

Transport Layer – Overview


Kameswari Chebrolu

All the figures used as part of the slides are either self created or from the public domain with either 'creative commons' or 'public domain dedication' licensing. The public sites from which some of the figures have been picked include: <http://commons.wikimedia.org> (Wikipedia, Wikimedia and workbooks); <http://www.sxc.hu> and <http://www.pixabay.com>

Milestones

- Progression in scale of networks
 - Point-to-point link (2 nodes)
 - Small local area networks (tens of nodes)
 - Extended local area networks (thousands of nodes)
 - Heterogeneous inter-networks (millions of nodes)

Milestones

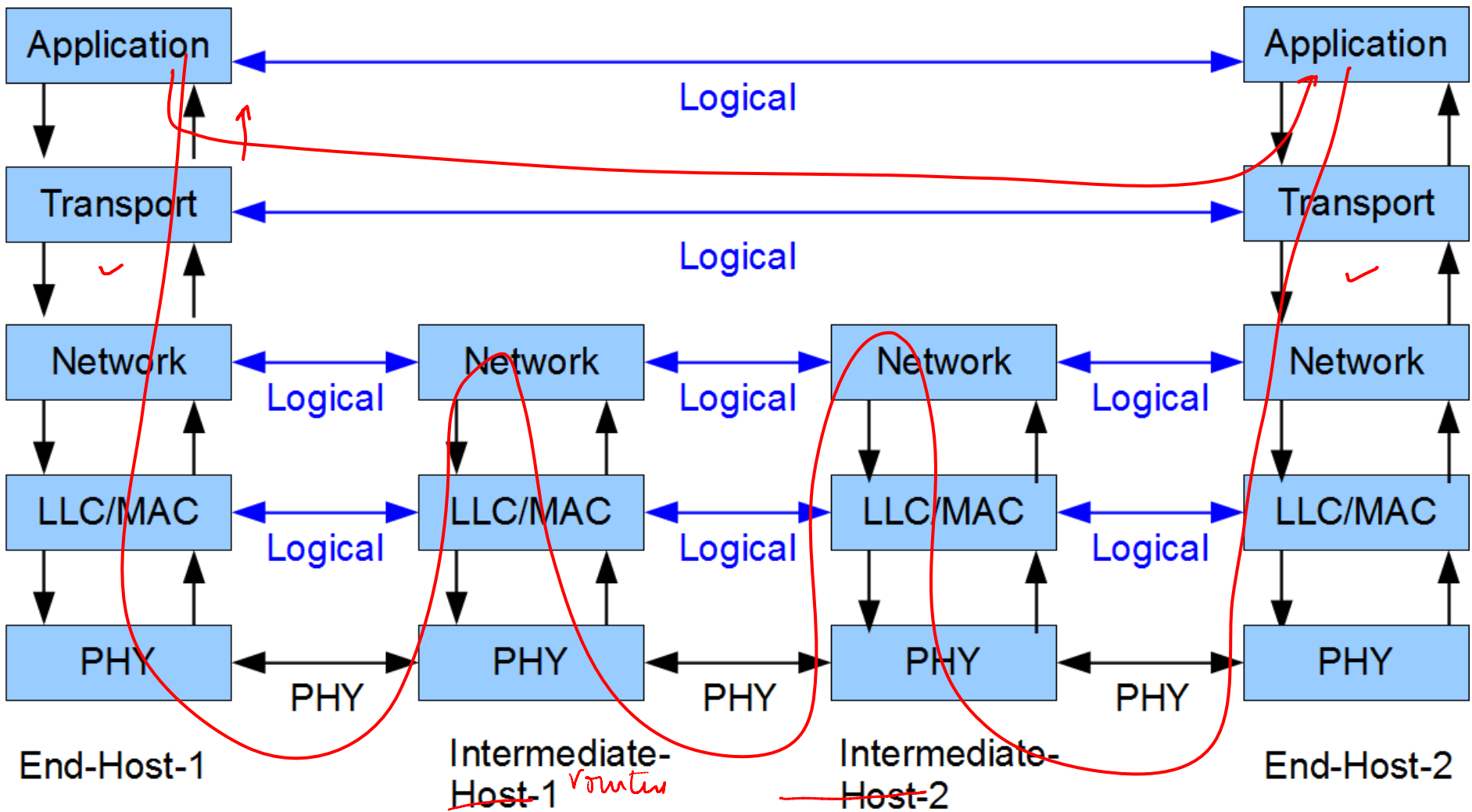
- Can now handle host-to-host delivery
 - Network layer (determines which next hop) uses services of link layer (delivers to next hop) which in turn uses services of physical layer (converts bits to signals) to deliver packets
- Next: Process to process communication →
role of the transport layer

