Computer Network

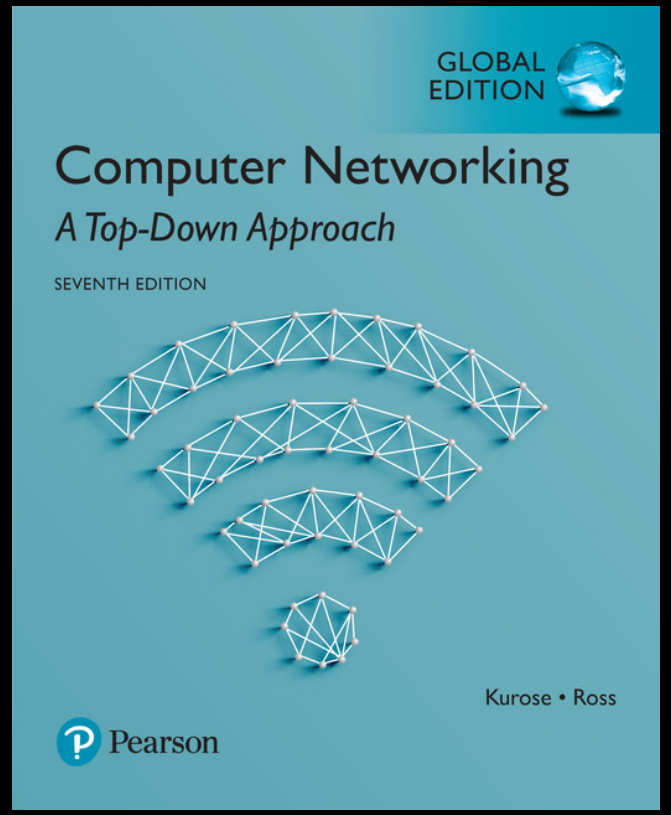
Application Layer

School of Electric and Computer Engineering

Pusan National University, KOREA

Younghwan Yoo

Textbook



Computer Networking

A Top–Down Approach

7th edition

Jim Kurose, Keith Ross

Pearson

April 2016

Contents

Computer Network introduction

Contents

Computer Network introduction

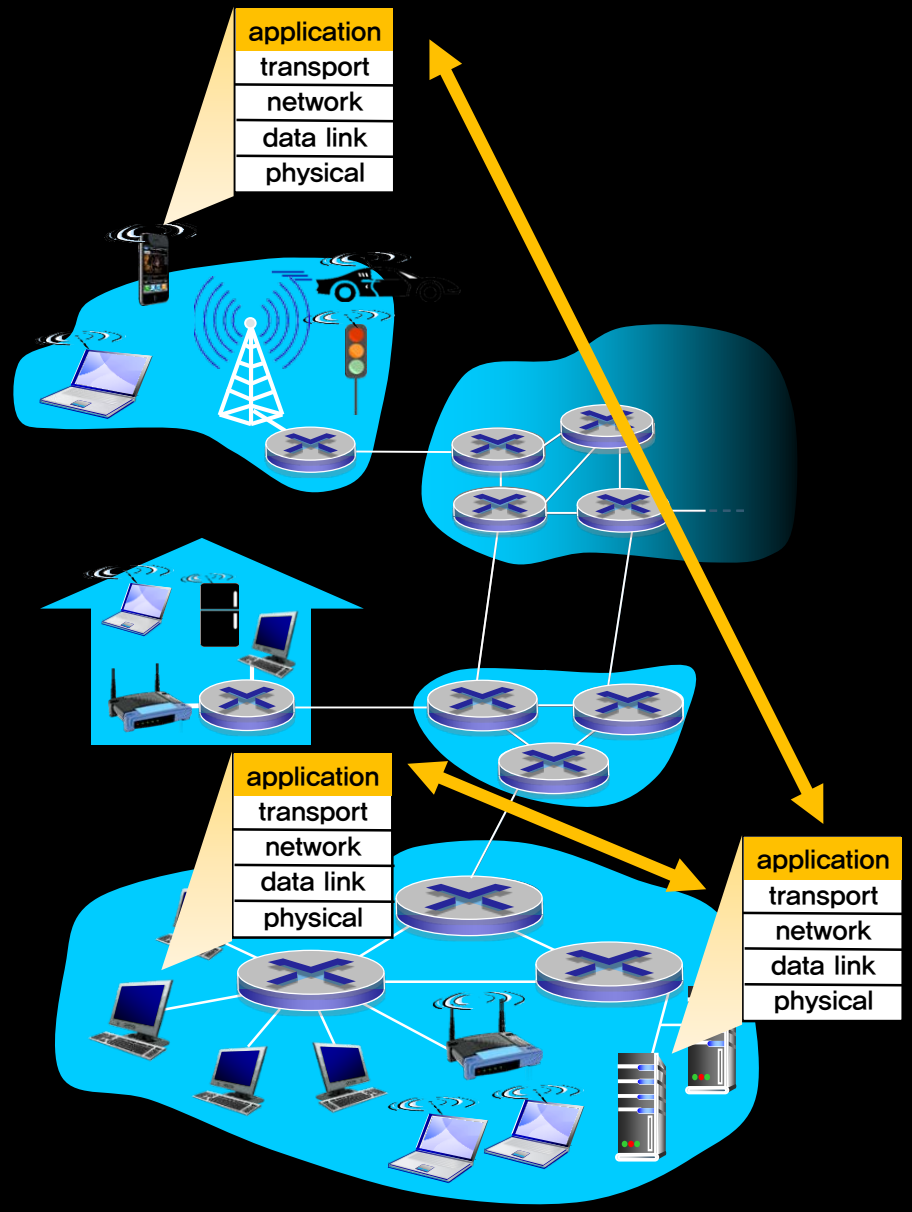
Video Clip: HTTP & HTML

- The Internet: HTTP & HTML (https://www.youtube.com/watch?v=kBXQZMmiA4s&index=5&list=PLzdnOPI1iJNfMRZm5DDxco3UdsFegvuB7)

