인하공업전문대학
INHA TECHNICAL COLLEGE

TCP/IP
네트워크 프로그래밍
4주차

인하공업전문대학 컴퓨터 정보과

김한결 강사

목차

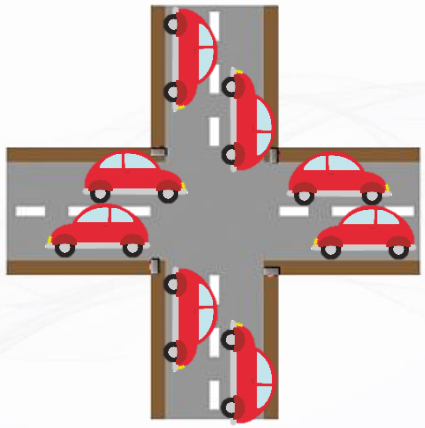
- 네트워크 기초 복습
  - ✓ 패킷이란?
  - ✓ 비트와 바이트란?
  - ✓ 문자코드
  - ✓ 랜과 왠의 차이
  - ✓ 가정에서의 네트워크 구성
  - ✓ 회사에서의 네트워크 구성

• 패킷이란?

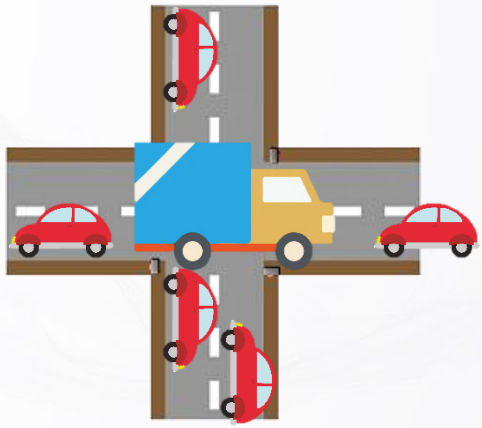
- 네트워크를 통해 전송되는 데이터의 작은 조각
- 큰 데이터가 있더라도 작게 나누어서 보내는 게 규칙

패킷 = 택배

흐름 원활



흐름 정체



대역폭 -> 일반적으로 네트워크에서 이용 가능한 최대 전송 속도로 정보를 전송할 수 있는 단위 시간당 전송량을 말함.

• 비트 와 바이트란?

