CS 372 Lecture #8

The Application Layer:

- Network applications
- Application architectures
- Process communication

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.

Application Layer

Objectives:

- Understand concepts and implementation aspects of network application protocols
  - transport-layer service models
  - client-server paradigm
  - peer-to-peer paradigm
- Understand protocols, by examining popular application-layer protocols
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
- Program network applications
  - socket API

Example network applications

- e-mail
- web
- instant messaging
- remote login
- P2P file sharing
- multi-user network games
- streaming stored video clips
- voice over IP
- real-time video conferencing
- grid computing

- Question: Why are these called “applications”?
- Answer:
  - They communicate data generated by user programs (messages, queries responses, etc.)

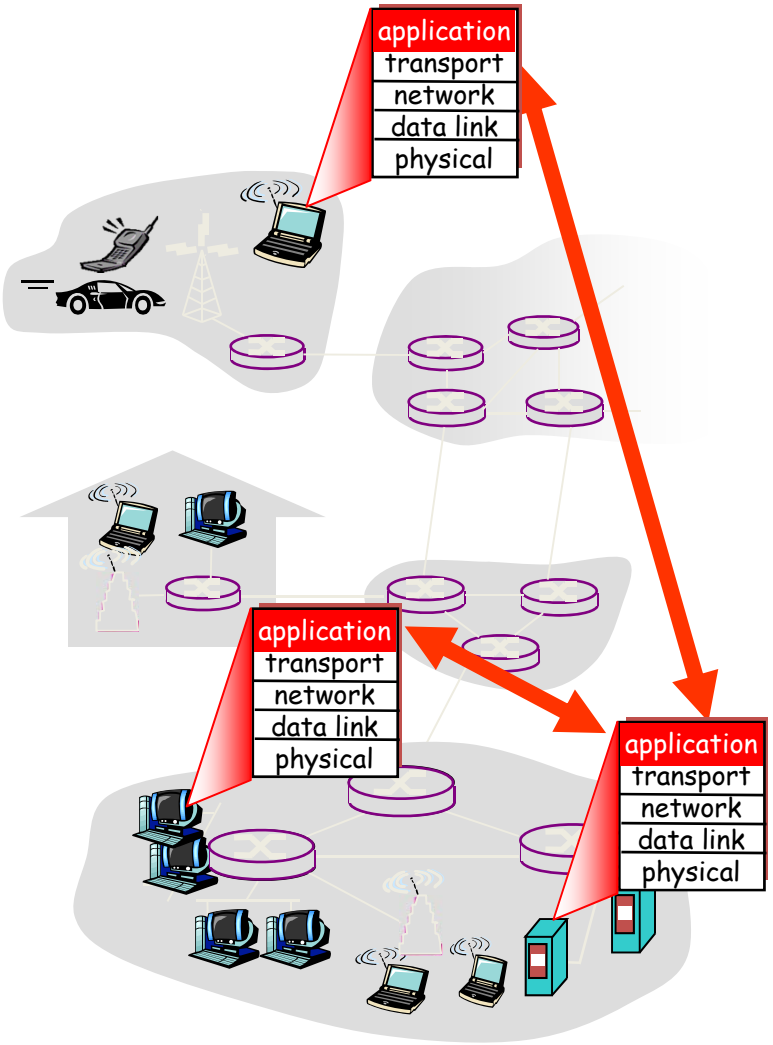
Creating a network app

Write programs that can

- run on (different) *end systems* at the network edge
- communicate via network
  - e.g., web server software communicates with browser software

Not much software is written for devices in network core

- network core devices do not run user applications
- most operations implemented in hardware