01. Principles of Application

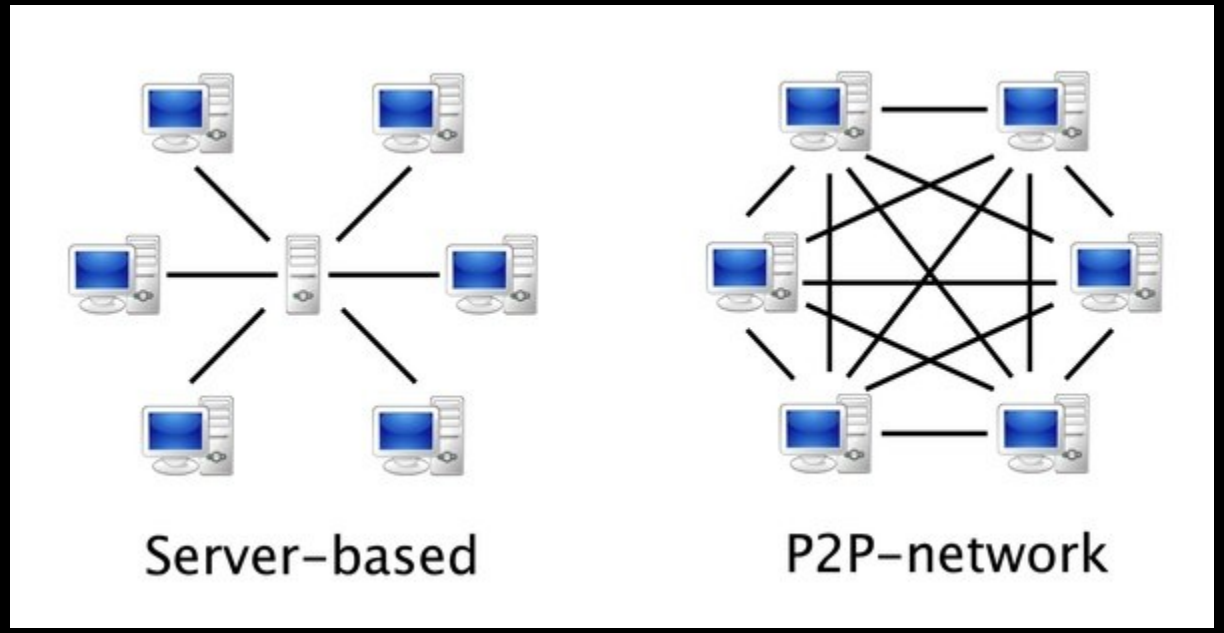
Network Applications

- Types
  - email
  - web (server software, browser)
  - P2P file sharing
  - SNS (Social Network Service)
  - messenger program
  - online-game
  - streaming stored video (YouTube, Netflix)
- Run on (different) *end systems*
  - network-core devices do not run user applications
- Communicate over network



Application Architectures (1/3)

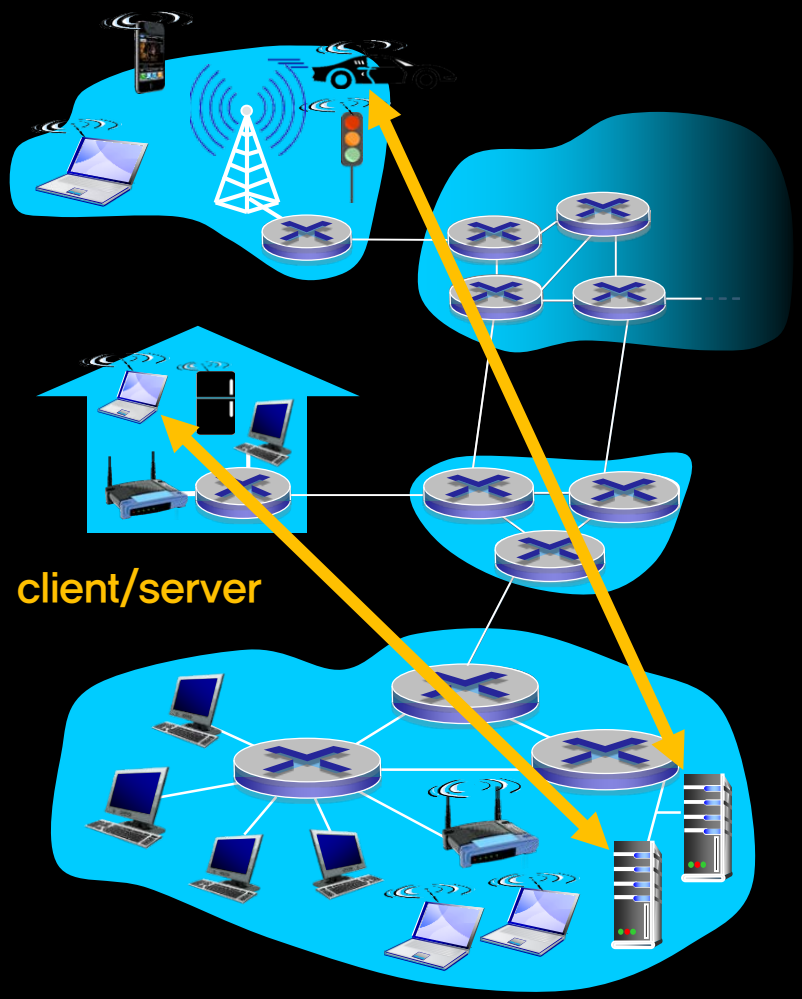
- Two kinds of application structures:
  - Client–server model
  - Peer–to–peer (P2P) model



출처 – https://www.quora.com/Whats-difference-between-p2p-and-cdn/

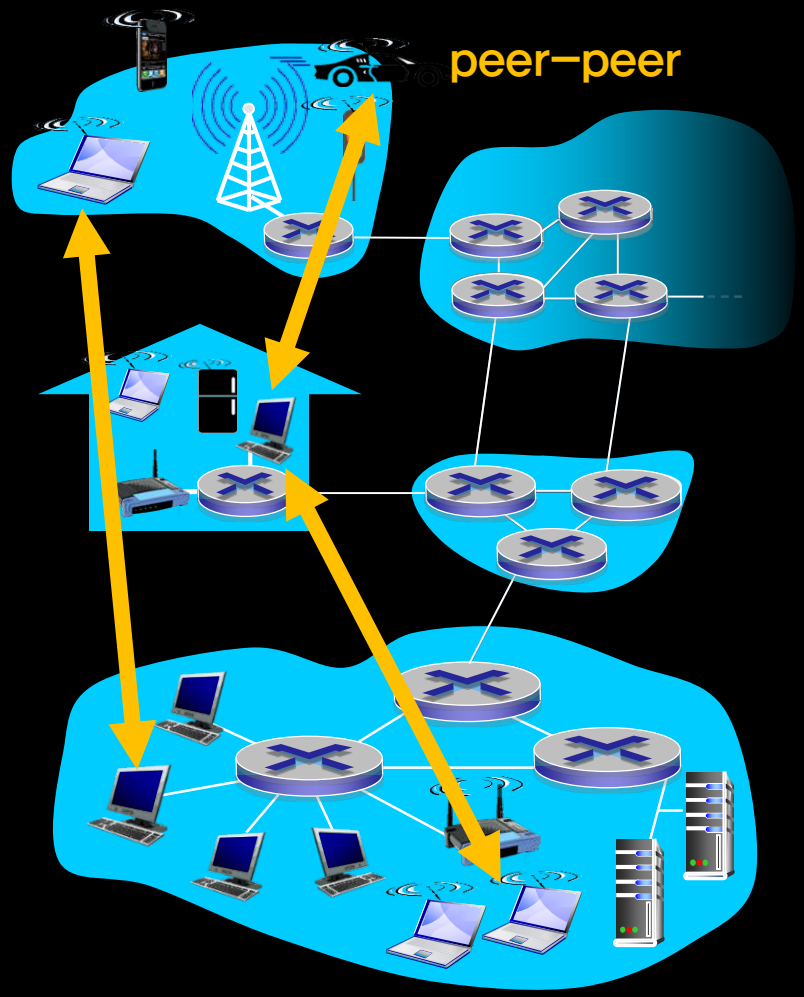
Application Architectures (2/3)

