CS 372 Lecture #14

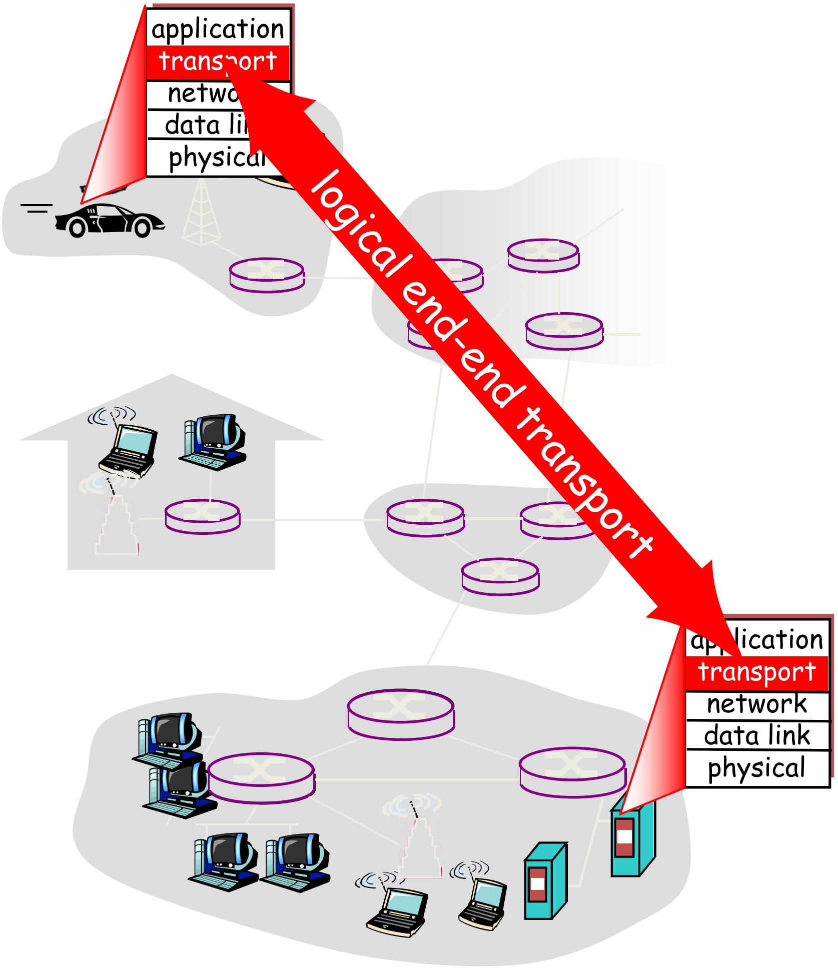
The Transport Layer:

- Introduction
- Multiplexing

Note: Many of the lecture slides are based on presentations that accompany *Computer Networking: A Top Down Approach,* 6th edition, by Jim Kurose & Keith Ross, Addison-Wesley, 2013.

