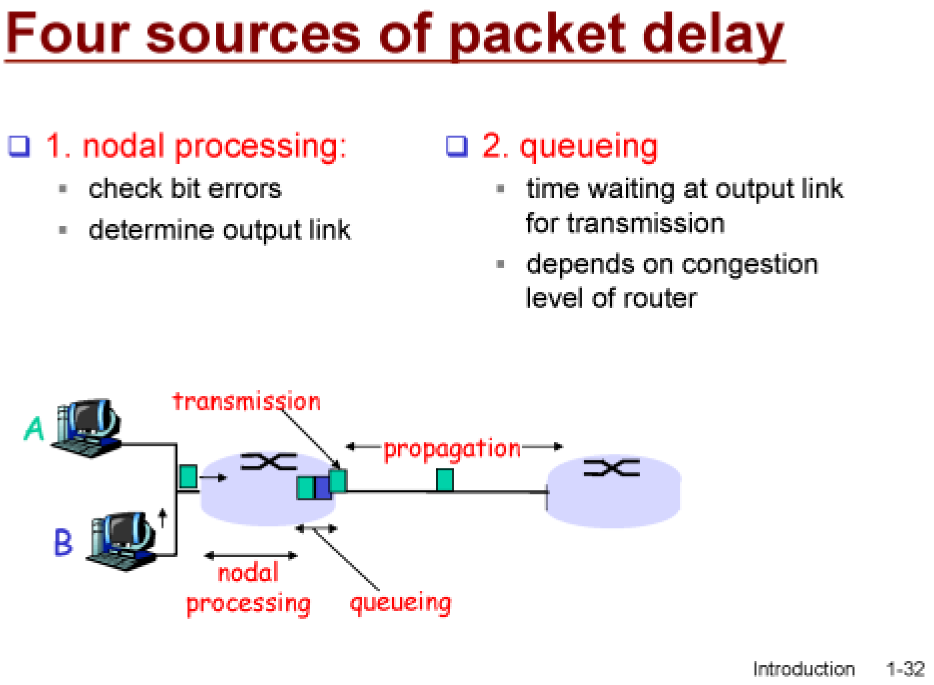
컴퓨터 네트워크 기본 2

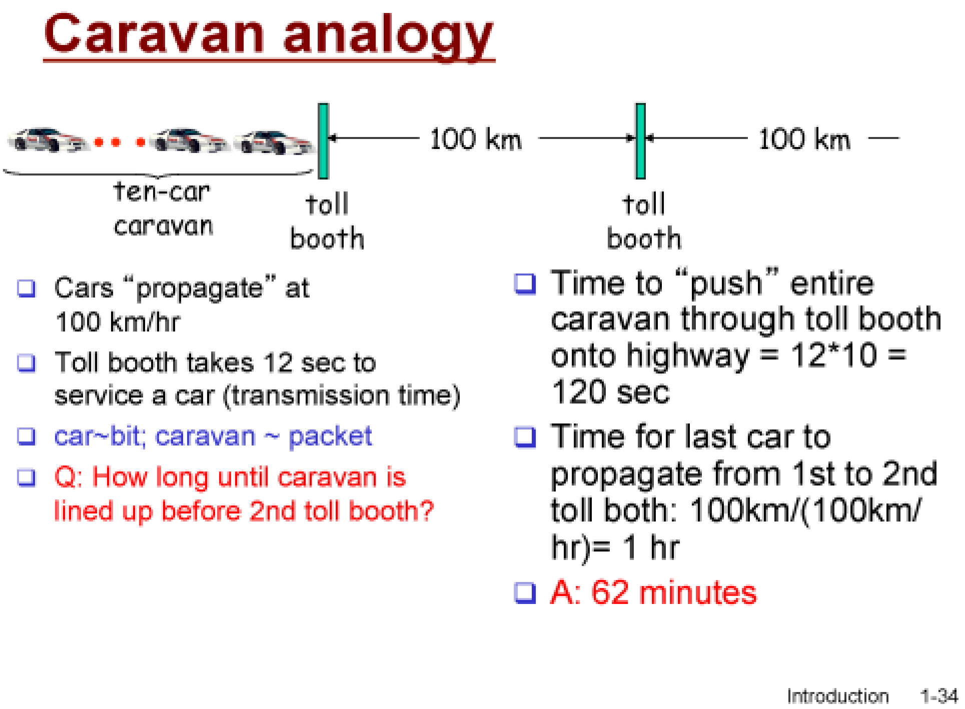
강의날짜	@2022/09/26
작성일시	@2022년 9월 26일 오후 10:11
편집일시	@2022년 9월 26일 오후 10:34
분야	네트워크
공부유형	스터디 그룹
복습	□
태그	



