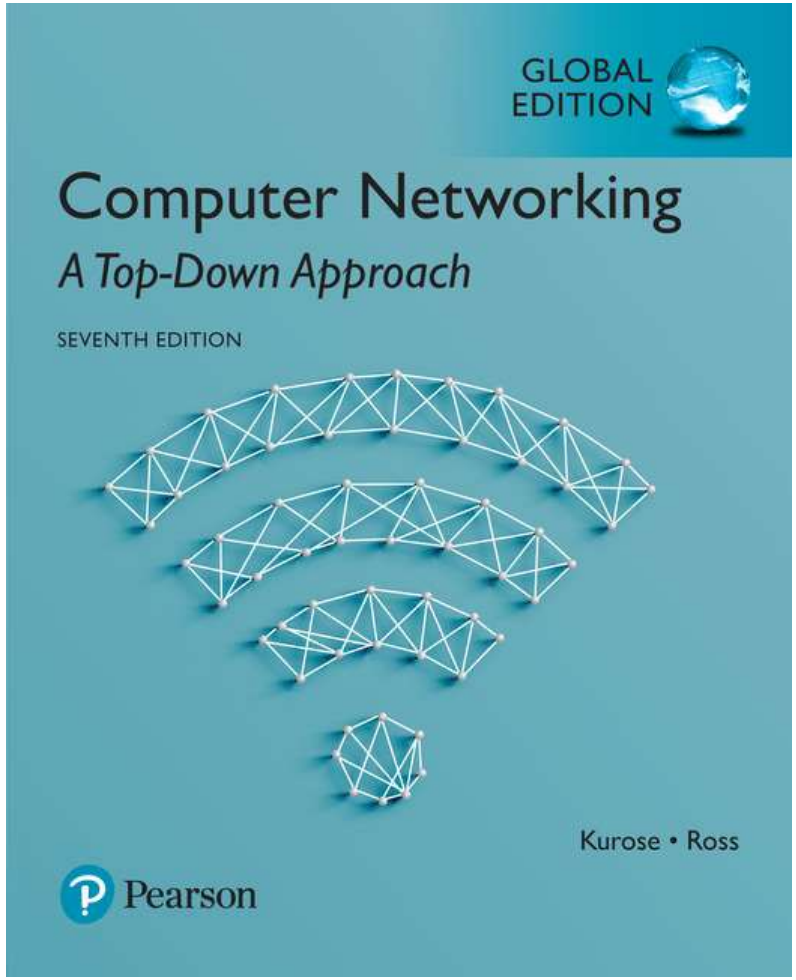
제7강
응용 계층

Computer Networking: A Top Down Approach

컴퓨터 네트워크
(2019년 1학기)

박승철교수

한국기술교육대학교
컴퓨터공학부



Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

2.7 socket programming with UDP and TCP

Pre-study Test :

1) 다음 중 응용 계층 프로토콜이 아닌 것은?

① TCP
② HTTP
③ SMTP
④ DNS

2) 다음 중 응용 계층 프로토콜을 반드시 지원할 필요가 없는 장치는?

① 클라이언트 컴퓨터
② 서버 컴퓨터
③ 스마트폰
④ 라우터

3) 다음 중 클라이언트-서버 구조의 특징이 아닌 것은?

① 일반적으로 서버는 영구적인 주소를 가진다.
② 일반적으로 서버는 항상 동작하고 있다.
③ 일반적으로 클라이언트에 의해 서비스가 개시된다.
④ 일반적으로 클라이언트와 클라이언트간에 직접 통신이 보장된다.

4) 다음 중 P2P 구조의 특징이 아닌 것은?

① 항상 동작 중인 서버가 존재하지 않는다.
② 피어(peer)는 항상 고정된 IP 주소를 가진다.
③ 피어는 서버와 클라이언트 역할을 함께 수행한다.
④ 일반적으로 피어는 다수의 피어와 통신한다.