Transport Layer Service

web browser
email
SSH

top

- Hosts run many application processes
- Transport layer provides logical communication between processes
 - Help multiplex/demultiplex packets to deliver to right process
 - Enhance network layer services
- Transport protocols also called end-to-end protocols since they are implemented on end hosts
- The unit of data at transport layer is termed 'segment'




Application Layer Expectations

Email
File transfer

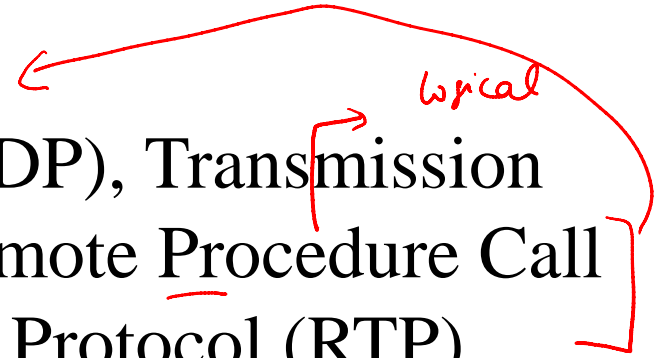
Guaranteed delivery

- Guaranteed message delivery ↗
↘
- Ordered delivery
- Delay guarantees
- No duplication
- Support arbitrarily large messages
- Support flow control ↗

Network Layer Limitations

- Best effort service model
- Packet Losses
- Re-ordering
- Duplicate copies
- Limit on maximum message size  MTU
- Long delays

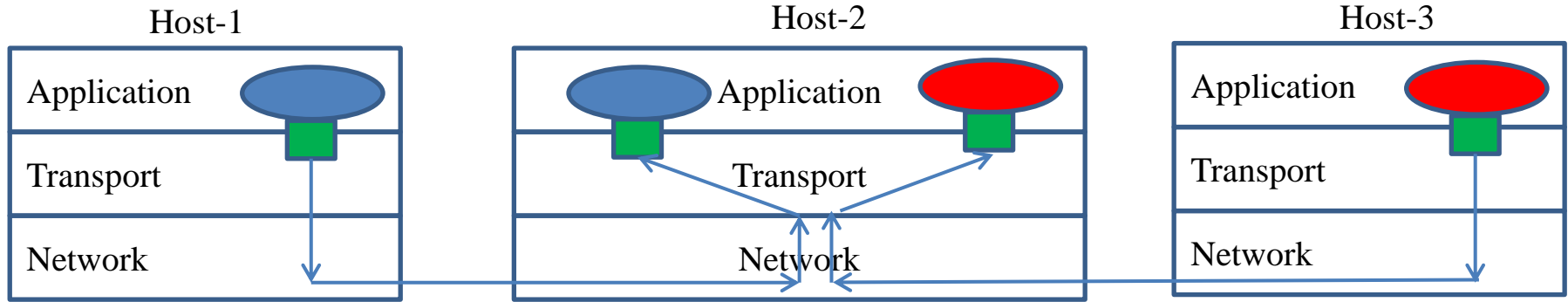
Challenge

- Enhance network layer services to meet application expectations
 - Cannot provide services that inherently cannot be supported by network layer (e.g. delay guarantees)
 - Different transport protocols offer different tradeoffs
 - User Datagram Protocol (UDP), Transmission Control Protocol (TCP), Remote Procedure Call (RPC), Real-time Transport Protocol (RTP)
- 
- A hand-drawn red arrow originates from the list of protocols (UDP, TCP, RPC, RTP) and points to the word 'typical' written in red above the list. A red bracket is also drawn around the list of protocols.