- 비트 -> 정보를 나타내는 최소 단위
- 2진수 '0' 과 '1' 집합인 디지털 데이터

출처 : https://www.samsungsemiconstory.com/2440

• 문자코드

- ASCII 코드
- 네트워크 통신, 메일 메세지 에서 사용되는 기본 문자

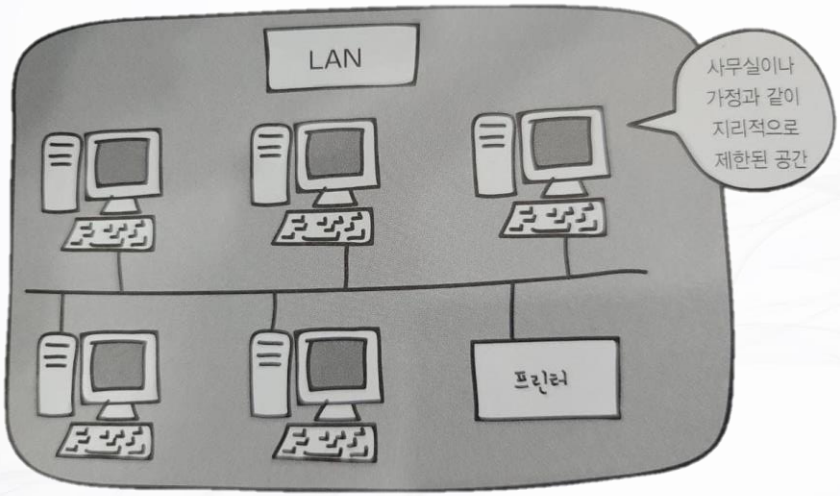
제어 문자 공백 문자 구두점 숫자 알파벳

10진	16진	문자	10진	16진	문자	10진	16진	문자	10진	16진	문자
0	0x00	NUL	32	0x20	SP	64	0x40	@	96	0x60	`
1	0x01	SOH	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	STX	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	ETX	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	EOT	36	0x24	$	68	0x44	D	100	0x64	d
5	0x05	ENQ	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	ACK	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	BS	40	0x28	(	72	0x48	H	104	0x68	h
9	0x09	HT	41	0x29	)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	SO	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	SI	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	DLE	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	DC1	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	DC2	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	DC3	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	DC4	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	NAK	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	SYN	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	ETB	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	CAN	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	EM	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	SUB	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	ESC	59	0x3B	;	91	0x5B	[	123	0x7B	{
28	0x1C	FS	60	0x3C	<	92	0x5C	\	124	0x7C	|
29	0x1D	GS	61	0x3D	=	93	0x5D	]	125	0x7D	}
30	0x1E	RS	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	US	63	0x3F	?	95	0x5F	_	127	0x7F	DEL

출처 :
hhttps://m.blog.naver.com/PostView.nhn?blogId=kyuhgmi&logNo=220073551241&proxyReferer=https:%2F%2Fwww.google.com%2F

• 랜(Lan)과 왠(Wan)의 차이

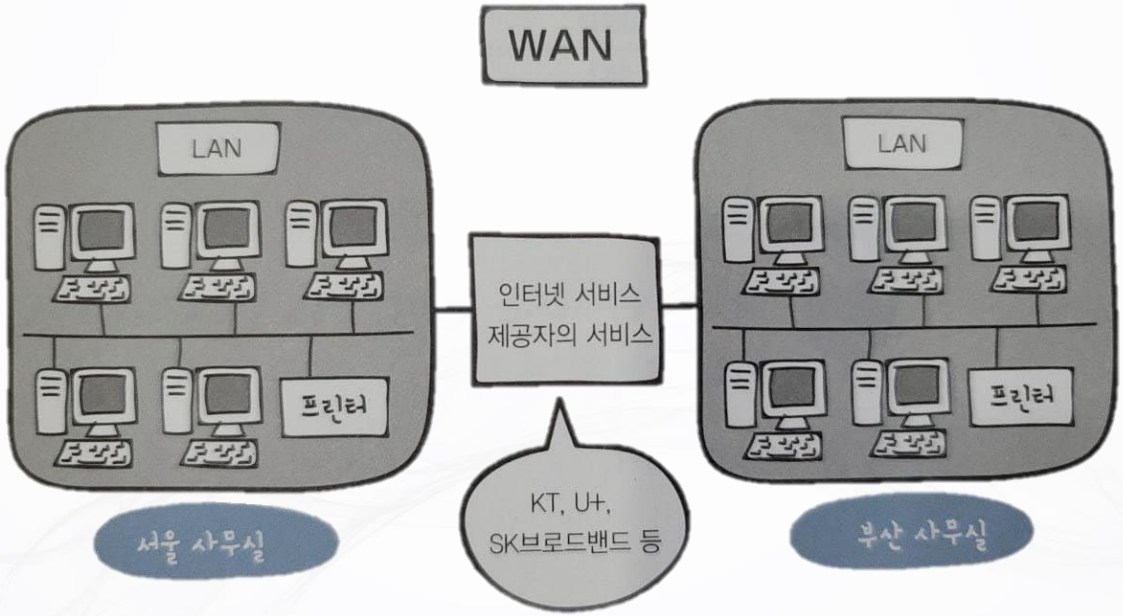
- Local Area Network (근거리 통신망)
- 사무실, 가정 같이 지리적으로 제한된 공간