Four sources of packet delay

- 1. nodal processing:
  - check bit errors
  - determine output link
- 2. queueing
  - time waiting at output link for transmission
  - depends on congestion level of router

Introduction 1-32

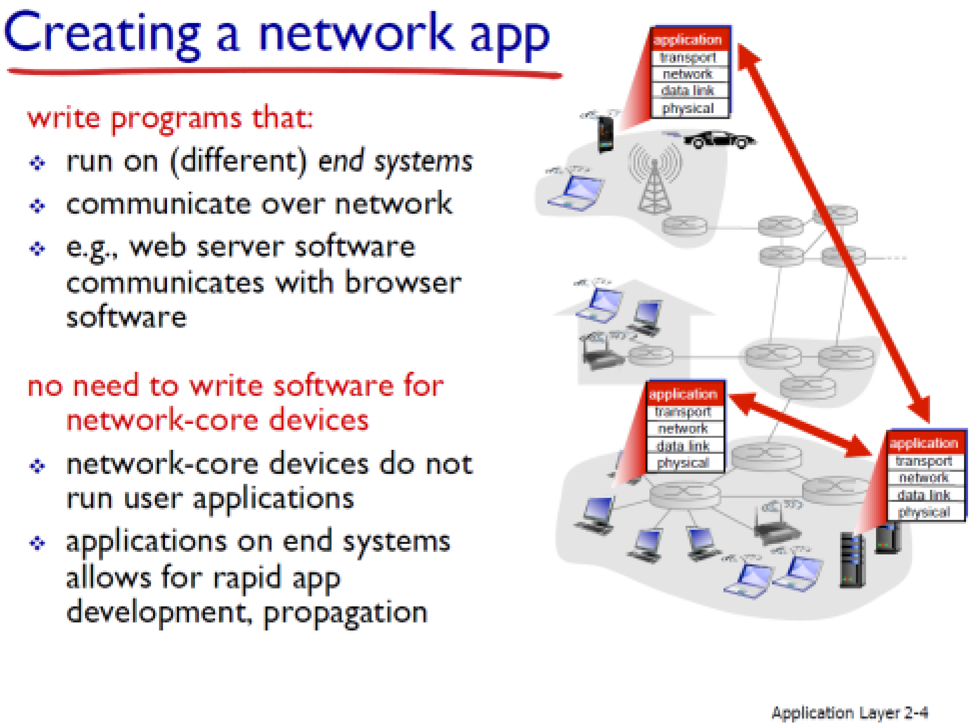


Caravan analogy

- ❑ Cars "propagate" at 100 km/hr
- ❑ Toll booth takes 12 sec to service a car (transmission time)
- ❑ car~bit; caravan ~ packet
- ❑ Q: How long until caravan is lined up before 2nd toll booth?
- ❑ Time to "push" entire caravan through toll booth onto highway = 12*10 = 120 sec
- ❑ Time for last car to propagate from 1st to 2nd toll both: 100km/(100km/hr)= 1 hr
- ❑ A: 62 minutes

Introduction 1-34

- package switching : 뒤에 것이 올때까지 다 기렸다가 출발함



Creating a network app

write programs that:

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

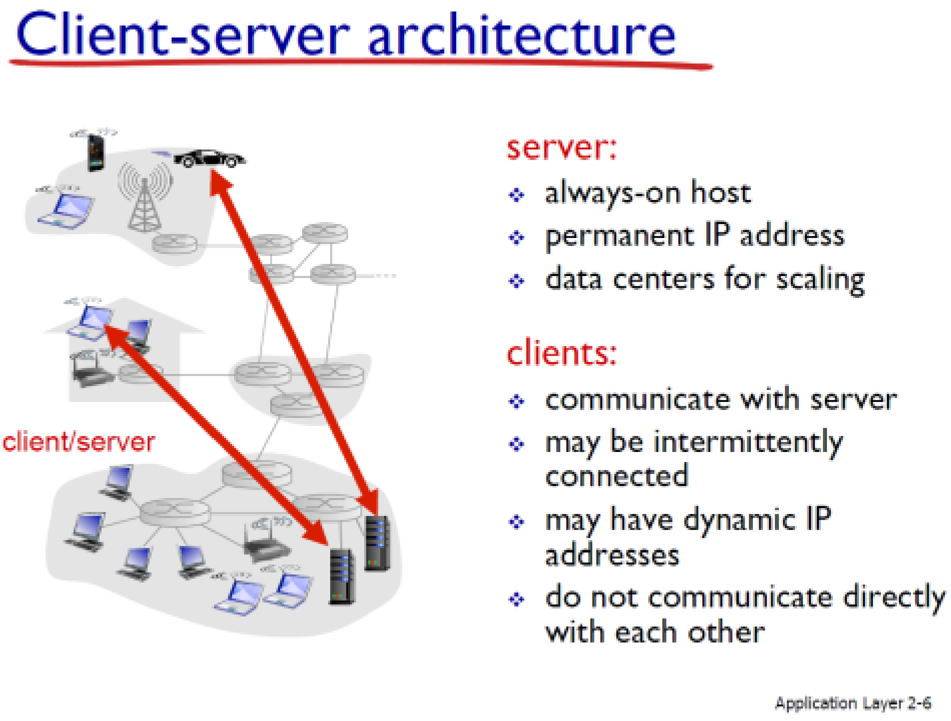
no need to write software for network-core devices

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

Application Layer 2-4

- APP : HTTP
- transport : TCP / UDP

- network : IP
- data link : WIFI LTE
- physical
- 네트워크 계층은 엣지에만 존재하는 것
- 하위계층이 상위계층에게 기능 제공



Client-server architecture

server:

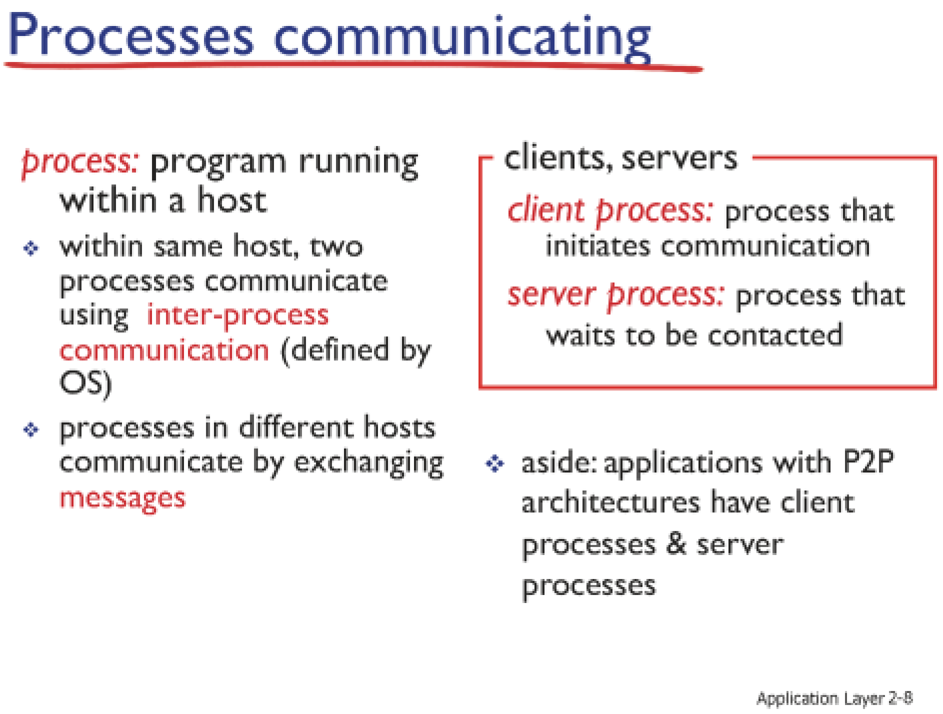
- always-on host
- permanent IP address
- data centers for scaling

clients:

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

Application Layer 2-6

- server : 각자 자기 주소를 가지고 있어야함 : 고정된 IP 주소
- client : 고정되지 않은 IP 주소



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

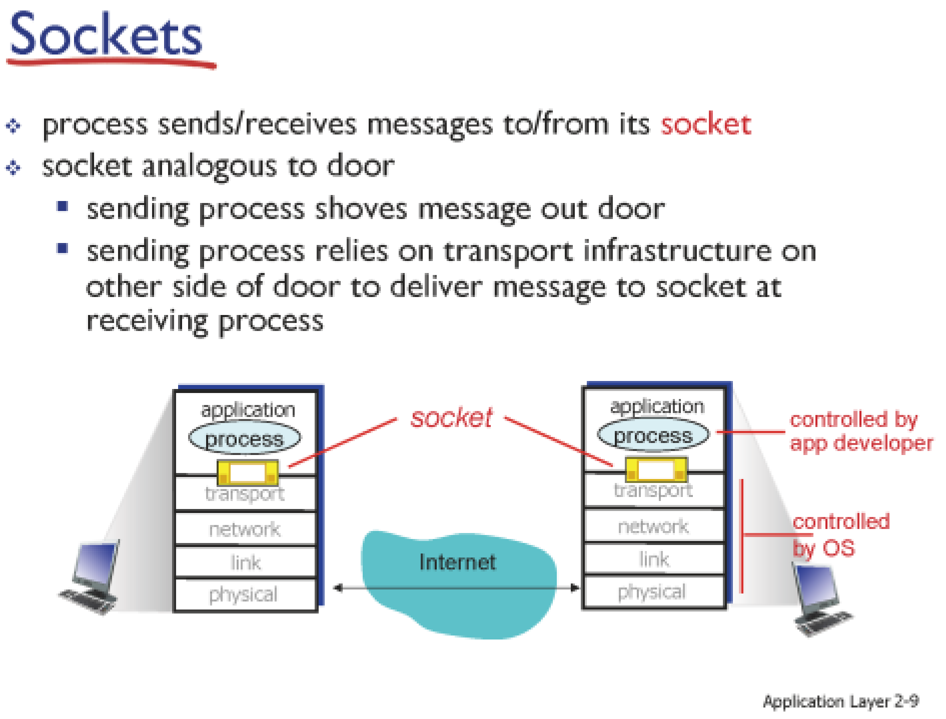
clients, servers

client process: process that initiates communication

server process: process that waits to be contacted

- aside: applications with P2P architectures have client processes & server processes

Application Layer 2-8

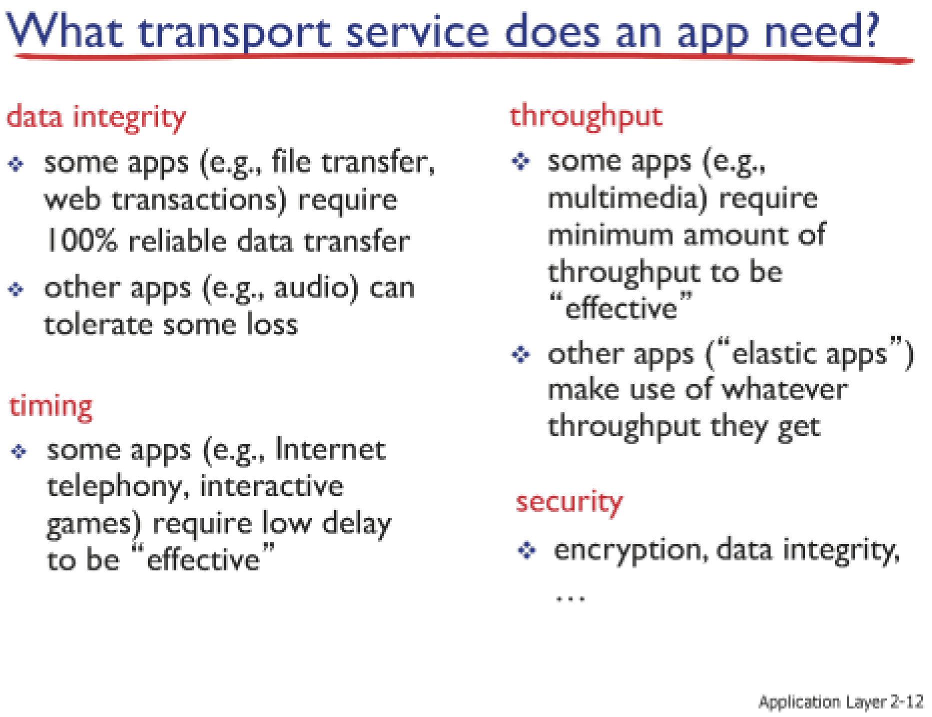


Sockets

- 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

Application Layer 2-9

- url주소 - DNS - IP주소
- 왜 80번이라는 같은 포트를 쓰는가?
  - 서버는 24시간 켜져있어야함 , IP주소가 각자 다 다르니 포트라도 같은 것을 쓰자