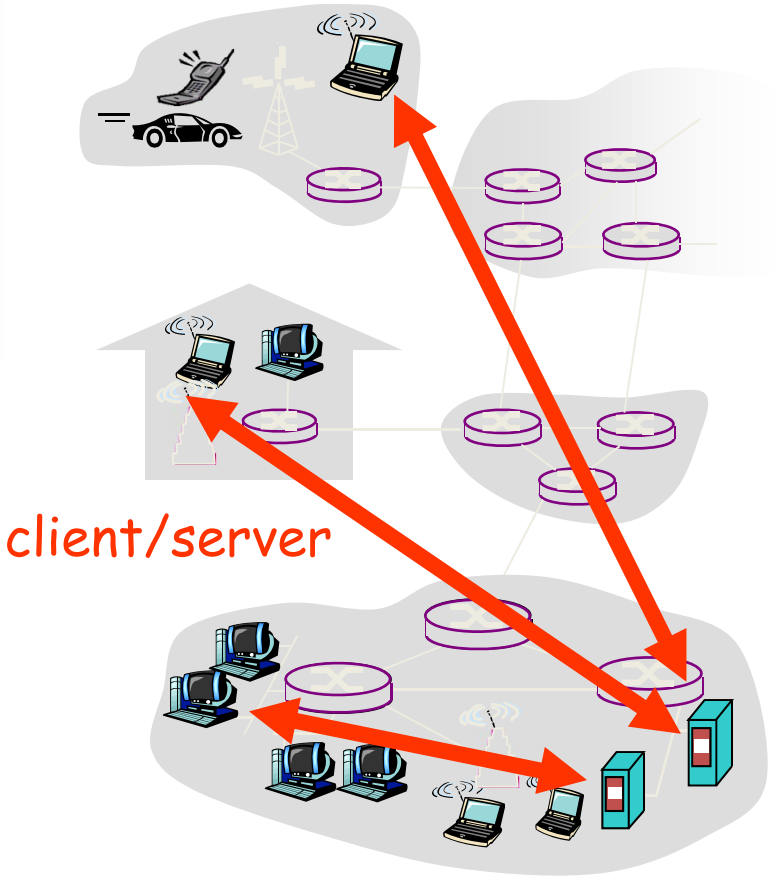


Application Architectures

3 types:

- Client-server
- Peer-to-peer (P2P)
- Hybrid of client-server and P2P

Client-server architecture



server:

- always-on
- fixed/known IP address
- server "farms" and multi-threading for **scaling**
  - serves many clients simultaneously

clients:

- communicate with server only
- intermittently connected
- may have dynamic IP addresses
- do not communicate directly with each other

examples:

- Get services from Google, Amazon, MySpace, YouTube, etc.

Pure P2P architecture

There is no always-on server

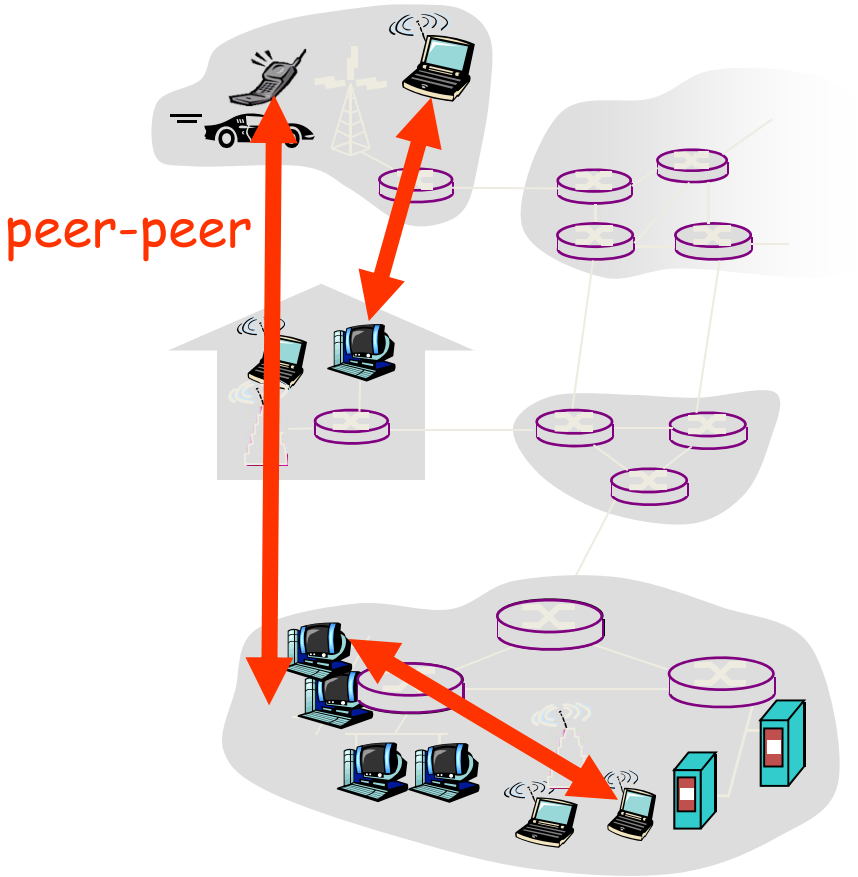
- arbitrary end systems communicate directly
- peers are intermittently connected and might change IP addresses
- example: BitTorrent (file distribution through P2P)

Pros:

- scalable
- distributive

Cons:

- difficult to manage
- not secure



Hybrid of client-server and P2P

Skype

- voice-over-IP P2P application
- centralized server: finding address of remote party
- client-client connection: direct (not through server)

Instant messaging

- chatting between two users is P2P
- centralized service: client presence/location
  - user registers IP address with central server when it comes online
  - user contacts central server to find IP addresses of buddies

Processes communicating

Process: a program running within a host.