5) 응용 프로세스(process)가 다른 응용 프로세스와 메시지를 교환하는 통로를 무엇이라 하는가?

① 포트(port)
② IP 주소
③ 소켓(socket)
④ API(Application Programming Interface)

6) 인터넷상의 특정 프로세스(process)는 무엇으로 식별되는가?

① IP 주소
② Port 번호
③ URL(Uniform Resource Locator)
④ IP 주소 + Port 번호

7) 다음 중 일반적으로 응용 계층 프로토콜이 정의하지 않는 것은?

① 메시지 유형(type)
② 메시지 오류 복구(Error Recovery)
③ 메시지 구문(syntax)
④ 메시지 의미(semantics).

8) 다음 중 반드시 TCP만을 사용해야 하는 응용 서비스는?

① 웹 서비스
② 비디오 스트리밍
③ 인터넷 전화
④ 인터넷 방송

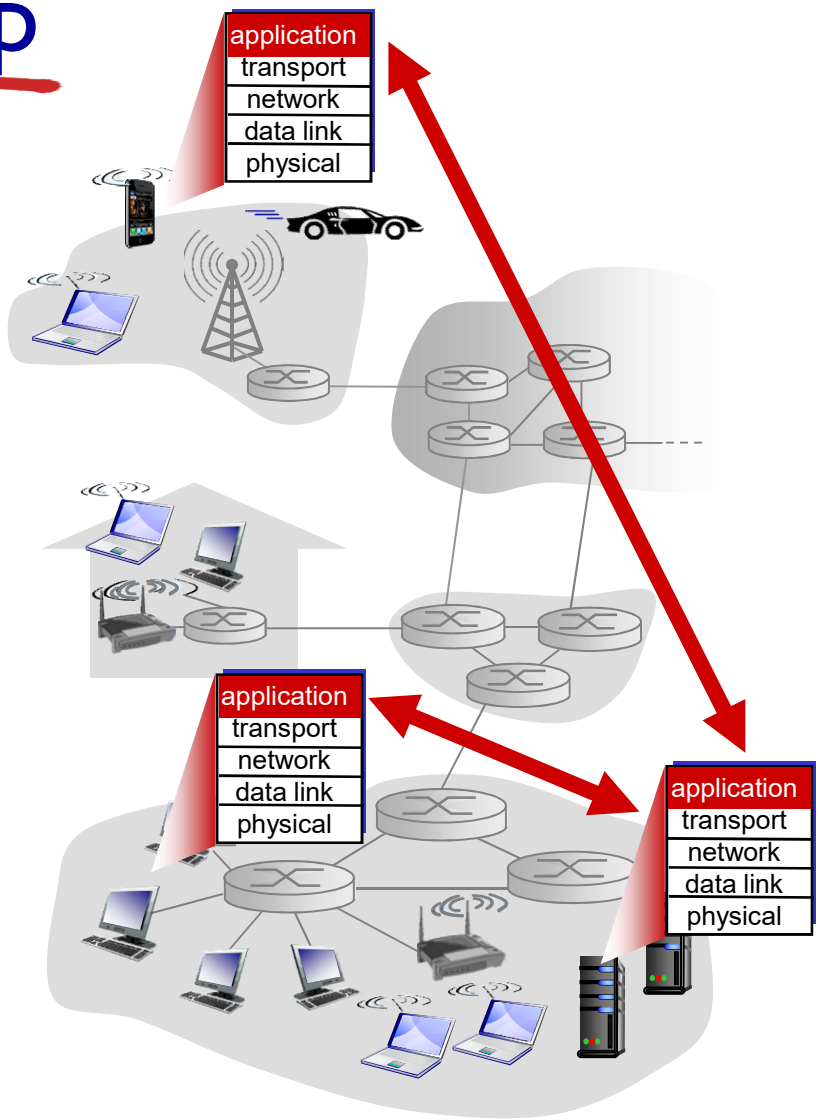
Some network apps

- e-mail
- web
- text messaging
- remote login
- P2P file sharing
- multi-user network games
- streaming stored video (YouTube, Hulu, Netflix)
- voice over IP (e.g., Skype)
- real-time video conferencing
- social networking
- search
- …
- …