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

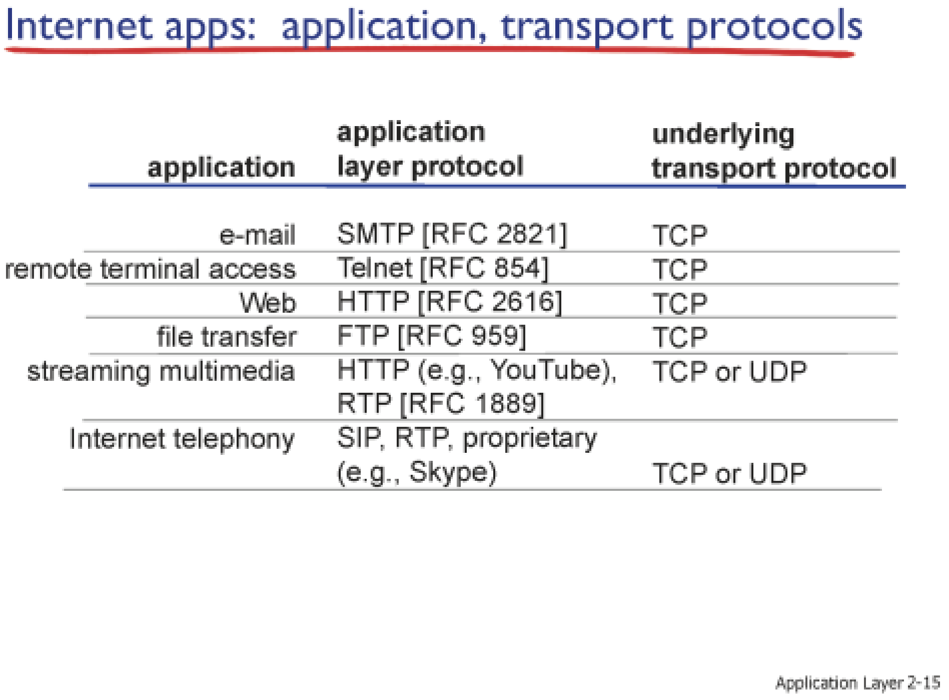
- some apps (e.g., multimedia) require minimum amount of throughput to be "effective"
- other apps ("elastic apps") make use of whatever throughput they get