- Client–server model
  - server:
    - always–on
    - permanent IP address
    - data centers for scaling
  - client:
    - communicate with server
    - may be intermittently connected
    - may have dynamic IP addresses
    - do not communicate directly with each other



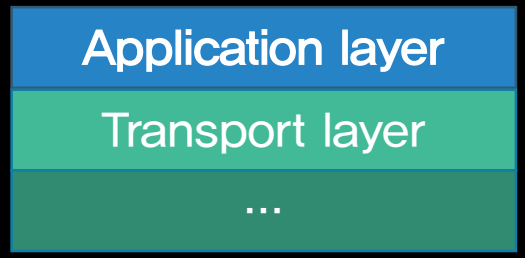
Application Architectures (3/3)

- Peer–to–peer (P2P) model
  - no always–on server
  - arbitrary end systems directly communicate
  - self scalability – new peers bring new service capacity, as well as new service demands
  - peers are intermittently connected and change IP addresses
    - complex management

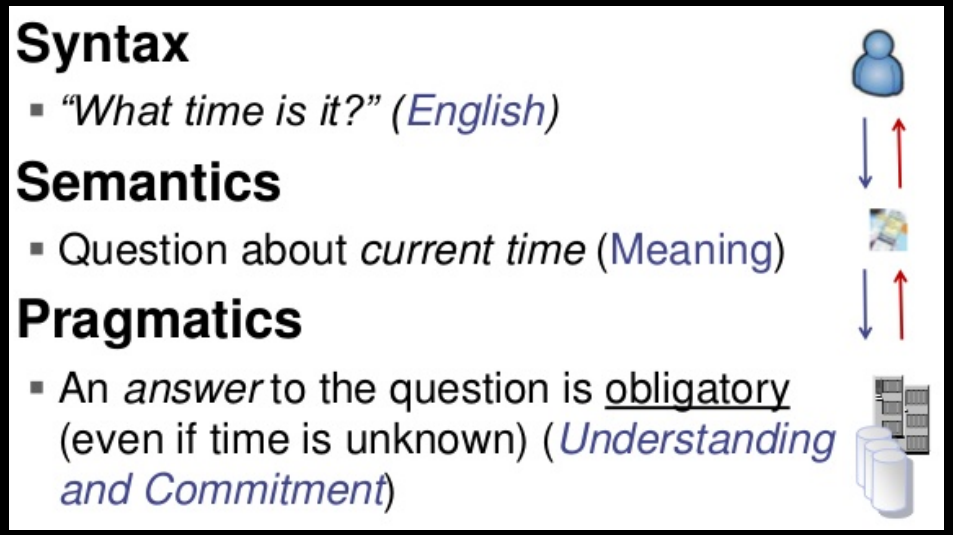


Application Layer Protocol

- Top layer of Internet protocol stack



- Human communication protocol

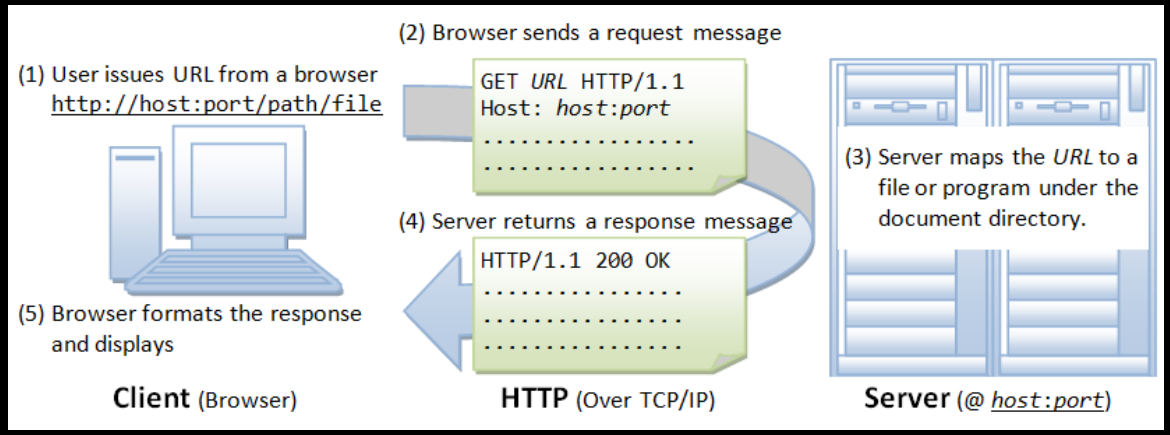


출처 – https://www.slideshare.net/swadpasc/pragmatic-web-paschke/

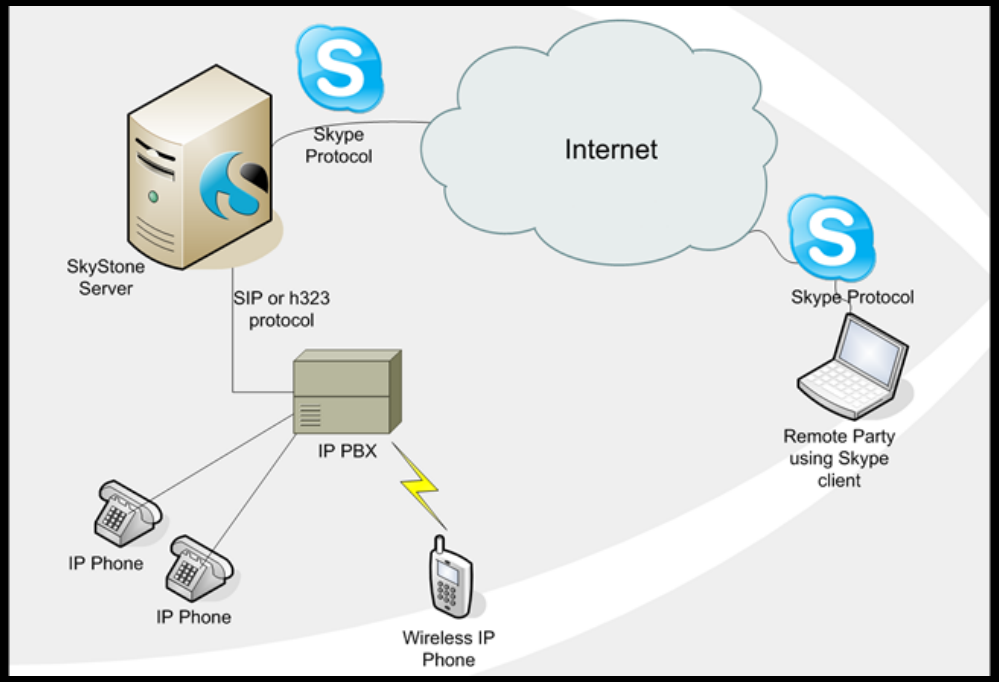
- Network application protocol
  - types of messages exchanged
    - e.g., request, response
  - message syntax