Break

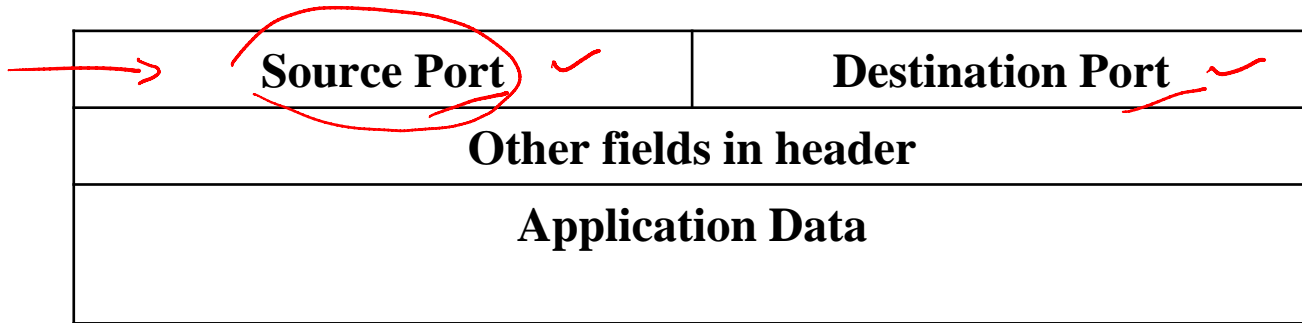


Multiplexing/Demultiplexing



Demultiplexing: Deliver segments to the right socket

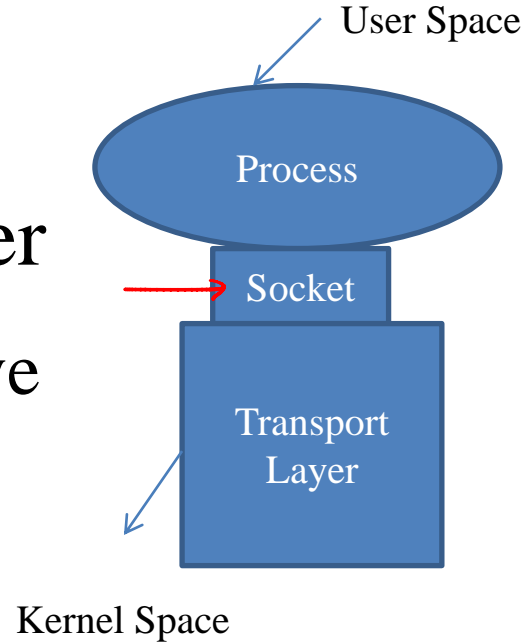
Multiplexing: Assemble segments such that they get delivered to right socket



Transport Layer Segment

Sockets

- Socket: An interface between an application process and transport layer
 - The application process can send/receive messages to/from another application process (local or remote) via a socket
- In Unix jargon, a socket is a file descriptor – an integer associated with an open file



