Introduction to the Application Layer

Computer Networks

Spring 2013

Intro to Application Layer Outline

- Current Application Layer Protocols
- Creating an Application
- Application Architectures
  - Client-Server
  - P2P
  - Hybrid
- Processes, Addressing and Sockets
- Transport Layer Services

Goals

- Conceptual and implementation aspects of application protocols
- Popular application layer protocols:
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
  - RTP, SIP

Application Layer Observations

- Important to distinguish application programs from application protocols.
  - Browsers are examples of web client programs that use both HTTP and DNS which are application protocols.
- Since many application protocols follow the request/response communication pattern, TCP is frequently used to support these protocols.
- Many application layer protocols have a companion specification language that defines the format of the data exchanged.

Examples: MIME for SMTP, HTML for HTTP

Popular Network Applications

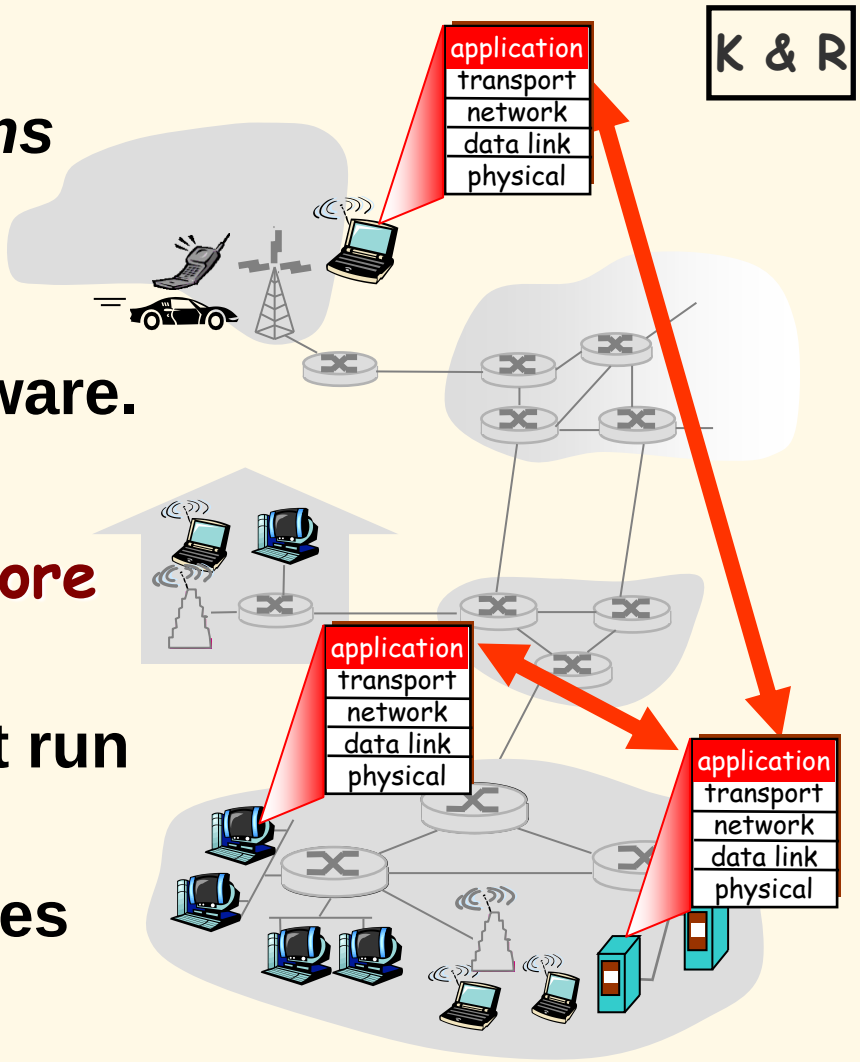
- e-mail
- web
- instant messaging
- remote login
- P2P file sharing
- multi-user network games
- streaming stored video clips
- social networks
- voice over IP
- real-time video conferencing
- grid computing

Creating a Network App

Write programs to

- run on (different) *end systems*
- communicate over network

Example: Web server software communicates with browser software.