    - what fields in messages & how fields are delineated
  - message semantics
    - meaning of information in fields
  - message pragmatics
    - when and how processes send & respond to messages

Application Protocol

- Open protocols
  - for interoperability
  - e.g., HTTP, SMTP
- Proprietary protocol
  - e.g., Skype



출처 – https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html/



출처 – http://in.myinfoline.com/forum/reply/1826/

Requirements of Network Applications

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
text messaging	no loss	elastic	yes and no

- Who does meet these requirements?

Transport layer protocols!!!

Transport Protocol Services

TCP service:

- Error control: 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?

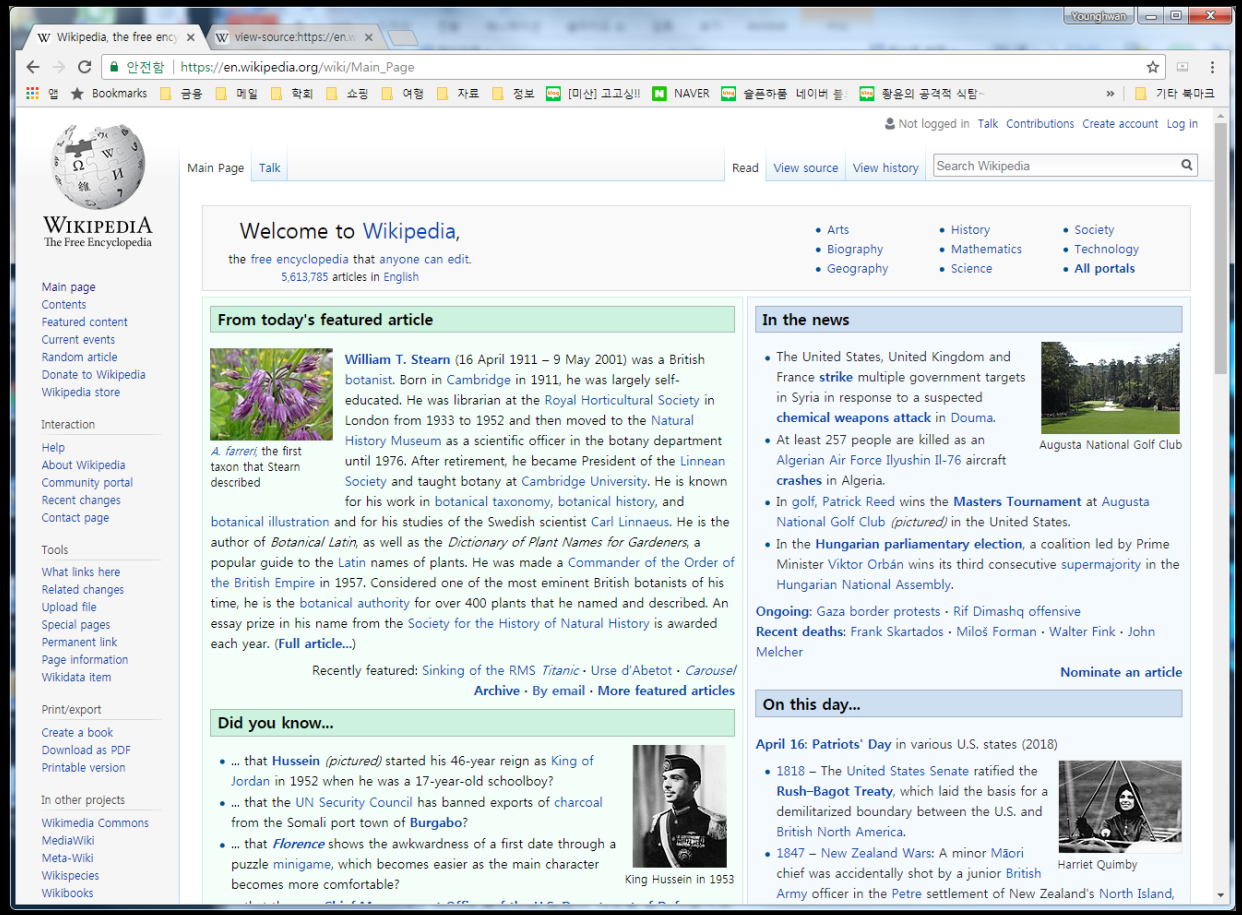
Application & Transport Protocol Pairs

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

02. Web and HTTP

World Wide Web (WWW) (1/2)

- Web page consists of objects
- Object can be HTML file, JPEG image, Java applet, audio file,…