출처 : 미즈구치 카츠야 지음, 모두의 네트워크

• 랜(Lan)과 왠(Wan)의 차이

- Wide Area Network(광역 통신망)
- ISP, Internet Service Provider 인터넷 서비스 제공자

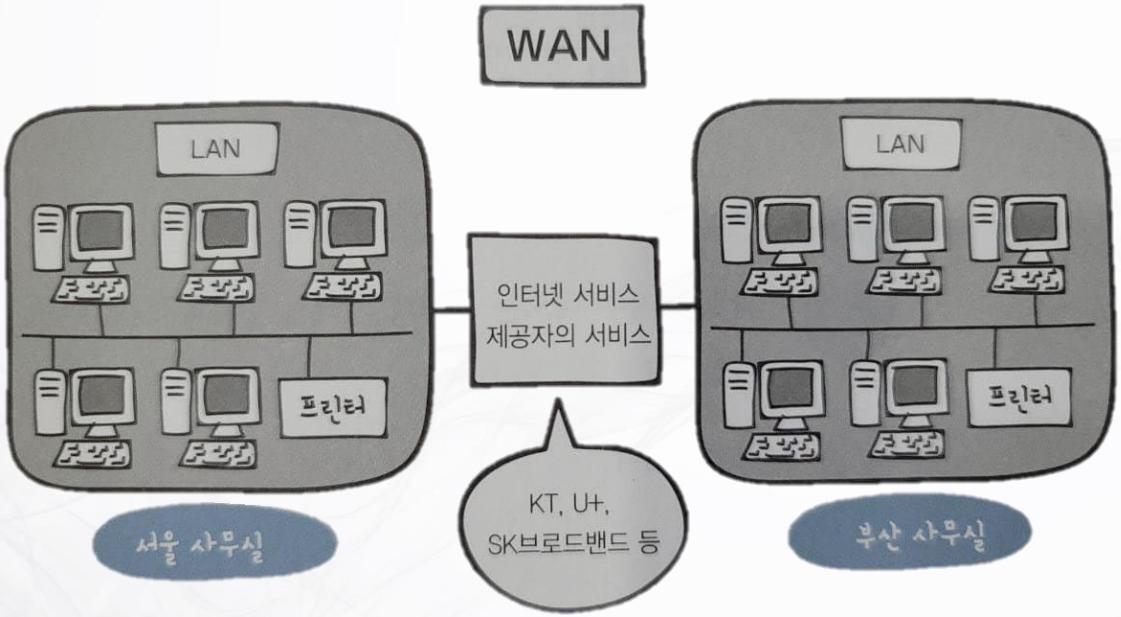


출처 : 미즈구치 카츠야 지음, 모두의 네트워크

• 랜(Lan)과 왠(Wan)의 차이

- Wide Area Network(광역 통신망)
- ISP, Internet Service Provider
  인터넷 서비스 제공자

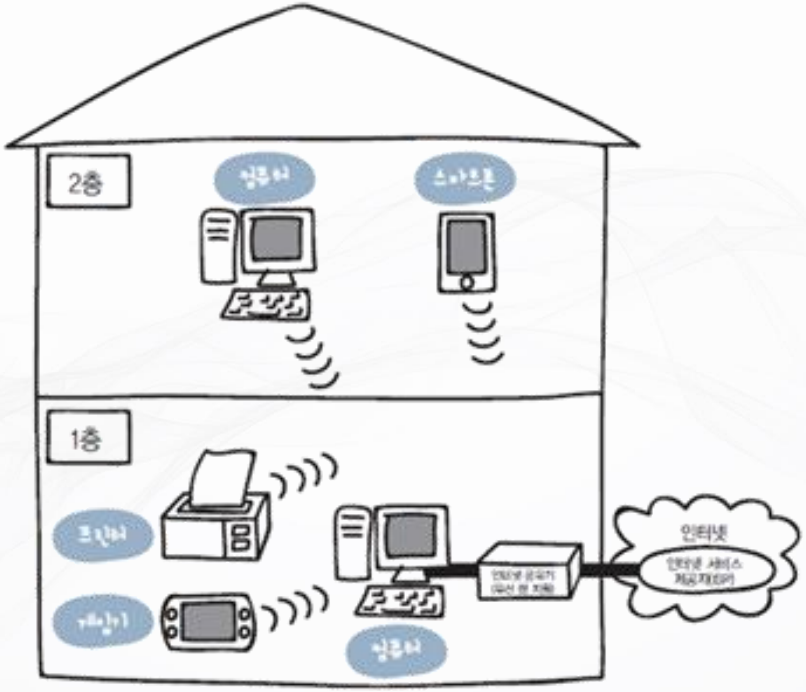
	LAN	WAN
범위	좁다(건물이나 특정지역)	넓다(랜과 랜을 연결)
속도	빠르다	느리다
오류	적다	많다