security

- encryption, data integrity, …

Application Layer 2-12

- data integrity : data가 유실안되게하는 것만 현재 완벽히 기능

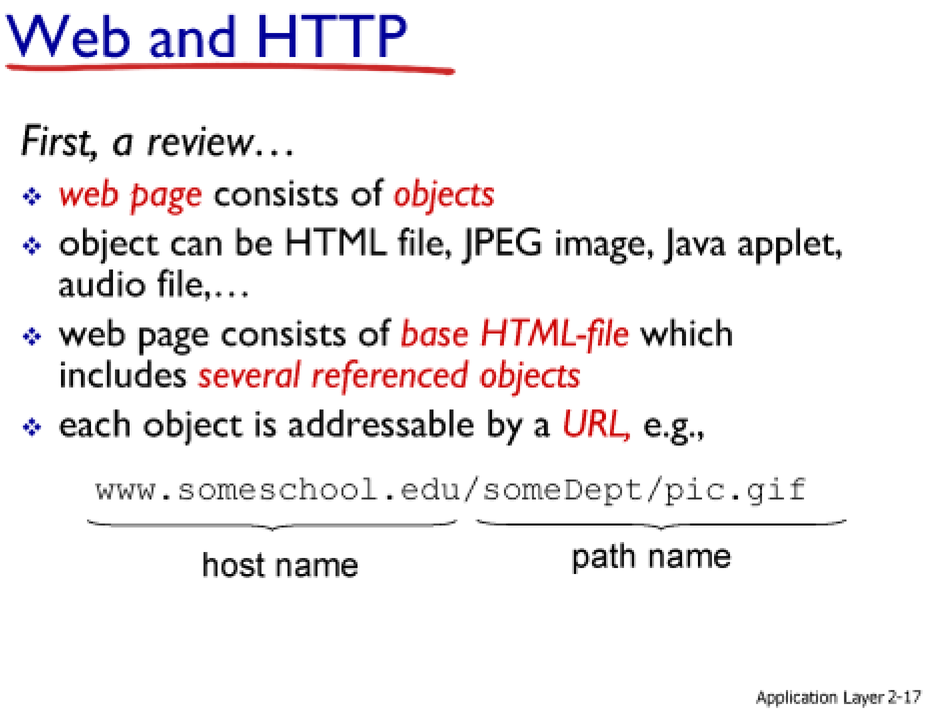


Internet apps: application, transport protocols

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

Application Layer 2-15

- 가장 유명한 것 : 웹을 동작시키는 HTTP



Web and HTTP

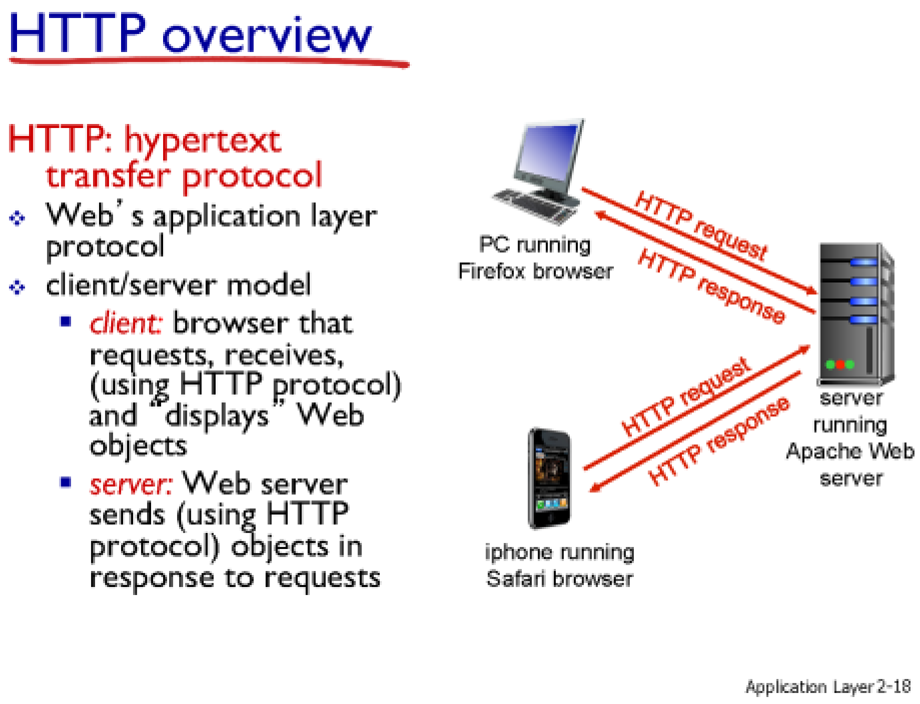
First, a review…

- *web page* consists of *objects*
- object can be HTML file, JPEG image, Java applet, audio file,…
- web page consists of *base HTML-file* which includes *several referenced objects*
- each object is addressable by a *URL,* e.g.,

$$\underbrace{\texttt{www.someschool.edu}}_{\text{host name}}\underbrace{\texttt{/someDept/pic.gif}}_{\text{path name}}$$