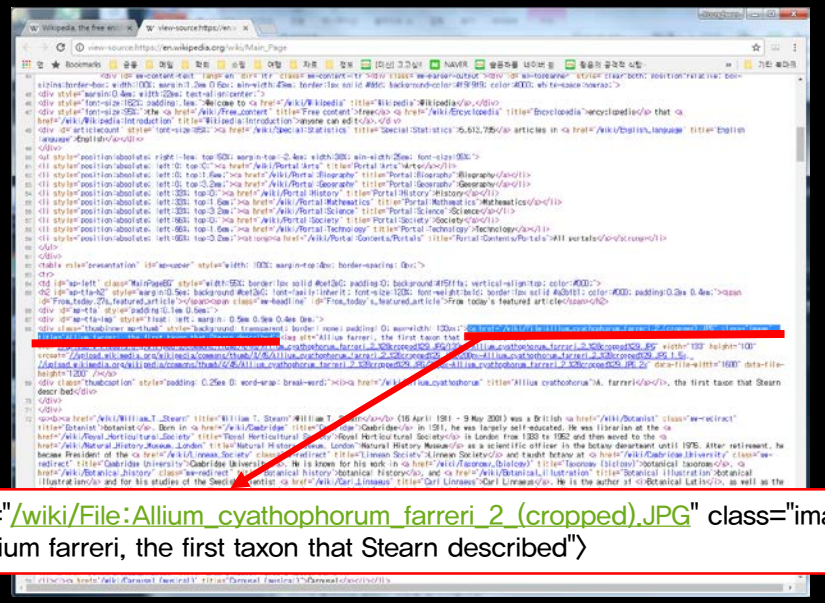
World Wide Web (WWW) (2/2)

- Web page is described by HTML-file which includes several referenced objects
- Each object is addressable by a URL, e.g.,

www.someschool.edu/someDept/pic.gif

host name | path name

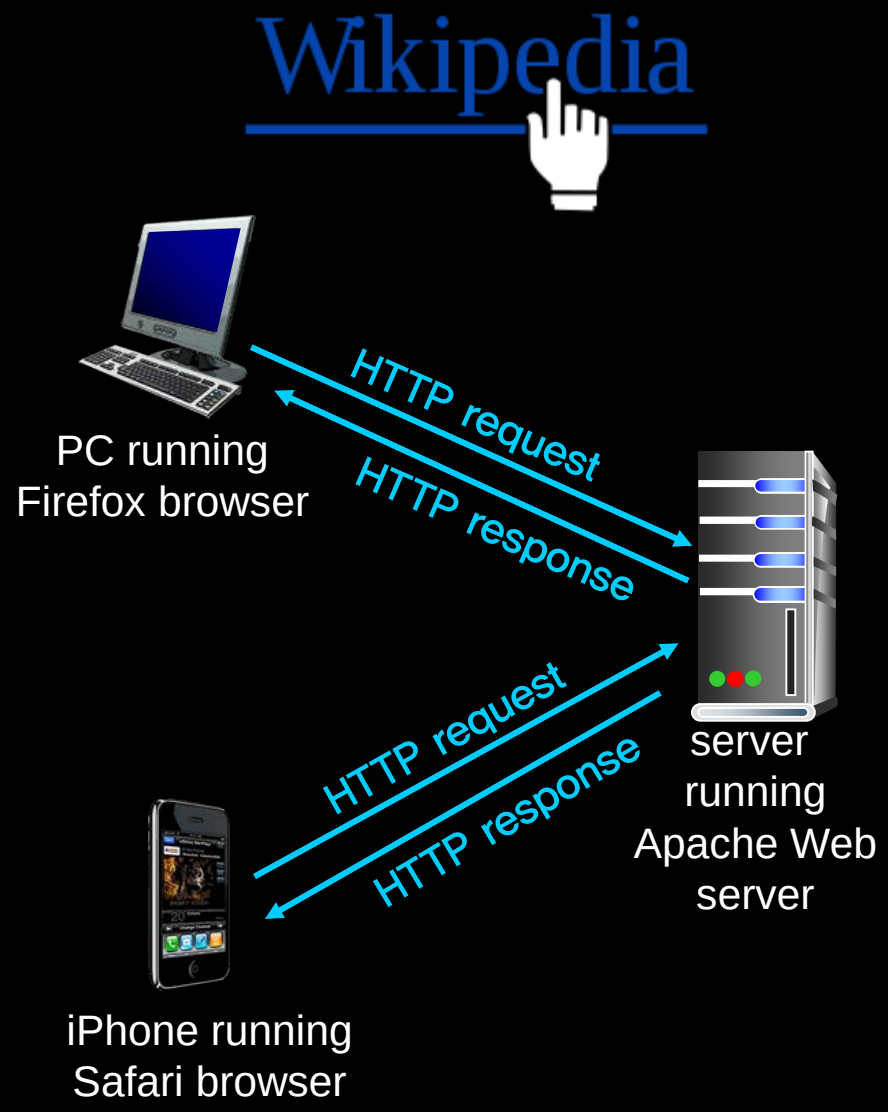
마우스오른쪽버튼
-> 페이지소스보기



HTTP Overview (1/3)

HTTP (HyperText Transfer Protocol)

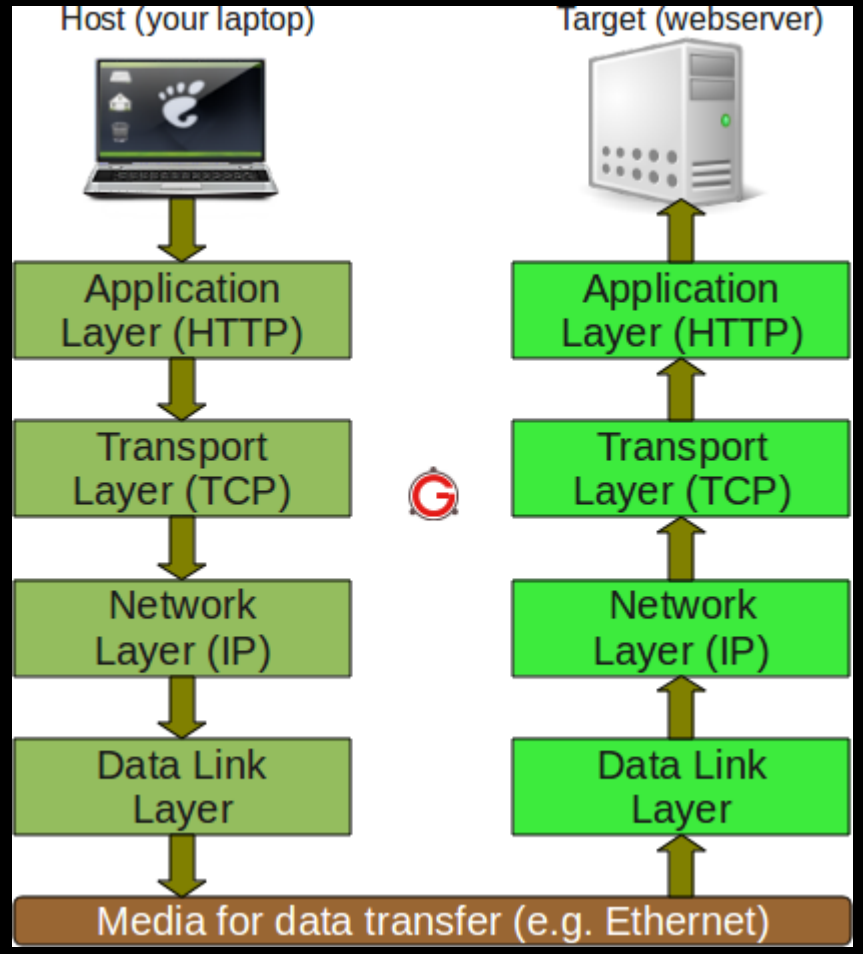
- Web's application layer protocol
- Hyperlink: a reference to data the reader can directly follow by clicking
- Client/server model
  - client: browser that requests, receives, and "displays" Web objects
  - server: web server sends objects in response to requests