출처 : 미즈구치 카츠야 지음, 모두의 네트워크

• 가정에서의 네트워크 구성

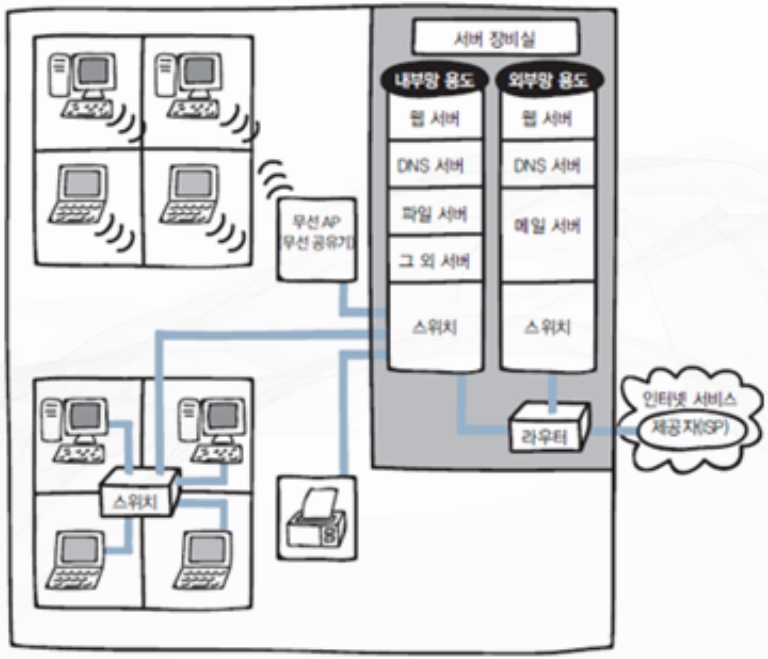
- 인터넷 서비스 제공자, 인터넷 회선을 결정 하고 계약
- 네트워크 장비 -> 유무선 공유기 사용
- 접속 방식 -> 유선 랜 방식, 무선 랜 방식



출처 : https://gourmet-eundong.tistory.com/16

• 회사에서의 네트워크 구성

- DMZ, Demilitarized Zone -> 외부에 공개하기 위한 네트워크
- 외부에 공개하는 서버, 웹서버, DNS 서버, 메일서버 등
- 회사의 서버는 온프레미스나 클라우드로 운영

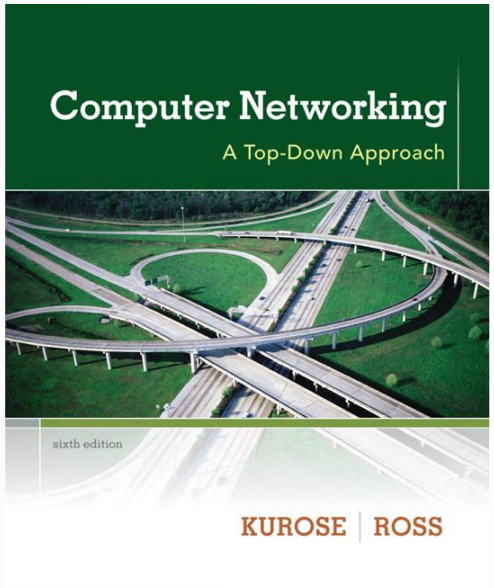


출처 : https://gourmet-eundong.tistory.com/16

• 연습문제

1. 네트워크에서 전송되는 작은 데이터 조각을 (**패킷**) 이라고 한다
2. 컴퓨터는 (**0**) 과 (**1**)만 이해한다
3. 정보를 표시하는 최소의 단위를 (**bit**)라고 한다.
4. 특정 건물이나 지역을 범위로 하고 속도가 빠르며 오류 발생 확률이 낮은 네트워크를 (**LAN**) 라고 한다.
5. 전기 통신 사업자가 제공하는 서비스를 사용하여 구축된 속도가 느리고 오류가 발생하기 쉬운 네트워크를 (**WAN**)이라고 한다.
6. 인터넷에 연결하려면 우선 (**ISP**) 와 인터넷 회선을 결정하고 계약한다.
7. 외부에 공개하기 위한 네트워크를 (**DMZ**)라고 한다.
8. 기업의 서버는 (**온프레미스**)나 클라우드 중 하나로 운영되고 있다.

