

# Computer Networks

## Network Applications

Based on Computer Networking, 3<sup>rd</sup> Edition by Kurose and Ross

Stan Kurkovsky

### Network applications

#### Sample applications

- E-mail
- Web
- Instant messaging
- Remote login
- P2P file sharing
- Multi-user network games
- Streaming stored video clips
- Internet telephone
- Real-time video conference
- Massive parallel computing
- ...

#### Typical architectures

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

Stan Kurkovsky

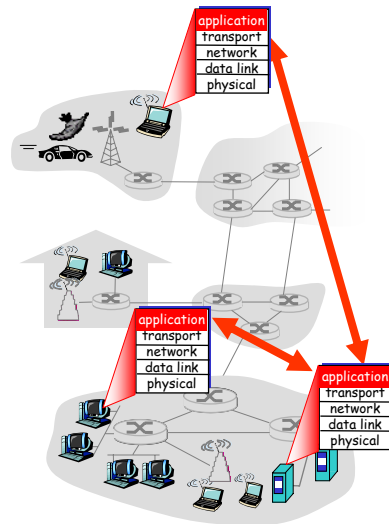
## Creating a network application

### Write programs that

- run on different end systems and
- communicate over a network.
- e.g., Web: Web server software communicates with browser software

### Little software written for devices in network core

- network core devices do not run user application code
- application on end systems allows for rapid app development, propagation



Stan Kurkovsky

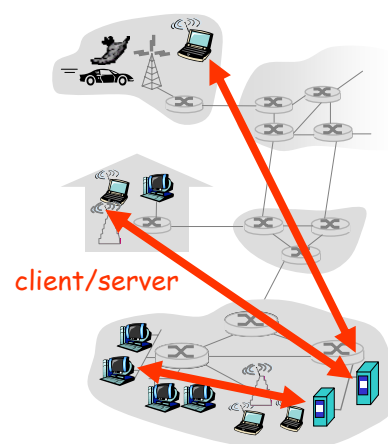
## Client-server architecture

### 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



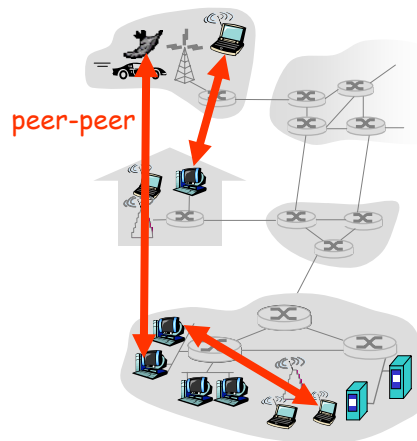
Stan Kurkovsky

## Pure P2P architecture

- no always-on server
- arbitrary end systems directly communicate
- peers are intermittently connected and change IP addresses
- example: Gnutella

Highly scalable

But difficult to manage



Stan Kurkovsky

## Hybrid of client-server and P2P

### Napster

- File transfer P2P
- File search centralized:
  - Peers register content at central server
  - Peers query same central server to locate content

### 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
- Presence detection/location centralized:
  - User registers its IP address with central server when it comes online
  - User contacts central server to find IP addresses of buddies

Stan Kurkovsky

## 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

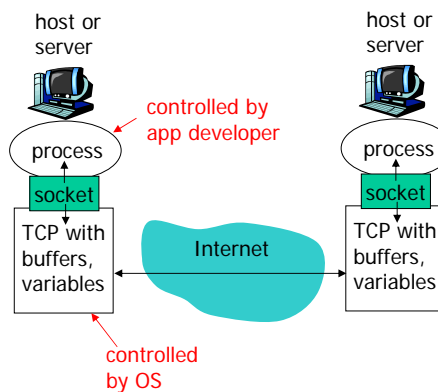
**Server process:** process that waits to be contacted

- Note: applications with P2P architectures have client processes & server processes
- Peers in P2P architectures have both client and server functionalities
  - Peers are sometimes called *servents* (*server* and *client*)

Stan Kurkovsky

## 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



- Application programming interface (API):
  - (1) choice of transport protocol
  - (2) ability to fix a few parameters

Stan Kurkovsky

## Addressing processes

- For a process to receive messages, it must have an identifier
- A host has a unique 32-bit **IP address**
- **Q:** does the IP address of the host on which the process runs suffice for identifying the process?
- **Answer:** No, many processes can be running on same host
- Identifier includes both the IP address and **port numbers** associated with the process on the host.
- Example port numbers:
  - HTTP server: 80
  - Mail server: 25
- to send HTTP message to gaia.cs.umass.edu web server:
  - **IP address:** 128.119.245.12
  - **Port number:** 80

Stan Kurkovsky

## Application-layer protocol

- **Application-layer protocols define**
  - Types of messages exchanged, e.g., request & response messages
  - Syntax of message types: what fields in messages & how fields are delineated
  - Semantics of the fields, i.e., 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
- **Proprietary protocols:**
  - e.g., KaZaA, Skype

Stan Kurkovsky

## What transport service does an application 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"

### Bandwidth

- some apps (e.g., multimedia) require minimum amount of bandwidth to be "effective"
- other apps ("elastic apps") make use of whatever bandwidth they get

Stan Kurkovsky

## Transport service requirements of common applications

Application	Data loss	Bandwidth	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

Stan Kurkovsky

## 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 won't overwhelm receiver
- *congestion control*: throttle sender when network overloaded
- *does not provide*: timing, minimum bandwidth guarantees

### UDP service:

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

Q: why bother? Why is there a UDP?

Stan Kurkovsky

## Internet applications: application, transport protocols

<u>Application</u>	<u>Application layer protocol</u>	<u>Underlying transport protocol</u>
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	proprietary (e.g. RealNetworks)	TCP or UDP
Internet telephony	proprietary (e.g., Vonage, Dialpad)	typically UDP

Stan Kurkovsky