- processes inside a single host communicate using inter-process communication (managed by OS).
- processes in two different hosts communicate by exchanging messages (managed by protocols).

Client process: process that initiates communication

Server process: process that waits to be contacted

- Discussion question: Applications with P2P architectures have both client processes & server processes
  - Why?

Addressing processes

- To receive messages, process must have an *identifier*
- Receiving host device has unique IP address
- *Q:* is IP address of host (on which process runs) sufficient to identify the process?

- *A:* No, *many* processes can be running on same host.

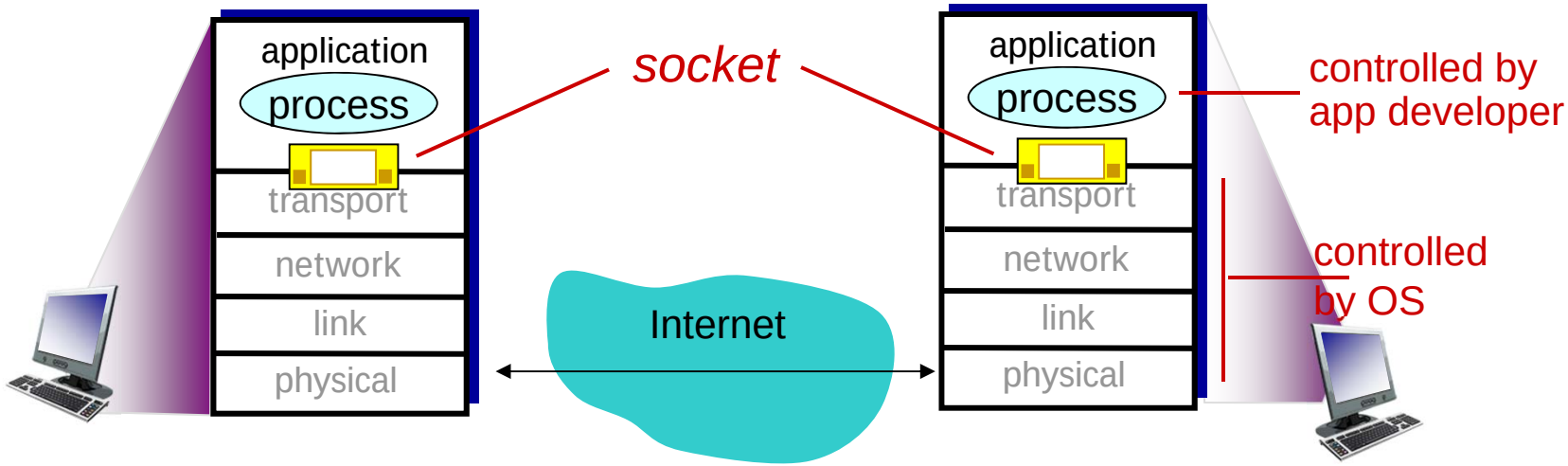
Addressing processes

- *identifier* consists of:
  - IP address (host)
  - port number (process)
- This identifier is called a *socket* or an *endpoint*
- Example port numbers:
  - HTTP server: 80
  - Mail server: 25
- To send HTTP message to `oregonstate.edu` web server:
  - IP address: 128.193.4.112
  - Port number: 80
  - Try it ! (http://128.193.4.112:80)

Discussion question: Can a server process at a specific port number serve more than one client concurrently? If so … how?

Sockets and Connections

- Process sends/receives messages to/from its *socket*
- A *connection* is a socket pair (4-tuple consisting of the client IP address, client port number, server IP address, and server port number) that uniquely identifies the two endpoints.



Some "well-known" logical port numbers

Port	Name	Description
7	echo	Echo input back to sender
11	systat	System statistics
13	daytime	Time of day (ASCII)
17	quote	Quote of the day
20,21	ftp	File Transfer Protocol
22	ssh	Secure Shell remote login
23	telnet	Telnet
37	time	System time (seconds since 1970)
53	domain	Domain Name Service (DNS)
80	web, http	World Wide Web (HTTP)
123	ntp	Network Time Protocol (NTP)
161	snmp	Simple Network Management Protocol (SNMP)

Summary Lecture #8

- Application Layer
  - Examples of network applications
  - Application architectures
    - Client-server, Peer-to Peer (P2P), etc.
- How processes communicate
- Definitions
  - client, server
  - port number
  - socket, connection