Application Layer 2-17

- HTTP : hypertext transfer protocol
  - 단순히 텍스트만 전달하는 프로토콜



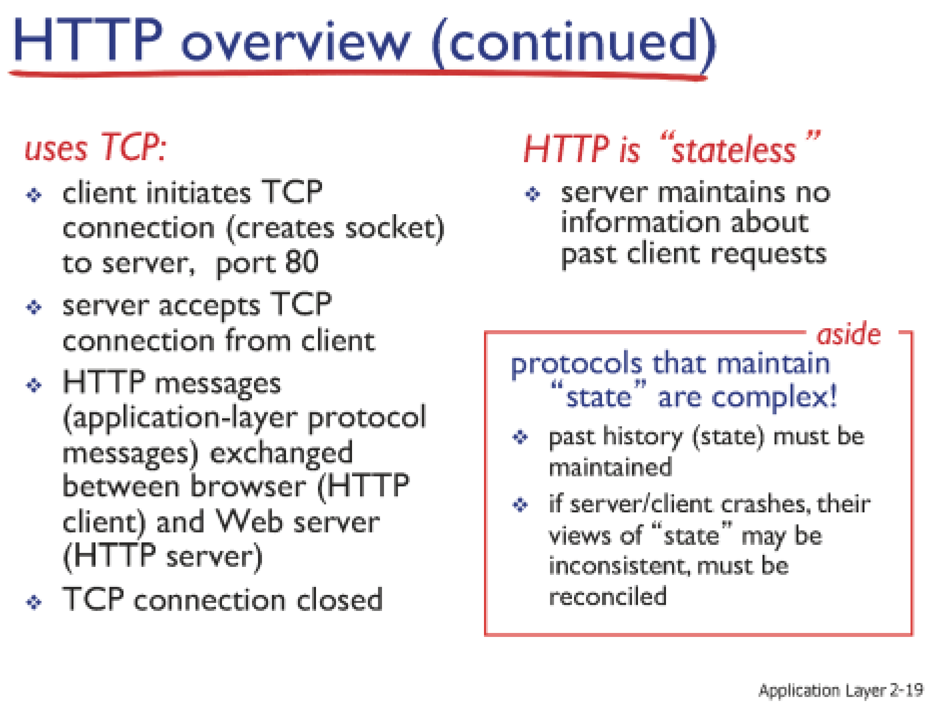
HTTP overview

HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server model
  - *client:* browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - *server:* Web server sends (using HTTP protocol) objects in response to requests

PC running Firefox browser
HTTP request
HTTP response
server running Apache Web server
HTTP request
HTTP response
iphone running Safari browser

Application Layer 2-18



HTTP overview (continued)

uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

HTTP is "stateless"