Creating a network app

write programs that:

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



no need to write software for network-core devices

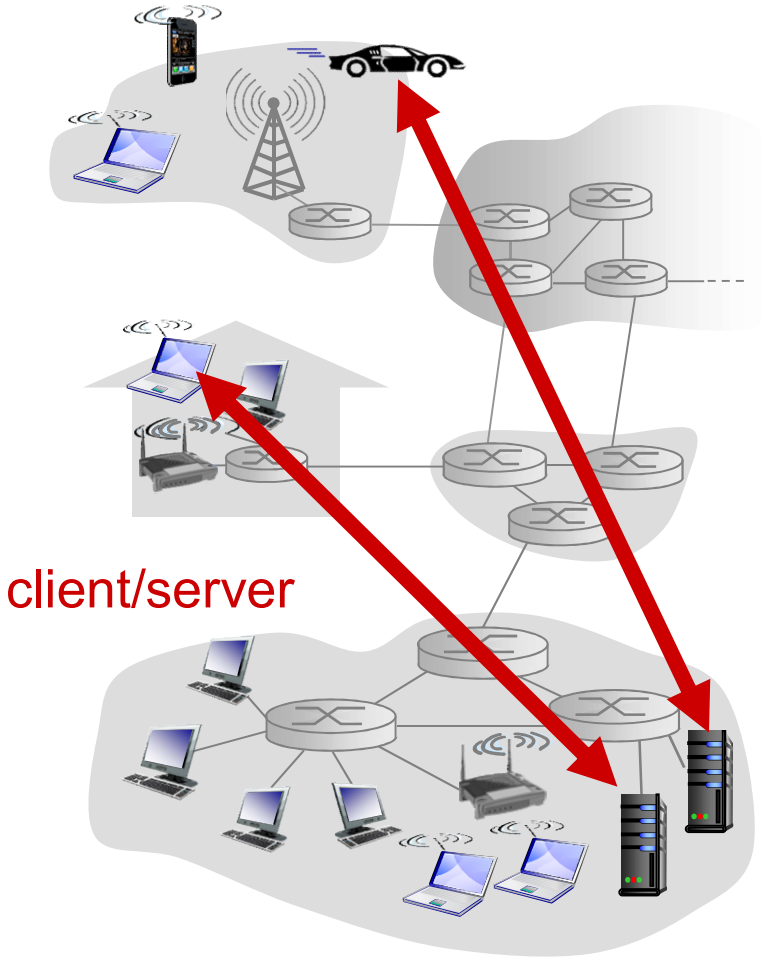
- network-core devices do not run user applications
- applications on end systems allows for rapid app development, propagation

Application architectures

possible structure of applications:

- client-server
- peer-to-peer (P2P)

Client-server architecture



server:

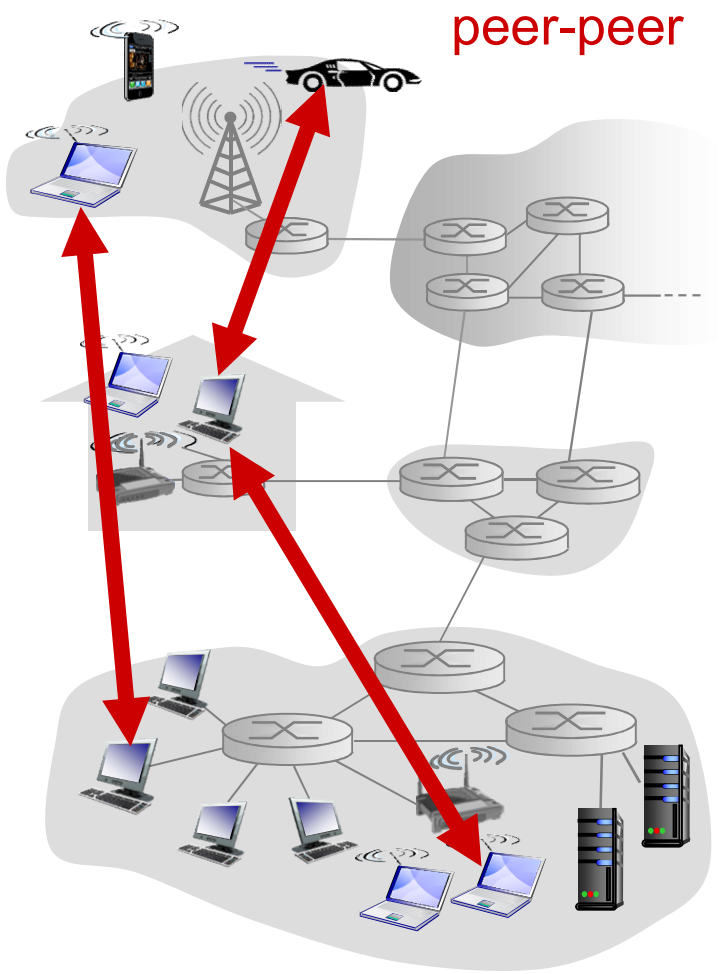
- always-on host
- permanent IP address
- data centers for scaling

clients:

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

P2P architecture

- *no* always-on server
- arbitrary end systems directly communicate
- peers request service from other peers, provide service in return to other peers
  - *self scalability* – new peers bring new service capacity, as well as new service demands
- peers are intermittently connected and change IP addresses
  - complex management



P2P architecture

- 미래에 직면할 과제
  - ISP 우호적 – 비대칭적 대역폭을 제공하는 ISP 네트워크에 우호적으로 설계
  - 보안(security) – 참여자 전체의 보안성 확보
  - 보상(incentives) – 자발적 참여 유도

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

clients, servers

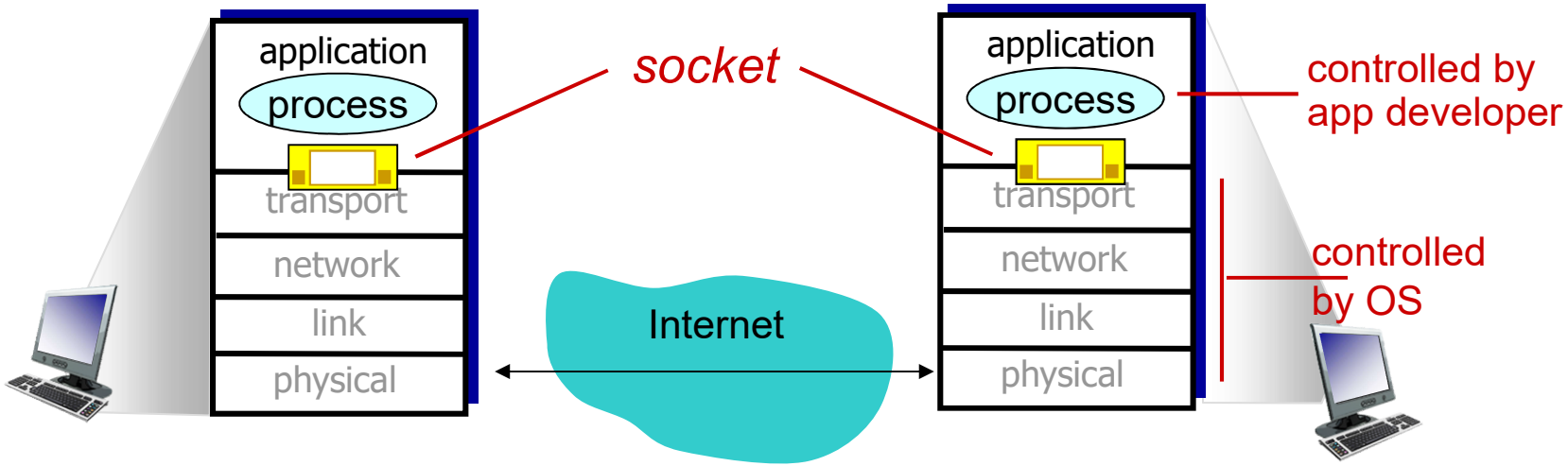
client process: process that initiates communication

server process: process that waits to be contacted

- aside: 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 to deliver message to socket at receiving process