참고서적



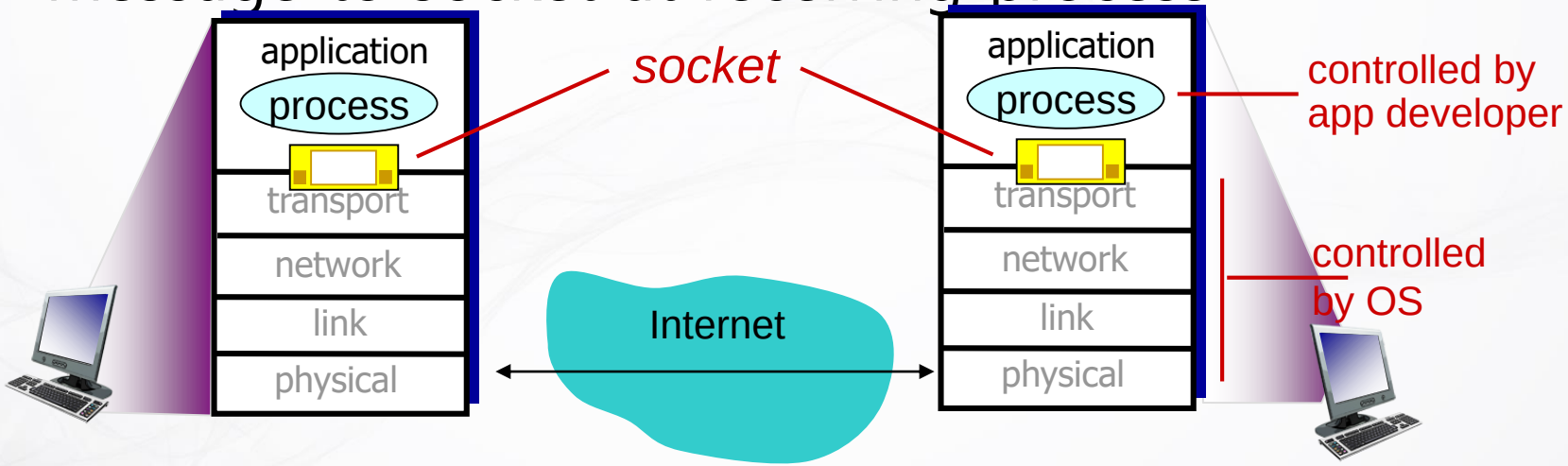
Computer Networking: A Top Down Approach
6th edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012

목차

- sockets
- Addresing processes
- protocol
- Transport protocol
- TCP UDP
- http
- Non-persistent http

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 "effect ive"
- other apps ("elastic apps") make use of whatever thro ughput 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 receive r
- *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, orconnection 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

