CSci 4061
Introduction to Operating Systems

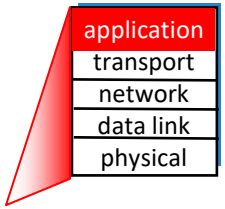
Network Systems Programming

Chapter 4.4, 18 (mostly 18.7 onwards) R&R

Goal:

- Enable communication between processes on different machines … (finally)

Network Layers



advantage of layers?

- Physical Layer
  - media: cat5, radio, transmits bits
  - electrical, data, rates, connectors
- Link layer
  - 802.3 (Ethernet),802.11 (wifi) transmits error-free
  - framing, CRC codes, …
- Network layer
  - Routing (path from machine A to machine B)
  - Send packets; packets contain addresses
- Transport layer: systems-programming interface

Protocols

- Transport protocols
  - TCP
    - Out of order pocket assembly
    - Re-transmits packets lost or corrupted
    - Connection-oriented (expensive)
    - Reliable

  - UDP
    - Connectionless
    - Unreliable

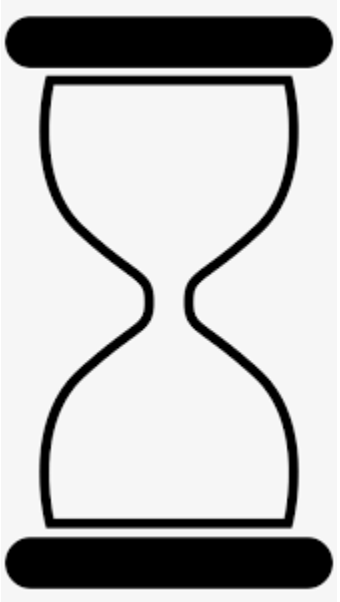
- Application protocols
  - Applications
    - Data formats
    - Meaning of data
    - …ftp,http,etc

phone call

mailing a letter

Some network app/protocols

- 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



App-layer protocol defines

- Types of messages exchanged
  - e.g. request, response
- Message syntax
  - What fields in messages & how fields are delineated
- Message semantics
  - Meaning of information in fields
- Rules for when and how processes send & respond to messages

HTTP Example

Request:

GET http://www.cnn.com/ HTTP/1.0

Accept: text/html If-Modified-Since:

Saturday, 15-January-2000 14:37:11

GMT

User-Agent: Mozilla/4.0

Response:

HTTP/1.0 200 OK

Date: Sat, 15 Jan 2000 14:37:12 GMT

Server: Microsoft-IIS/2.0

Content-Type: text/HTML

Content-Length: 1245

Last-Modified: Fri, 14 Jan 2000
08:25:13 GMT

This is the webpage contents ….

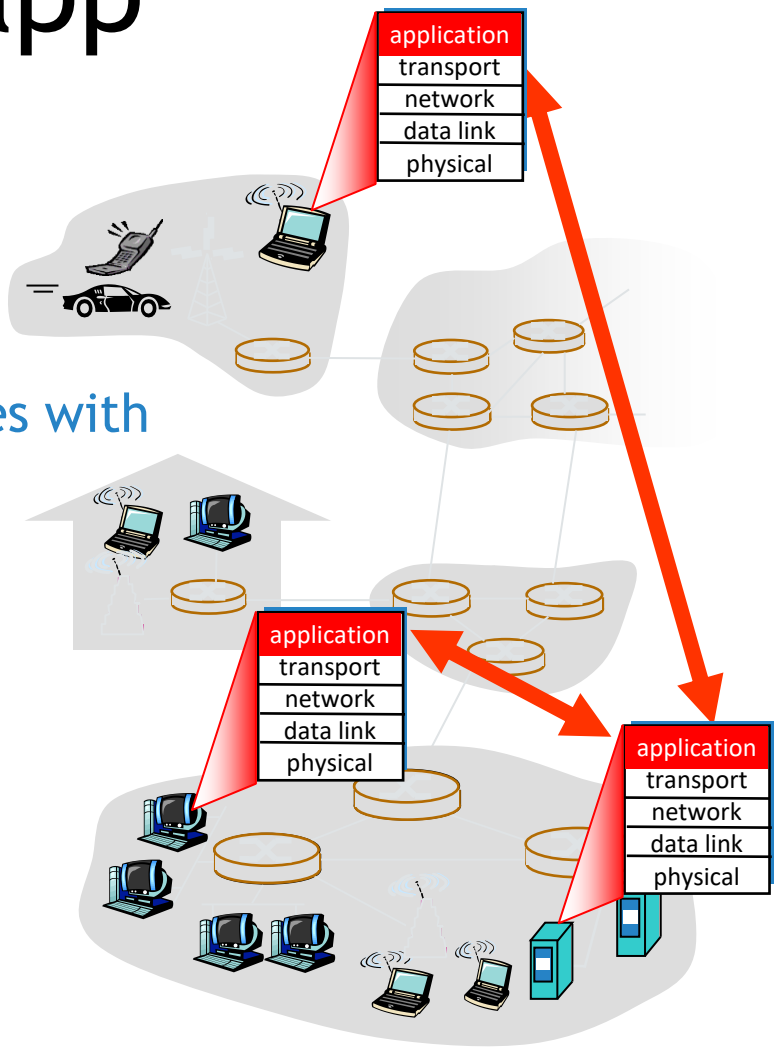
Addresses

- Local IPC—easy? pid, mailbox, etc
- Network communication: IP address
  - Machine + network interface
  - IPv4 a.b.c.d: 2^32, ~4 x 10^9 addresses
- Network layer routes packet to destination address (a.b.c.d)
- Symbolic names can be assigned to addresses to identify domains and end hosts
  - *caesar.cs.umn.edu* resolves to a.b.c.d by domain name service (DNS)

