Network Transport Services

end host to end host communication services

- Connection-Oriented, Reliable Service
  - Mimic “dedicated link”
  - Messages delivered in correct order, without errors
  - Transport service aware of connection in progress
    - Stateful, some “state” information must be maintained
  - Require explicit connection setup and teardown
- Connectionless, Unreliable Service
  - Messages treated as independent
  - Messages may be lost, or delivered out of order
  - No connection setup or teardown, “stateless”

What transport service does an app need?

Data loss

- some apps (e.g., audio) can tolerate some loss
- other apps (e.g., file transfer, telnet) require 100% reliable data transfer

Timing

- some apps (e.g., Internet telephony, interactive games) require low delay to be "effective"

Throughput

❐ some apps (e.g., multimedia) require minimum amount of throughput to be "effective"

❐ other apps ("elastic apps") make use of whatever throughput they get

Security

❐ Encryption, data integrity, …

Internet apps: their protocols and transport protocols

Application	Application layer protocol	Underlying transport protocol
e-mail	smtp [RFC 821]	TCP
remote terminal access	telnet [RFC 854]	TCP
Web	http [RFC 2068]	TCP
file transfer	ftp [RFC 959]	TCP
streaming multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
remote file server	NSF	TCP or UDP
Internet telephony	proprietary (e.g., Vocaltec)	typically UDP

Processes communicating

Process: program running within a host.

- within same host, two processes communicate using inter-process communication (defined by OS).
- processes in different hosts communicate by exchanging messages

Client process: process that initiates communication

Server process: process that waits to be contacted

❒ Note: applications with P2P architectures have client processes & server processes

Network Applications: some jargon

- A process is a program that is running within a host.
- Within the same host, two processes communicate with interprocess communication defined by the OS.
- Processes running in different hosts communicate with an application-layer protocol

- A user agent is an interface between the user and the network application.
  - Web: browser
  - E-mail: mail reader
  - streaming audio/video: media player

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

Public-domain protocols:

- defined in RFCs
- allows for interoperability
- e.g., HTTP, SMTP, BitTorrent

Proprietary protocols:

- e.g., Skype, ppstream

Application Programming Interface

API: application programming interface

- defines interface between application and transport layer
- socket: Internet API
  - two processes communicate by sending data into socket, reading data out of socket

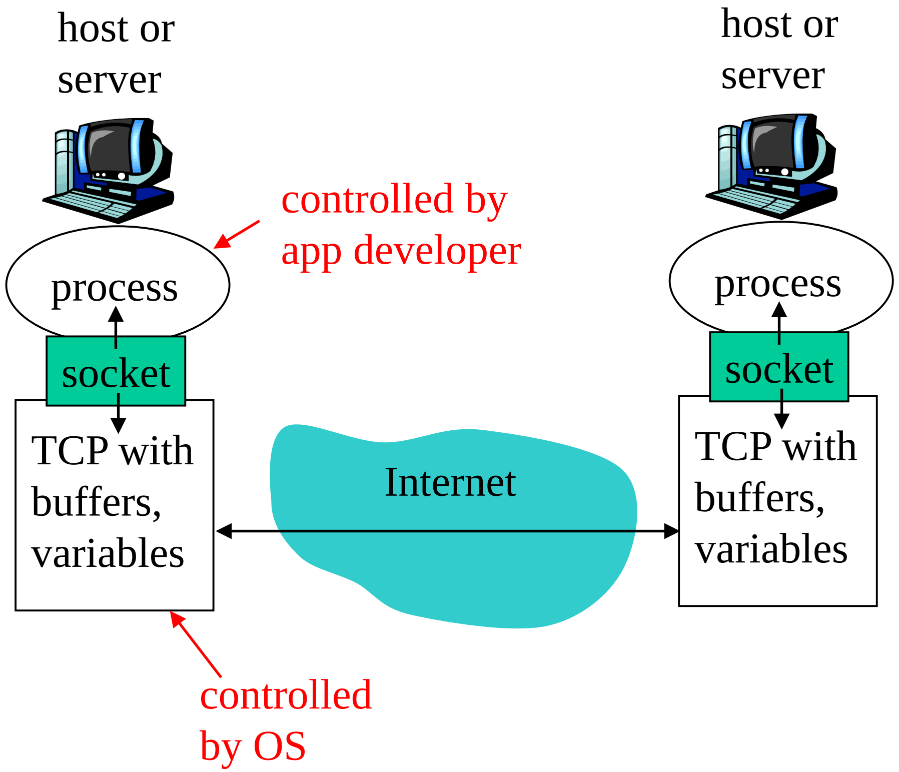
Q: how does a process "identify" the other process with which it wants to communicate?