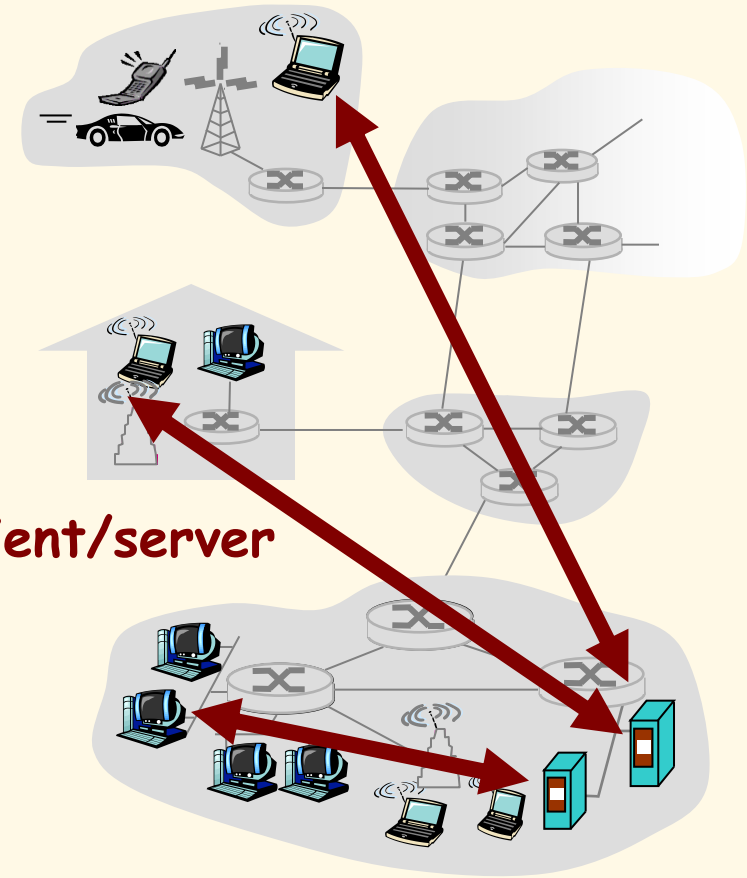
No need to write software for core network devices.

- Network-core devices do not run user applications.
- Apps on end systems enables rapid app development, propagation.

Application Architectures

- Client-server (CS)
  - Including data centers and cloud computing
- Peer-to-peer (P2P)
- Hybrid of client-server and P2P

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

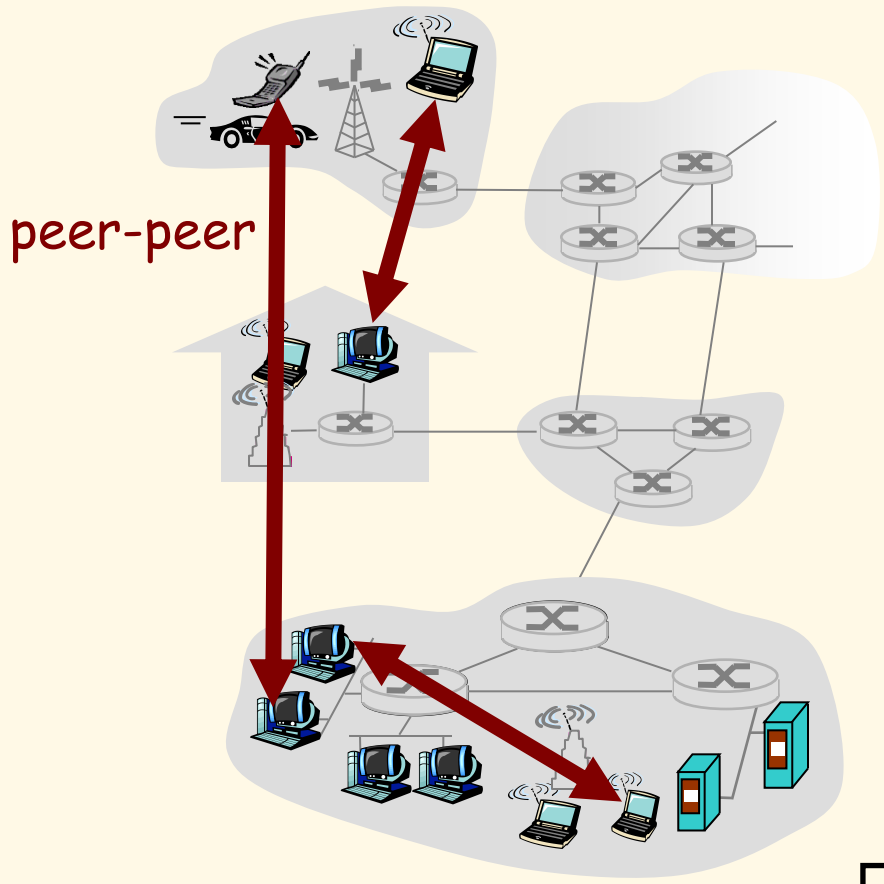
K & R

Server Example: Google Data Centers

- Estimated cost: $600M
- Google spent $2.4B in 2007 on new data centers
- Each data center uses 50-100 megawatts of power.

Pure P2P Architecture

- *no* always-on server
- arbitrary end systems communicate directly.
- peers are intermittently connected and change IP addresses.