- server maintains no information about past client requests

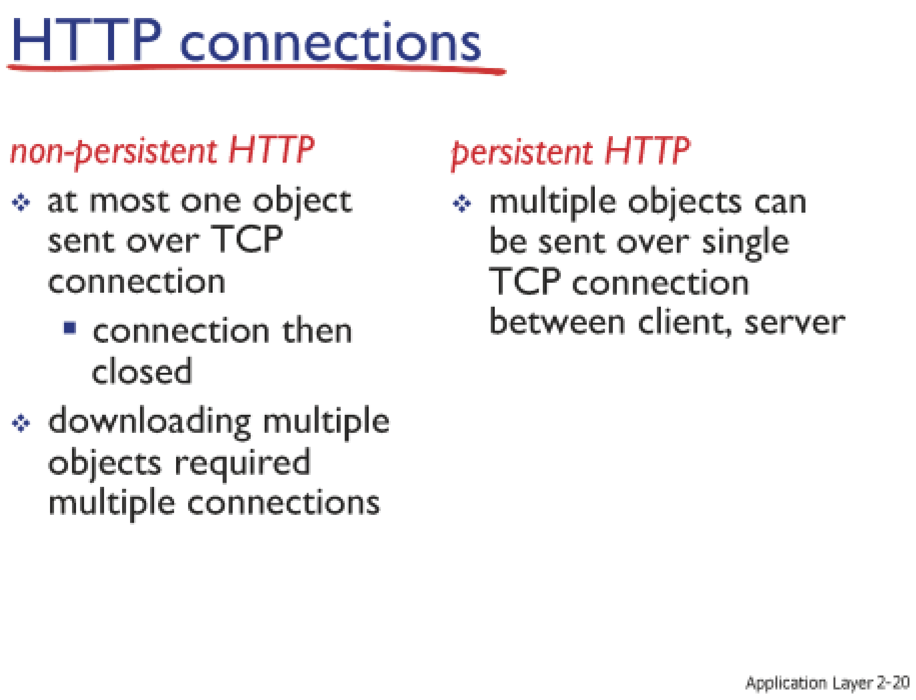
aside

protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, must be reconciled

Application Layer 2-19

- TCP 사용
- stateless : 상대방에 대한 상태를 기억하지 않는다



HTTP connections

non-persistent HTTP

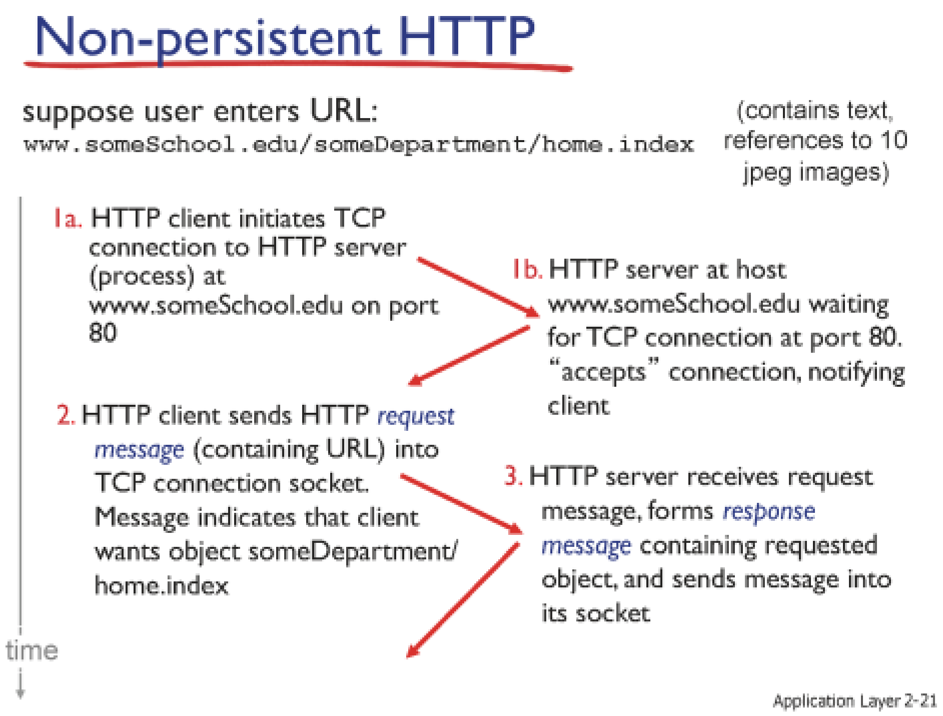
- at most one object sent over TCP connection
  - connection then closed
- downloading multiple objects required multiple connections

persistent HTTP

- multiple objects can be sent over single TCP connection between client, server

Application Layer 2-20

- tcp connection을 계속 재사용하면서 사용할 것인가



Non-persistent HTTP

suppose user enters URL:
`www.someSchool.edu/someDepartment/home.index`

(contains text, references to 10 jpeg images)

1a. HTTP client initiates TCP connection to HTTP server (process) at www.someSchool.edu on port 80

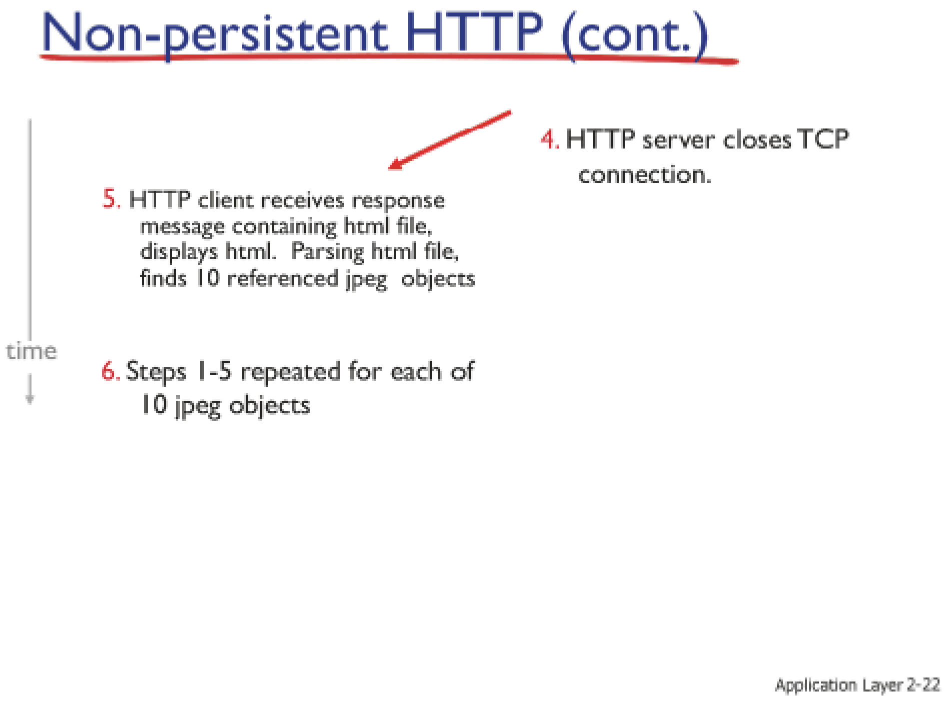
1b. HTTP server at host www.someSchool.edu waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* (containing URL) into TCP connection socket. Message indicates that client wants object someDepartment/ home.index

3. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

Application Layer 2-21

- tcp connection 을 바탕으로 해서 http request 이루어짐
- nonpersistent type : 연결을 끊음



Non-persistent HTTP (cont.)

4. HTTP server closes TCP connection.

5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

Application Layer 2-22

- 다시 끊고 연결하고 반복