Multiplexing/Demultiplexing

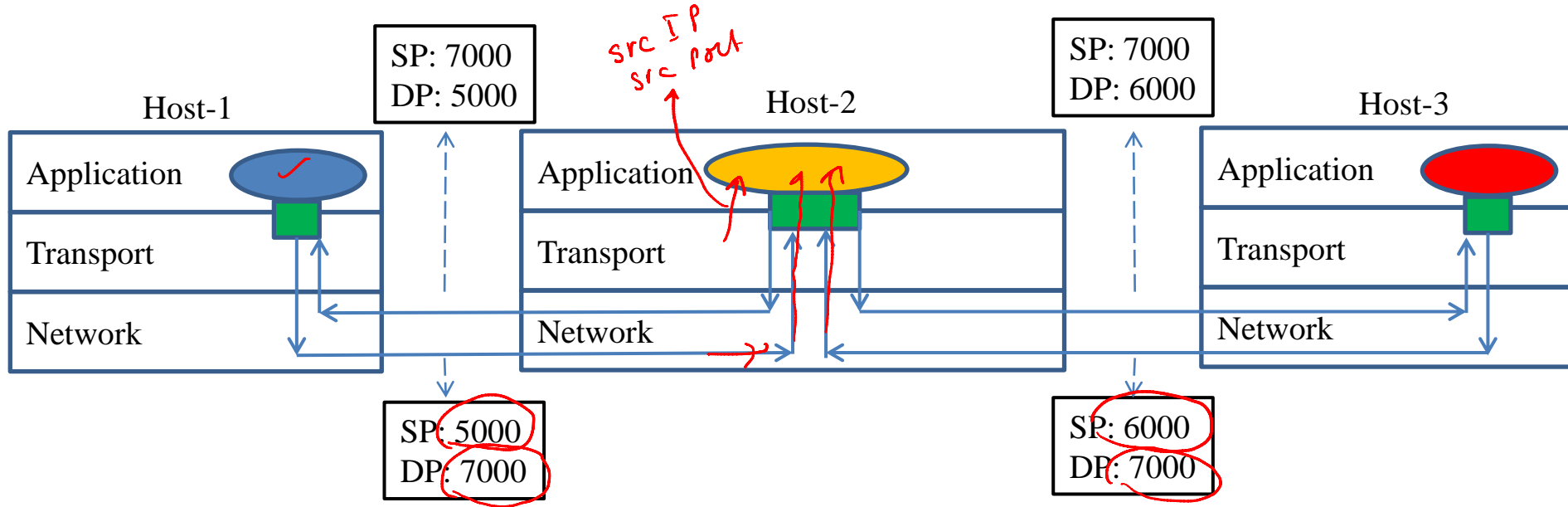
- Application developer can
 - specify type of transport protocol
 - configure a few parameters related to transport protocol
- To help mux/demux a segment
 - Sockets have unique identifiers (one of them is ports)
 - Segments carry fields that help identify right socket
 - Fields of relevance: Source and destination port

→ UDP
TCP

Connectionless Mux/Demux

- Used with UDP sockets
- Socket identified by two-tuple:
 - Destination IP address, Destination port number
- Transport layer checks port information in segment and directs to right socket
- IP datagrams with different source IP addresses and/or source port numbers directed to same socket