Highly scalable but difficult to manage

K & R

Hybrid: Client-Server and P2P

- Skype
  - Voice-over-IP (VoIP) P2P application
  - centralized server: finding address of remote party
  - client-client connection: often direct (not through server)
- Instant Messaging
  - chatting between two users is P2P
  - centralized service: client presence detection/location
    - user registers its IP address with central server when it comes online.
    - user contacts central server to find IP addresses of buddies.

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**.

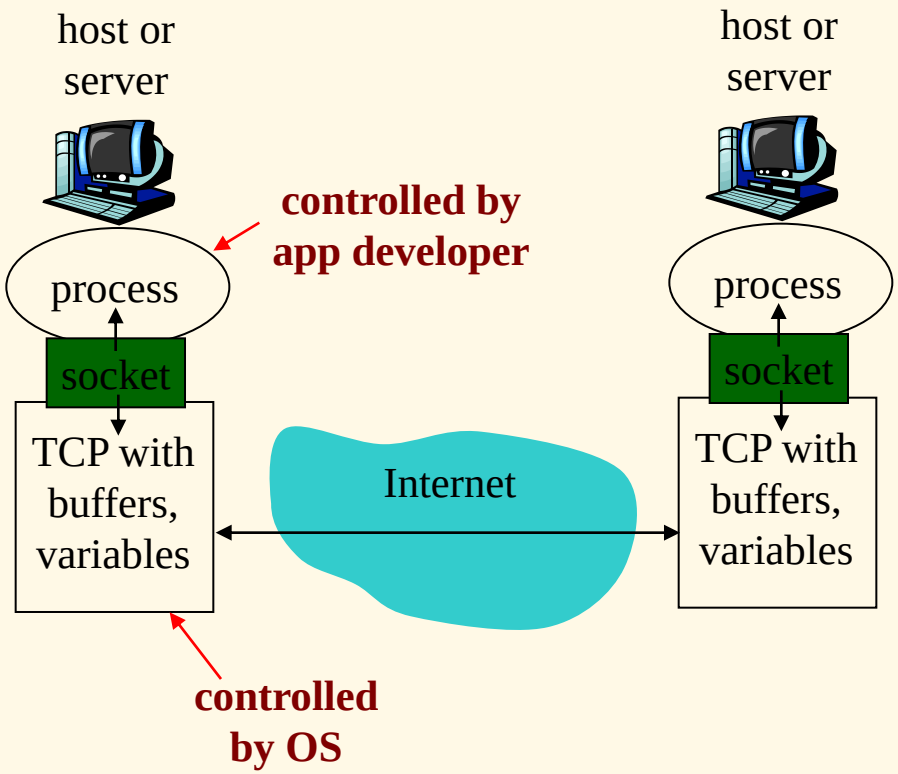
Client process: process that initiates communication.

Server process: process that waits to be contacted.

- Note: applications with P2P architectures have client processes & server processes.

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.



- API: (1) choice of transport protocol; (2) ability to fix a few parameters (see Sockets lecture)

K & R

Addressing Processes

- To receive messages, process must have an *identifier.*
- Host device has unique 32-bit IP address
- Exercise: use `ipconfig` from command prompt to get your IP address (Windows)
- `ifconfig` (under Linux)

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

  A: No, *many* processes can be running on same machine.
- Identifier includes both IP address and port numbers associated with process on host.
- Example port numbers:
  - HTTP server: 80
  - Mail server: 25

App-Layer Protocol Defines

- Types of messages exchanged,
  - e.g., request, response
- Message syntax:
  - specific 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

K & R

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, privacy …

CommonTransport Service App Requirements

Application	Data loss	Throughput	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps video:10kbps-5Mbps	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
interactive games	loss-tolerant	few kbps up	yes, 100's msec
instant messaging	no loss	elastic	yes and no

K & R

Internet Transport Protocols Services

TCP service:

- *connection-oriented:* setup required between client and server processes.
- *reliable transport* between sending and receiving process
- *flow control:* sender cannot overwhelm receiver.
- *congestion control:* throttle sender when network overloaded.
- *does not provide:* timing, minimum throughput guarantees, security

UDP service:

- unreliable data transfer between sending and receiving process
- does not provide: connection setup, reliability, flow control, congestion control, timing, throughput guarantee, or security.

Q: Why bother? Why is there a UDP?

Internet Apps: Application, Transport Protocols

Application	Application layer protocol	Underlying transport protocol
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP (eg Youtube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	typically UDP

K & R

Intro to Application Layer Summary

- Current Application Layer Protocols
- Creating an Application
- Application Architectures
  - Client-Server
  - P2P
  - Hybrid
- Processes, Addressing and Sockets
- Transport Layer Services
  - TCP versus UDP