Addressing processes

- to receive messages, process must have *identifier*
- host device has unique 32-bit IP address
- *Q:* does IP address of host on which process runs suffice for identifying the process?
  - *A:* no, *many* processes can be running on same host

- *identifier* includes both IP address and port numbers associated with process on 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
- more shortly…

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

open protocols:

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

proprietary protocols:

- e.g., Skype

What transport service does an app need?

data integrity

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

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, …

Transport service requirements: common apps

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	
interactive games	loss-tolerant	few kbps up	yes, few secs
text messaging	no loss	elastic	yes, 100's msec yes and no

Internet transport protocols services

TCP service:

- *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 throughput guarantee, security
- *connection-oriented:* setup required between client and server processes

UDP service:

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

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 (e.g., YouTube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	TCP or UDP

Securing TCP

TCP & UDP

- no encryption
- cleartext passwds sent into socket traverse Internet in cleartext

SSL

- provides encrypted TCP connection
- data integrity
- end-point authentication

SSL is at app layer

- apps use SSL libraries, that "talk" to TCP

SSL socket API

- cleartext passwords sent into socket traverse Internet encrypted
- see Chapter 8

응용 계층 프로토콜

- 정의 내용
  - 교환 메시지 유형(types)
  - 메시지의 문법(syntax)
  - 메시지의 각 필드의 의미(semantics)
  - 메시지 교환 방법과 시간에 관한 규칙

응용 프로토콜과 응용 프로그램

- 포함 관계 ?

After-study Test :

1) 다음 중 응용 계층 프로토콜이 아닌 것은?

① TCP
② HTTP
③ SMTP
④ DNS

2) 다음 중 응용 계층 프로토콜을 반드시 지원할 필요가 없는 장치는?

① 클라이언트 컴퓨터
② 서버 컴퓨터
③ 스마트폰
④ 라우터

3) 다음 중 클라이언트-서버 구조의 특징이 아닌 것은?

① 일반적으로 서버는 영구적인 주소를 가진다.
② 일반적으로 서버는 항상 동작하고 있다.
③ 일반적으로 클라이언트에 의해 서비스가 개시된다.
④ 일반적으로 클라이언트와 클라이언트간에 직접 통신이 보장된다.

4) 다음 중 P2P 구조의 특징이 아닌 것은?

① 항상 동작 중인 서버가 존재하지 않는다.
② 피어(peer)는 항상 고정된 IP 주소를 가진다.
③ 피어는 서버와 클라이언트 역할을 함께 수행한다.
④ 일반적으로 피어는 다수의 피어와 통신한다.

5) 응용 프로세스(process)가 다른 응용 프로세스와 메시지를 교환하는 통로를 무엇이라 하는가?

① 포트(port)
② IP 주소
③ 소켓(socket)
④ API(Application Programming Interface)

6) 인터넷상의 특정 프로세스(process)는 무엇으로 식별되는가?

① IP 주소
② Port 번호
③ URL(Uniform Resource Locator)
④ IP 주소 + Port 번호

7) 다음 중 일반적으로 응용 계층 프로토콜이 정의하지 않는 것은?

① 메시지 유형(type)
② 메시지 오류 복구(Error Recovery)
③ 메시지 구문(syntax)
④ 메시지 의미(semantics).

8) 다음 중 반드시 TCP만을 사용해야 하는 응용 서비스는?

① 웹 서비스
② 비디오 스트리밍
③ 인터넷 전화
④ 인터넷 방송