HTTP Overview (2/3)

Based on TCP

1. Client initiates TCP connection (creates socket) to server
2. Server accepts TCP connection from client
3. HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
4. TCP connection closed

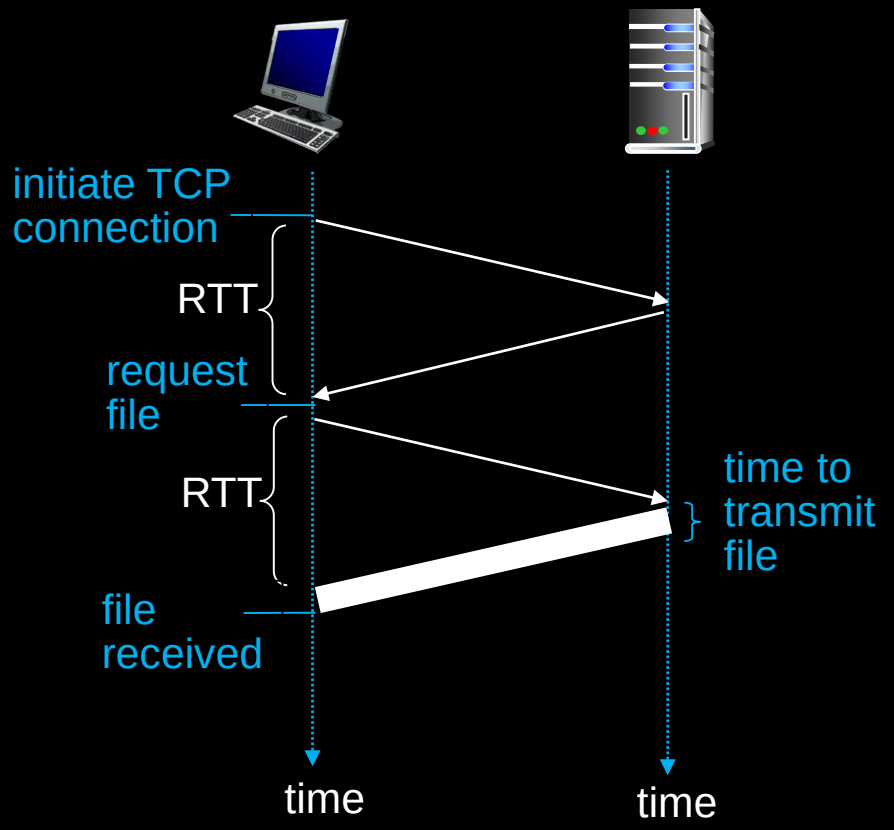


출처 – http://tamil-it-guru.blogspot.com/2017/02/tcpip-protocol-fundamentals-explained.html/

HTTP Overview (3/3)

- RTT: Round-Trip-Time
- 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
  - Total response time =

    2RTT + file transmission time



HTTP Version

HTTP/1 (1996)

- Non-persistent HTTP: one object per TCP connection
- Long latency

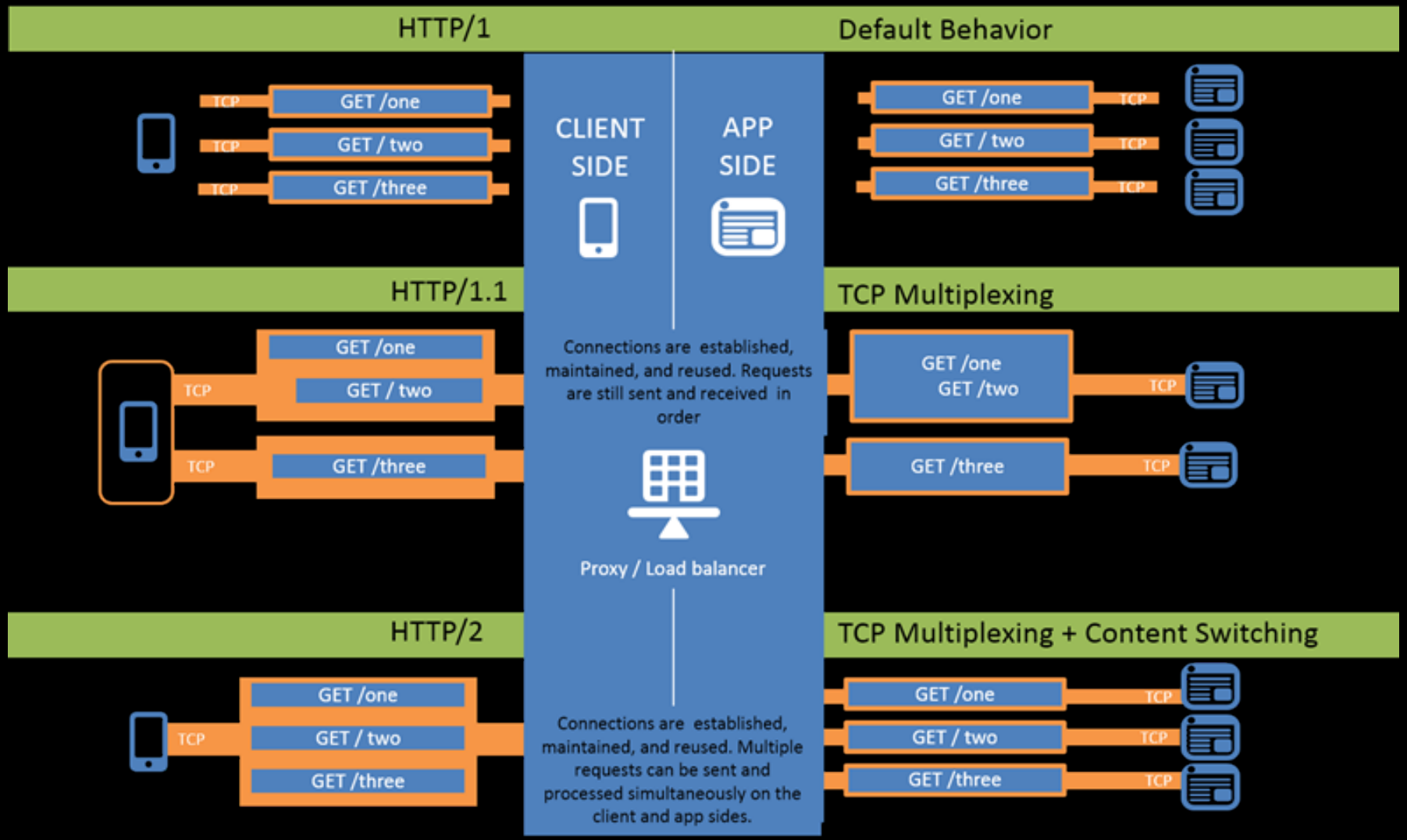
HTTP/1.1 (1999)