Creating a network app

write programs that

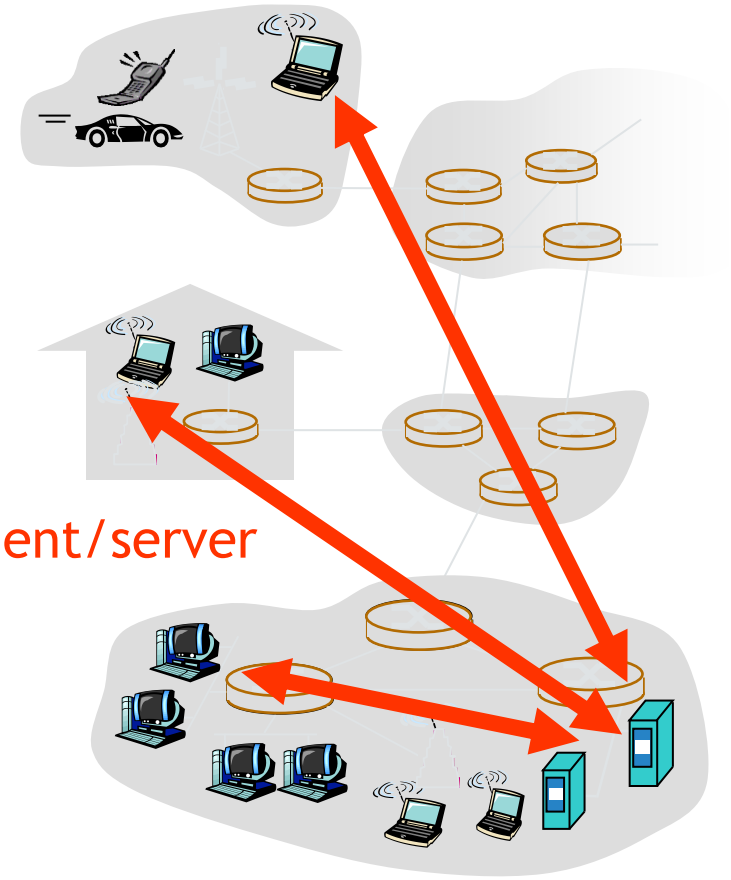
- run on (different) *end systems*
- communicate over network
- e.g., web server software communicates with browser software



Application architectures

- Client-server
- Peer-to-peer (P2P)

Client-server architecture



server:

- always-on host
- permanent IP address
- server farms for scaling

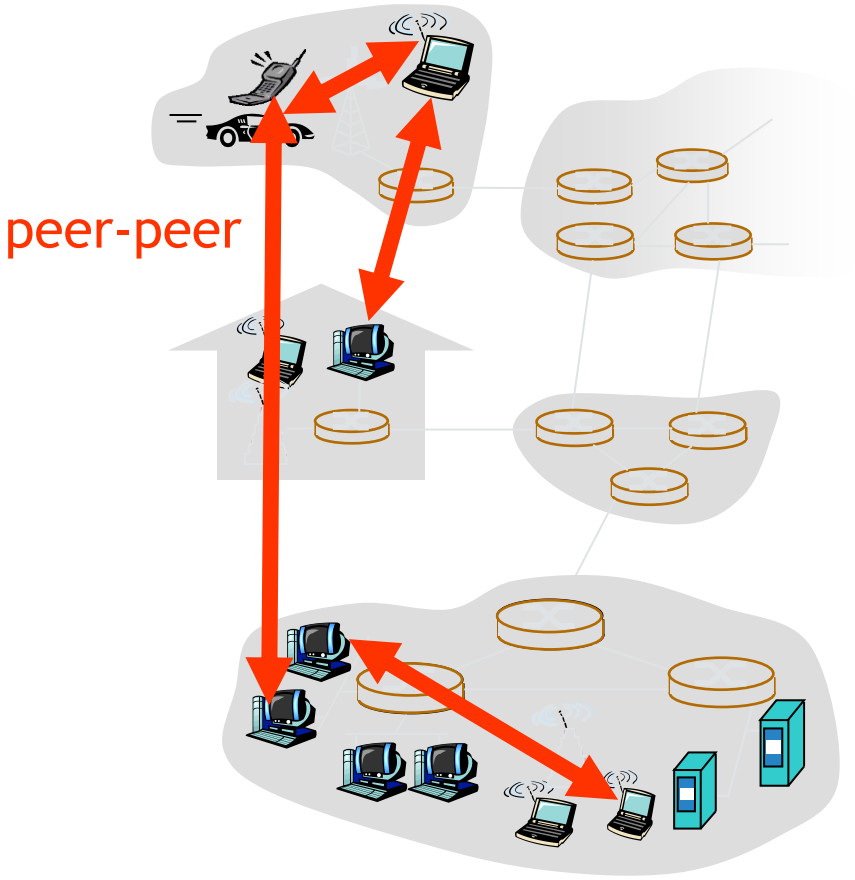
clients:

- communicate with server (know its IP address)
- may be intermittently connected
- may have dynamic IP addresses – why does a client have an IP address?
- do not communicate directly with each other

Pure P2P architecture

- *no* always-on server
- arbitrary end systems "directly" communicate
- peers are intermittently connected and change IP addresses could change
- Highly scalable but difficult to manage

  --why?



Systems programming: Processes communicating

Client process: process that initiates communication

Server process: process that waits to be contacted

- Processes in different hosts communicate by exchanging messages, packets are message fragments

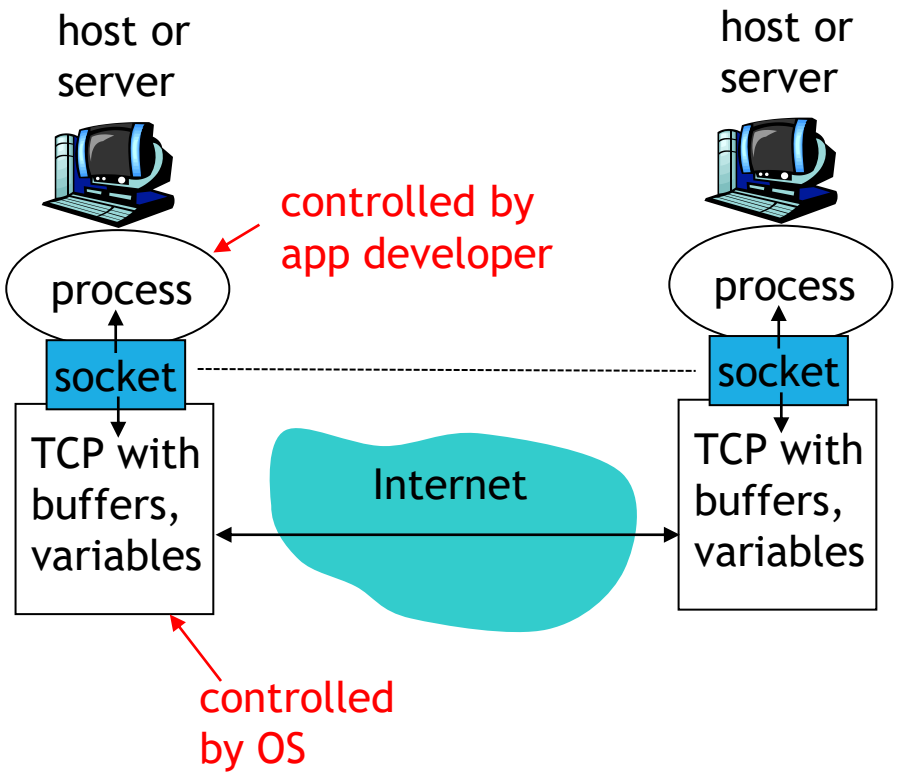
Abstraction:

- Socket – abstraction of a communication endpoint

- <picture>

Sockets

- Process sends/receives messages to/from its socket
- socket analogous to door
  - sending process shoves message out the door
  - sending process relies on transport infrastructure on other side of door which brings message to socket at receiving process



- API:
  - (1) choice of transport protocol
  - (2) ability to fix a few parameters
- TCP or UDP sockets

Addressing processes

- To receive messages, process must have an address or *identifier*

- Host device has unique 32-bit IP address

(well, could have >1, but ignore for now)

- *Q:* Does IP address of host on which process runs suffice for identifying the process?

Addressing processes

- To receive messages, process must have address or *identifier*
  - *Identifier includes both IP address and port numbers associated with process on host.*

- Host device has unique 32-bit IP address
  - Example port numbers:
    - HTTP server: 80
    - Mail server: 25
  - To send HTTP message to caesar.cs.umn.edu web server:
    - IP address: 128.119.245.12
    - Port number: 80

Internet transport protocols services

TCP service:

- ***Connection-oriented***: setup required between client and server processes
- ***Reliable*** *transport* between sending and receiving process
- *Does not provide:* timing, minimum throughput guarantees, security

UDP service:

- **Connectionless:** does **not** provide: **connectio**n setup, reliability, throughput guarantee, security
- **unreliable** data transfer between sending and receiving process

Message oriented

Q: why bother? Why is there a UDP?

UDP

- Packet is sent as soon as the host sends it!
- TCP adjusts rate of transmission based on congestion control signals