - if accepted connection established
Issues in Communication

How does $TSAP_x$ know that $TSAP_y$ on host2 is the directory server?

Possibility – this server always attaches itself to $TSAP_y$

Issues – many servers – not always used

Process server

proxy for less - heavily used servers
Properties of the Transport Layer

- Guarantees message delivery (if desired)
- Deliver message in the same order they were sent
- Deliver only one copy of each message
- Support arbitrarily large messages
Properties of the Transport Layer

• Support synchronisation between sender and receiver
• Allow receiver to apply flow control to sender
• Support multiple applications on each host
Transport Layer Services

- Limitations due to underlying Network:
  - A simple asynchronised demultiplexing service
  - A reliable byte stream
  - A request / reply service
UDP Header

- Address used to identify host address
- pid (OS assigned?)
- Distributed system/single OS
- Indirectly identify each other using a port / mailbox
- send source port
- receive port destination

Address used to identify destination address
UDP-Continued

- IP address + port uniquely identify a process
  - Demultiplexing key for UDP
- Error Checking: Checksum
  - UDP header, UDP data + Pseudo header (IP addresses + protocol number + UDP length)
Processes and Ports

• How does the client/server know each other’s port number:
  • Generally: Server talks on well known port
  • Example: DNS requests on 53
  • Unix talk on 517

• Mapping services to PortNum /etc/services
  (Published in a RFC)
Processes and Ports

- Once client talks to server, the server gets client port address
  - sends on that port
  - port – only an abstraction

- Vary from OS to OS
  - A message queue
  - Application process removes from queue
  - When message arrives appended to end of queue
UDP

Application process

Application process

Application process

port

port

port

No flow control

Packets arrive