- *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.,

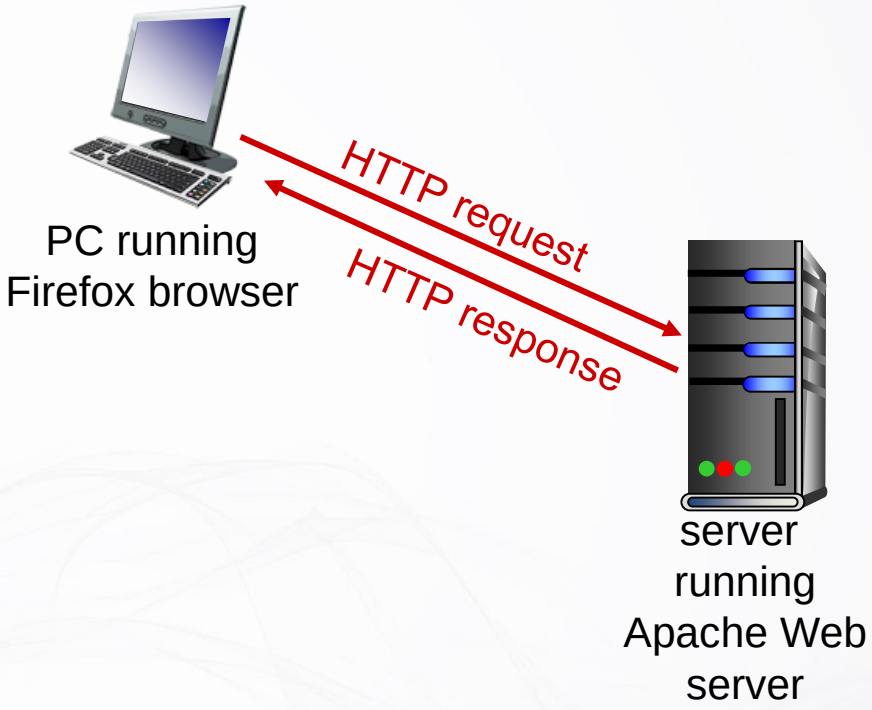
`www.inhatc.ac.kr/someDept/pic.gif`

host name | path name

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



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

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

Non-persistent HTTP

1a. HTTP client initiates TCP connection to HTTP server (process) at www.inhatc.ac.kr on port 80

1b. HTTP server at host www.inhatc.ac.kr 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

time

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

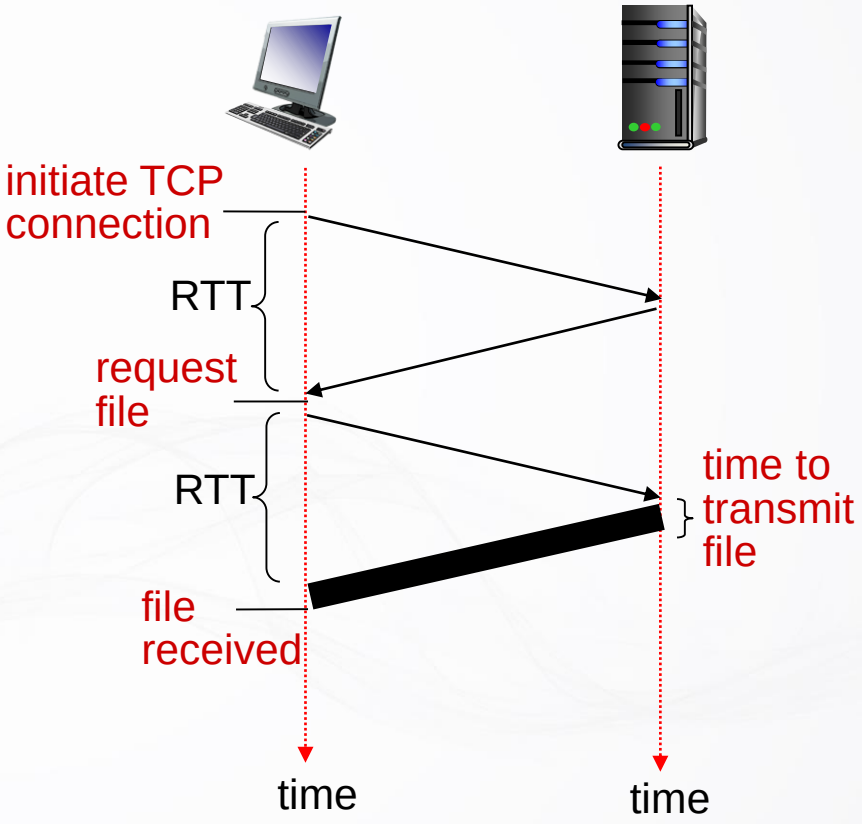
Non-persistent HTTP : response time

RTT (definition): time for a small packet to travel from client to server and back

HTTP response time:

- one RTT to initiate TCP connection
- one RTT for HTTP request and first few bytes of HTTP response to return
- file transmission time
- non-persistent HTTP response time =

  2RTT+ file transmission time



4주차 수업이 끝났습니다

고생하셨습니다.