- IP address of host running other process
- "port number" - allows receiving host to determine to which local process the message should be delivered

❒ API: (1) choice of transport protocol; (2) ability to fix a few parameters (lots more on this later)

Sockets

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


host or server
process
socket
TCP with buffers, variables
controlled by app developer
Internet
controlled by OS
host or server
process
socket
TCP with buffers, variables

Application Structure

Internet applications distributed in nature!

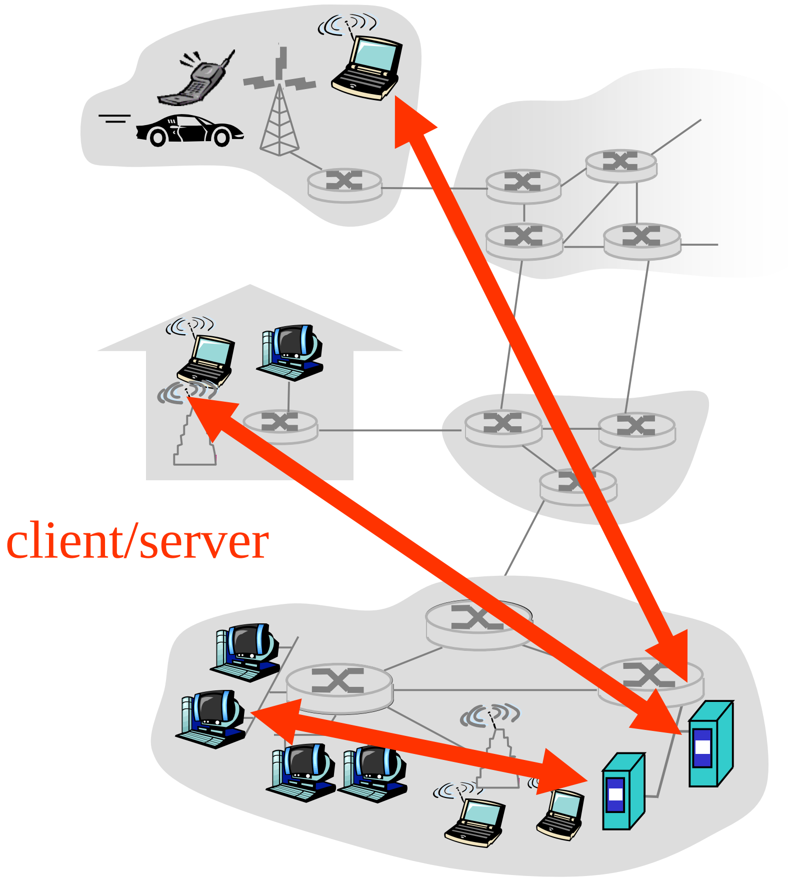
- Set of communicating application-level processes (usually on different hosts) provide/implement services

Programming Paradigms:

- Client-Server Model: Asymmetric
  - Server: offers service via well defined "interface"
  - Client: request service
  - Example: Web; cloud computing
- Peer-to-Peer: Symmetric
  - Each process is an equal
  - Example: telephone, p2p file sharing (e.g., Kazaar)
- Hybrid of client-server and P2P

All require transport of "request/reply", sharing of data!

Client-server architecture



server:

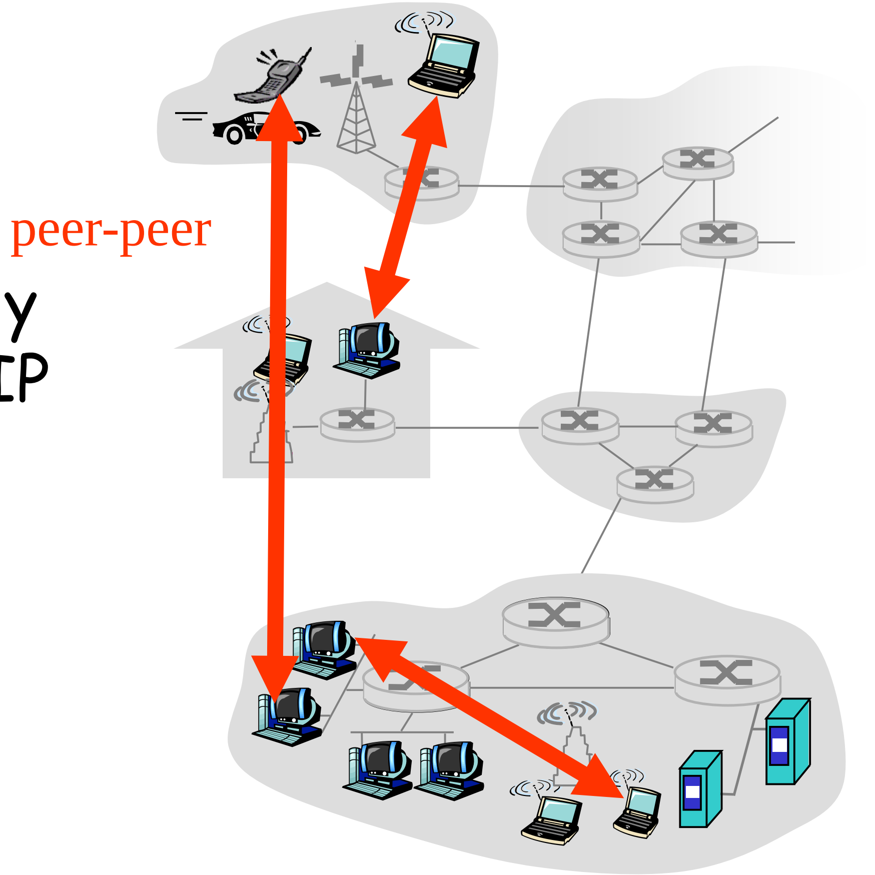
- always-on host
- permanent IP address
- server farms for scaling