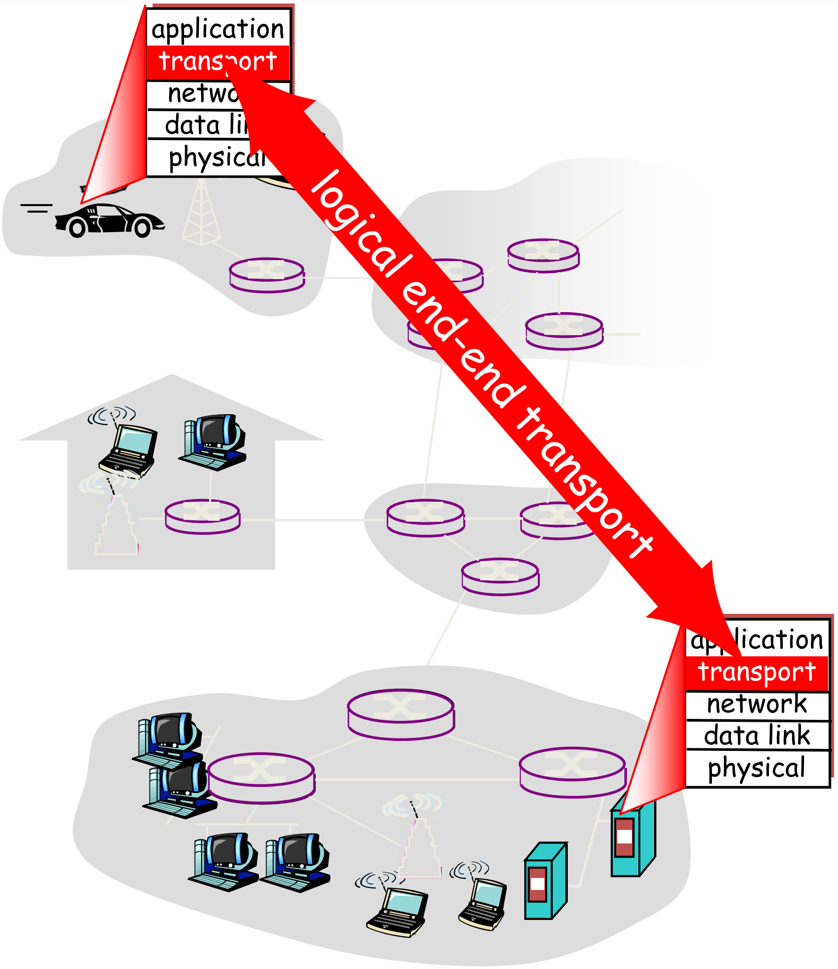
Transport services and protocols

- Transport protocols
  - provide *logical communication* between application processes running on different hosts
  - run on end systems



Transport services and protocols

- sender protocol:
  - accepts messages from application socket
  - breaks messages into data blocks
  - encapsulates blocks/ports into segments
  - passes segments/addresses to network layer
- receiver protocol:
  - accepts segments from network layer
  - re-assembles data blocks into messages
  - passes messages to sockets at application layer



Transport Layer requires some Network Layer services

- *transport layer:* logical communication between processes

- *network layer:* logical communication between hosts
  - Packet addressing
  - Route computation
  - Packet forwarding

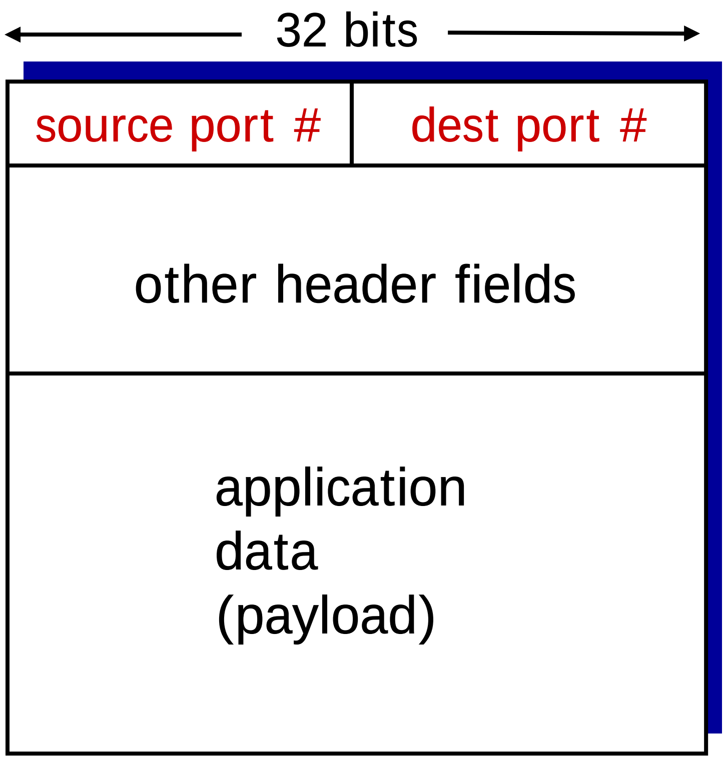
  - More later on the network layer …

Multiplexing/demultiplexing

- <u>Multiplexing at sending hosts:</u>
  - gathering data from multiple sockets, creating segments, encapsulating segments with header (later used for demultiplexing)
- <u>Demultiplexing at receiving host:</u>
  - delivering received segments to correct socket

How demultiplexing works

- host receives IP datagrams
  - each datagram has source IP address, destination IP address
  - each datagram encapsulates one transport-layer segment
  - each segment has source, destination port number
- host uses *IP addresses & port numbers* to direct segment to appropriate socket



32 bits

source port # | dest port #

other header fields

application data (payload)