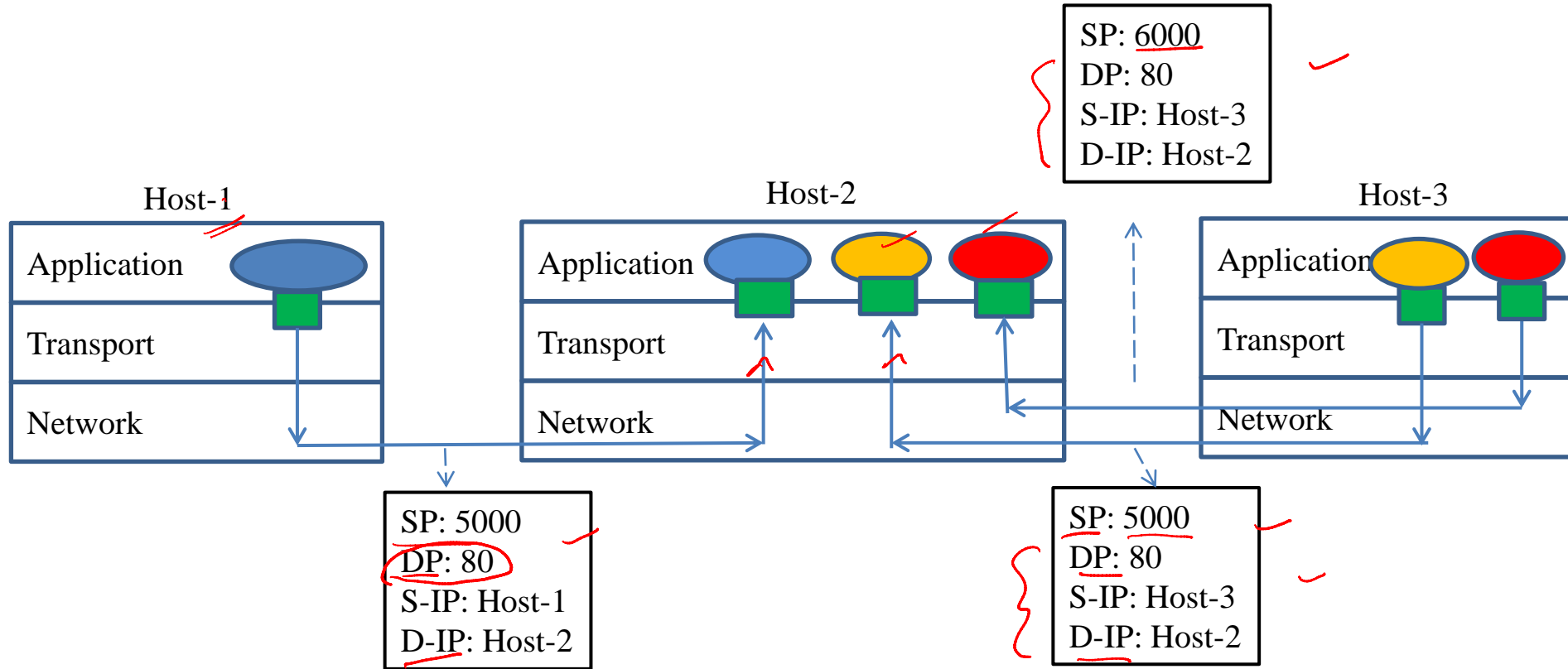
Example



Connection-oriented Mux/Demux

- Used with TCP sockets
- Socket identified by 4-tuple:
 - Source IP address ✓
 - Source port number ✓
 - Destination IP address ✓
 - Destination port number ✓
- All four values are used to direct segment to the right socket

Example

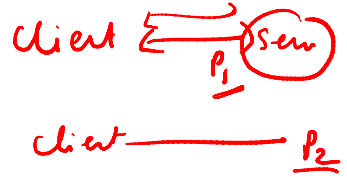


Obtaining Port Information

- Client contacts server
 - Client picks a random port and sends message
 - Server knows identity of client process (based on source port in received message)
- How does client know server's port info?
 - Server's listen to messages on well known ports
 - Refer to /etc/services in Unix systems
 - In some applications, well known port is the starting point to agree upon some other port

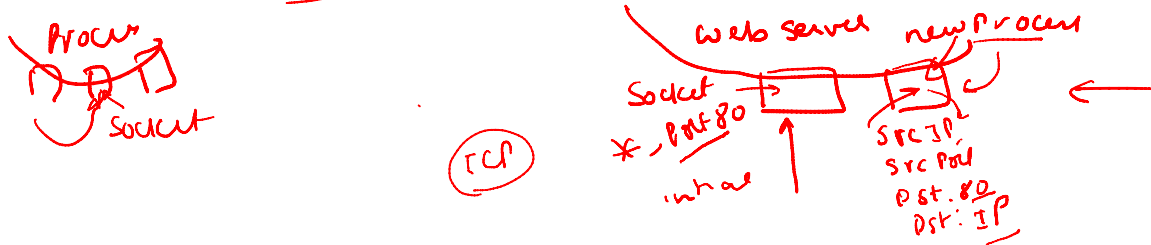
src \neq p, ^{src}port

src



A Note on Servers

- Server host listens on a designated port but has different socket for each connecting client
 - Each socket identified by its own 4-tuple
 - There need not be one-to-one correspondence with sockets and processes
 - E.g. Threaded server have many sockets but one process



Summary

- The role of transport layer is to provide logical communication between processes
 - All transport protocols provide multiplexing and demultiplexing capability
 - Others try to enhance network services to meet application specific requirements
- Different types of mux/demux and role of sockets