clients:

- communicate with server
- may be intermittently connected
- may have dynamic IP addresses
- 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



Highly scalable but
difficult to manage

Peer-to-Peer Paradigm

- How do we implement peer-to-peer model?
- Is email peer-to-peer or client-server application?
- How do we implement peer-to-peer using client-server model?

Difficulty in implementing "pure" peer-to-peer model?

- How to locate your peer?
  - Centralized "directory service:" i.e., white pages
    - Napters
  - Unstructured: e.g., "broadcast" your query: namely, ask your friends/neighbors, who may in turn ask their friends/neighbors,
    - Freenet
  - Structured: Distributed hashing table (DHT)

Client-Server Paradigm Recap

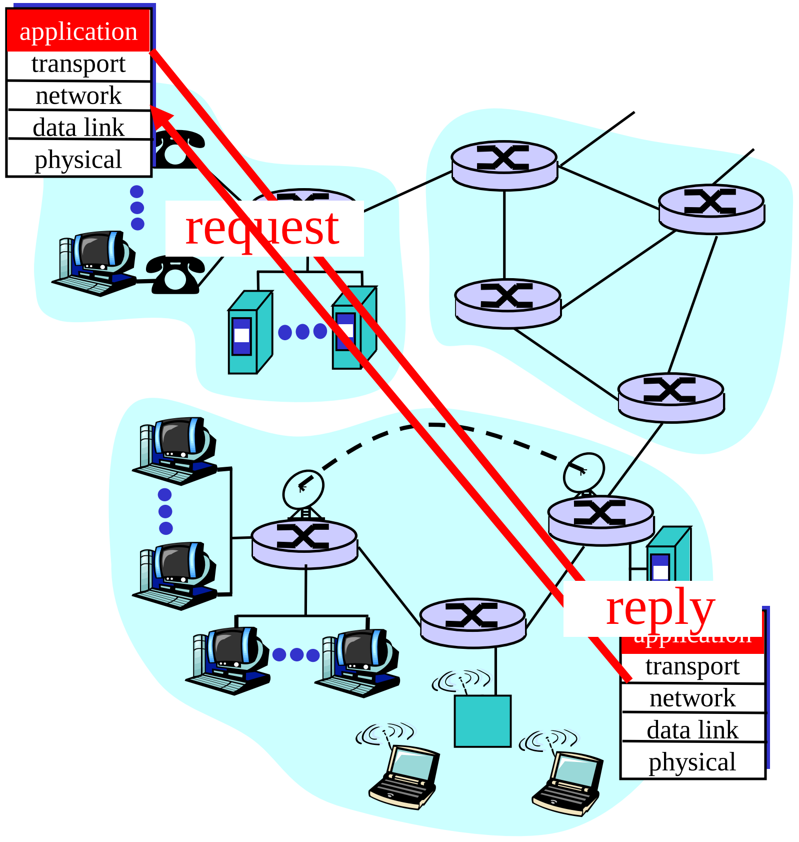
Typical network app has two pieces: *client* and *server*

Client:

- initiates contact with server ("speaks first")
- typically requests service from server,
- for Web, client is implemented in browser; for e-mail, in mail reader

Server:

- provides requested service to client
- e.g., Web server sends requested Web page, mail server delivers e-mail



Client-Server: The Web Example

some jargon

- Web page:
  - consists of "objects"
  - addressed by a URL
- Most Web pages consist of:
  - base HTML page, and
  - several referenced objects.
- URL has two components: host name and path name:

- User agent for Web is called a browser:
  - MS Internet Explorer
  - Netscape Communicator
- Server for Web is called Web server:
  - Apache (public domain)
  - MS Internet Information Server

`www.someSchool.edu/someDept/pic.gif`