TCP/UDP segment format

Connectionless demultiplexing

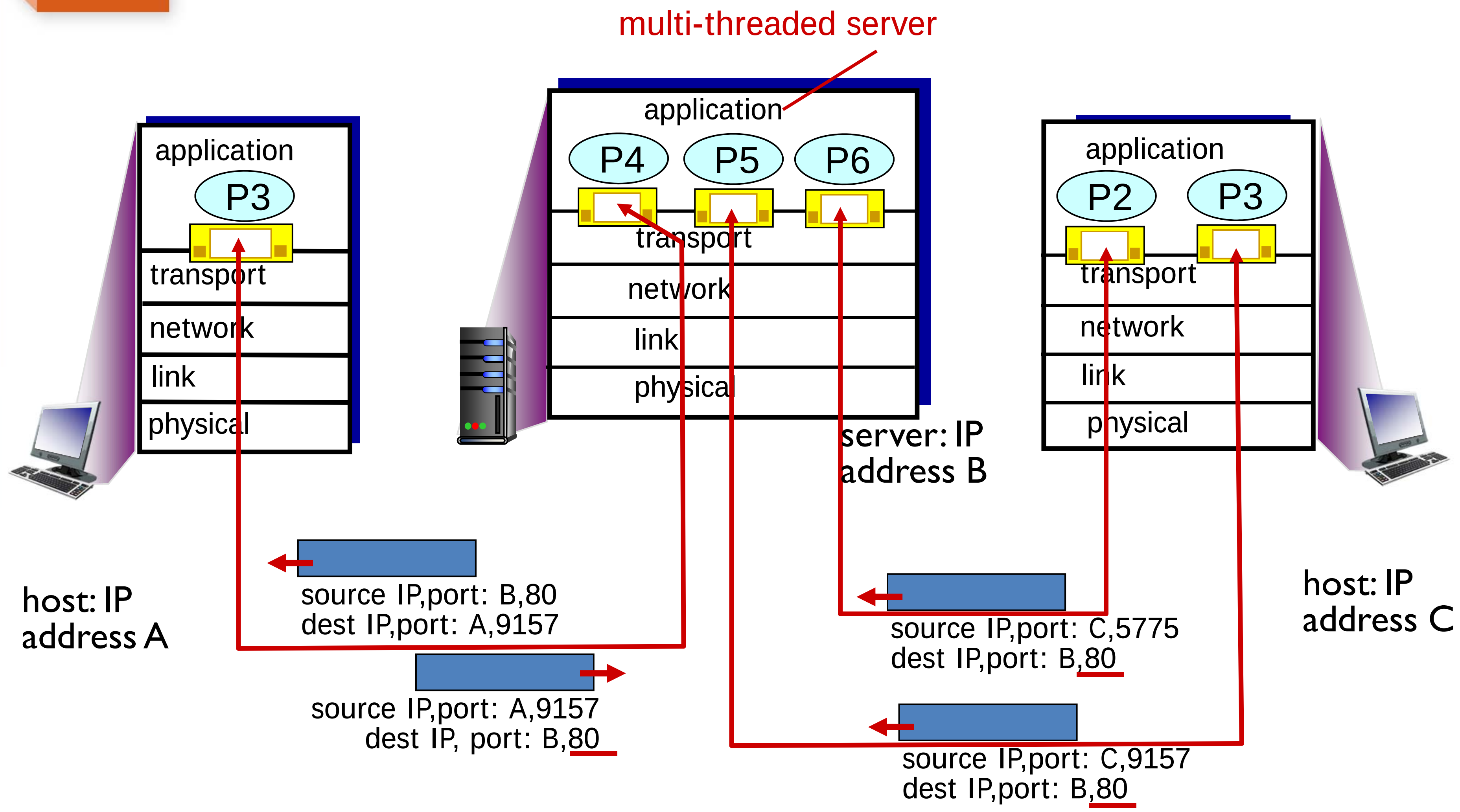
- UDP socket identified by destination (IP address, port number)
- When host receives UDP segment:
  - checks destination port number in segment
  - directs UDP segment to socket with that port number
- IP datagrams with different source IP addresses and/or source port numbers might be directed to same socket

Connection-oriented demultiplexing

- TCP socket identified by
  - source IP address
  - source port number
  - destination IP address
  - destination port number
- receiving host uses all four values to direct segment to appropriate socket

- Server host may support many simultaneous TCP sockets:
  - each socket identified by its own 4-tuple
- Web servers have different sockets for each connecting client
  - non-persistent HTTP will have different socket for each request

Demultiplexing: example



three segments, all destined to IP address: B,
dest port: 80 are demultiplexed to *different* sockets

Summary Lecture #14

- Definitions
  - segment
  - multiplexing
  - demultiplexing

- Transport layer
  - inputs, outputs
  - responsibilities
  - sockets