- Persistent HTTP: multiple objects over one TCP connection
- Latency decrease
- Synchronous order of response/request pairs over one TCP connection

HTTP/2 (2015)

- Persistent HTTP
- Asynchronous (parallel) multiple request/response pairs over one connection
- Compatible with HTTP/1.1

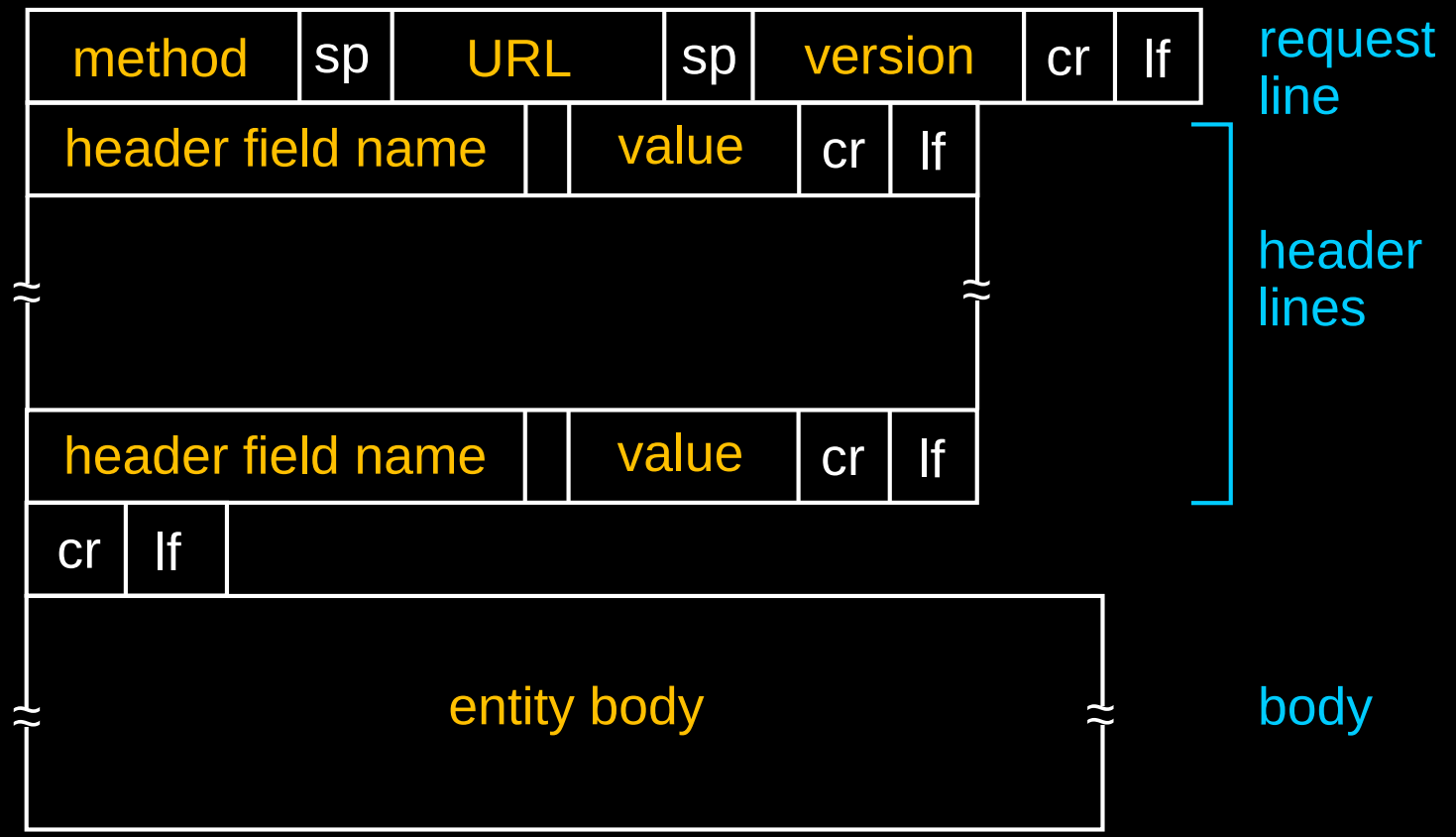
HTTP Version



출처 – https://devcentral.f5.com/articles/multiplexing-tcp-vs-http2

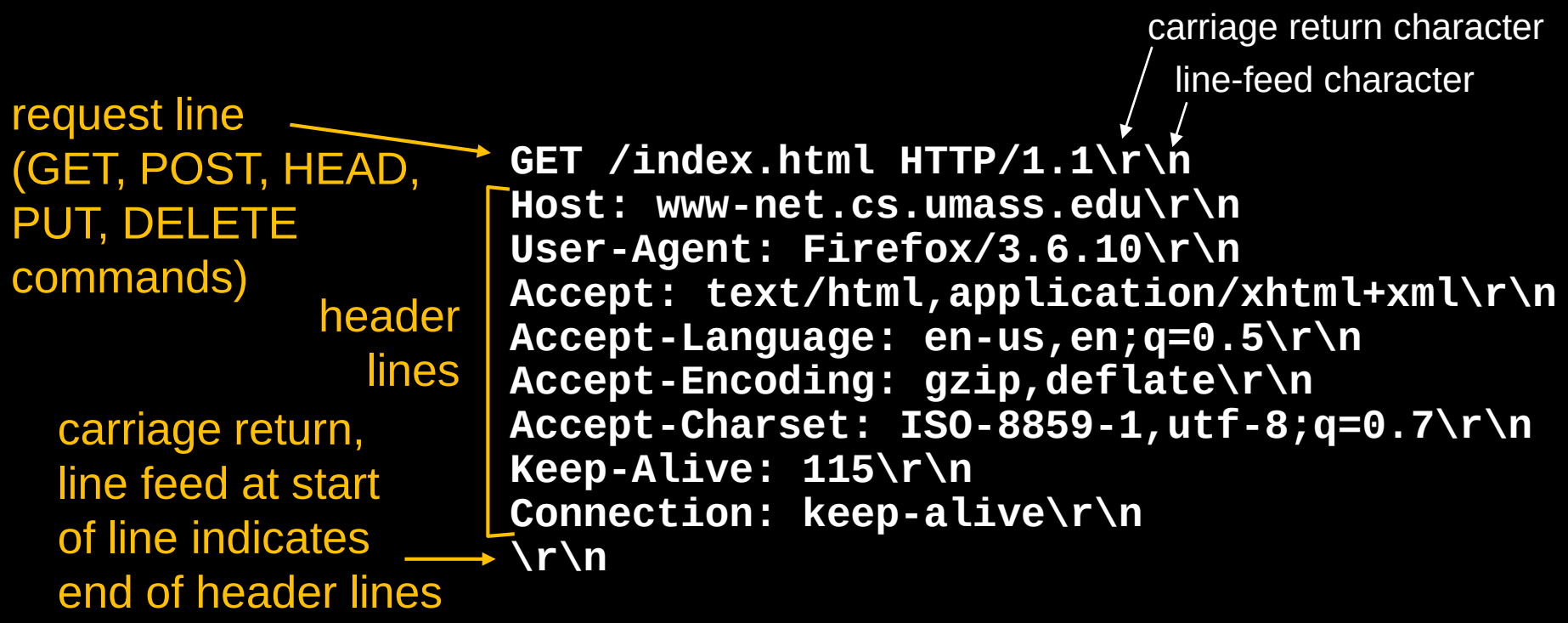
HTTP Message

- Two types of messages: *request*, *response*
- Message format

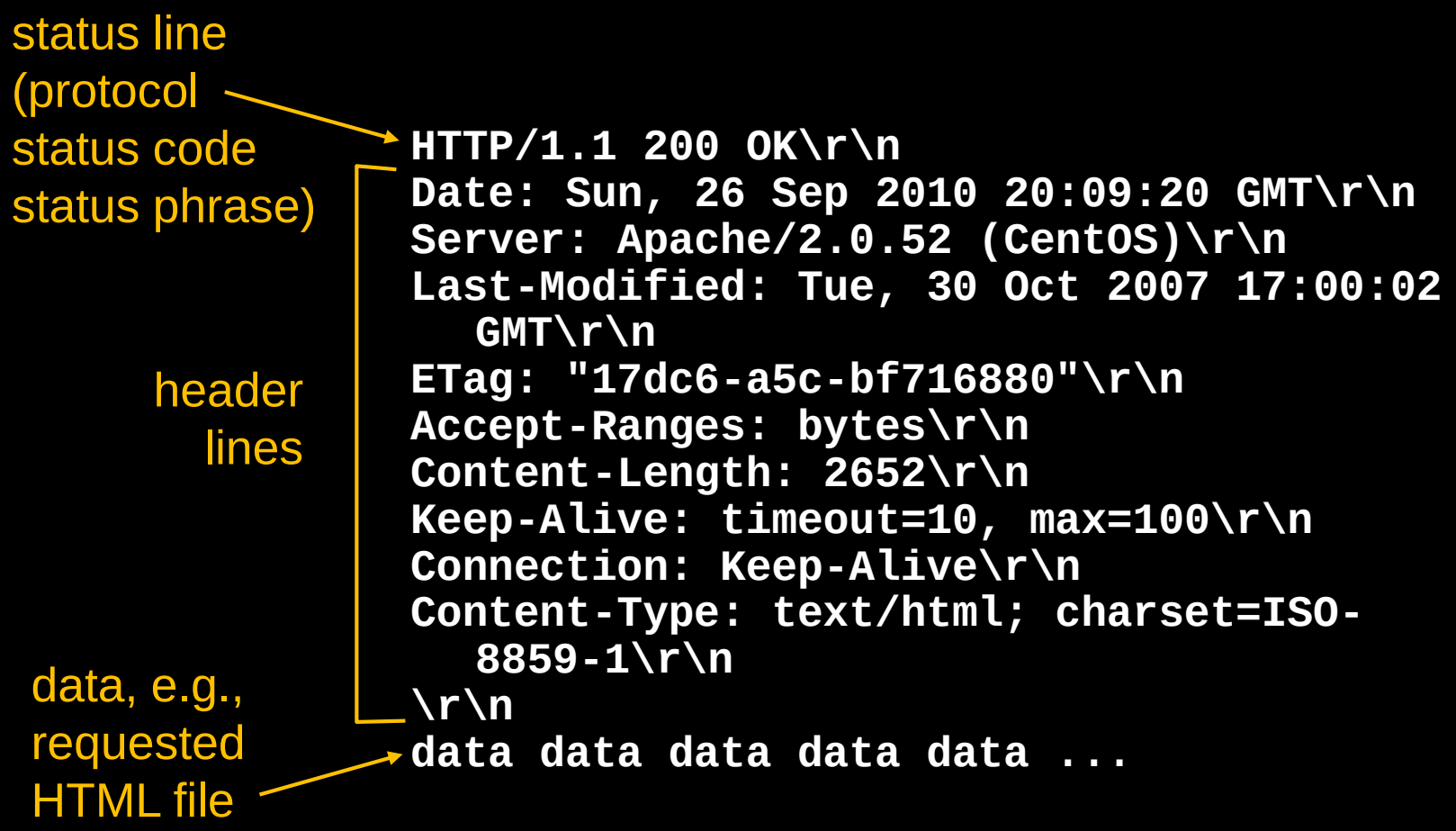


HTTP Request Message

- ASCII (human–readable format)



HTTP Response Message



- Response codes
  - 200 OK
  - 301 Moved Permanently
  - 400 Bad Request
  - 404 Not Found
  - 505 HTTP Version Not Supported

REST

- REpresentational State Transfer
  - HTTP should be “stateless”, proposed by PhD. Roy Fielding in 2000
  - Used in designing the HTTP/1.1 and Uniform Resource Identifiers (URI) standard
  - RESTful service: service that conforms to the REST architectural style
- Architectural constraints
  - client–server architecture: separation of the user interface concerns from the data storage concerns
  - statelessness: no client context being stored on the server between requests
  - cacheability: server responses cacheable on client and intermediaries
  - layered system: unable to tell whether a client is directly connected to the end server or to an intermediary along the way
  - code on demand (optional): able to transfer executable code such as Java applets and Java Script
  - uniform interface: simplification and decoupling of the architecture

03. Cookies & Web Caching

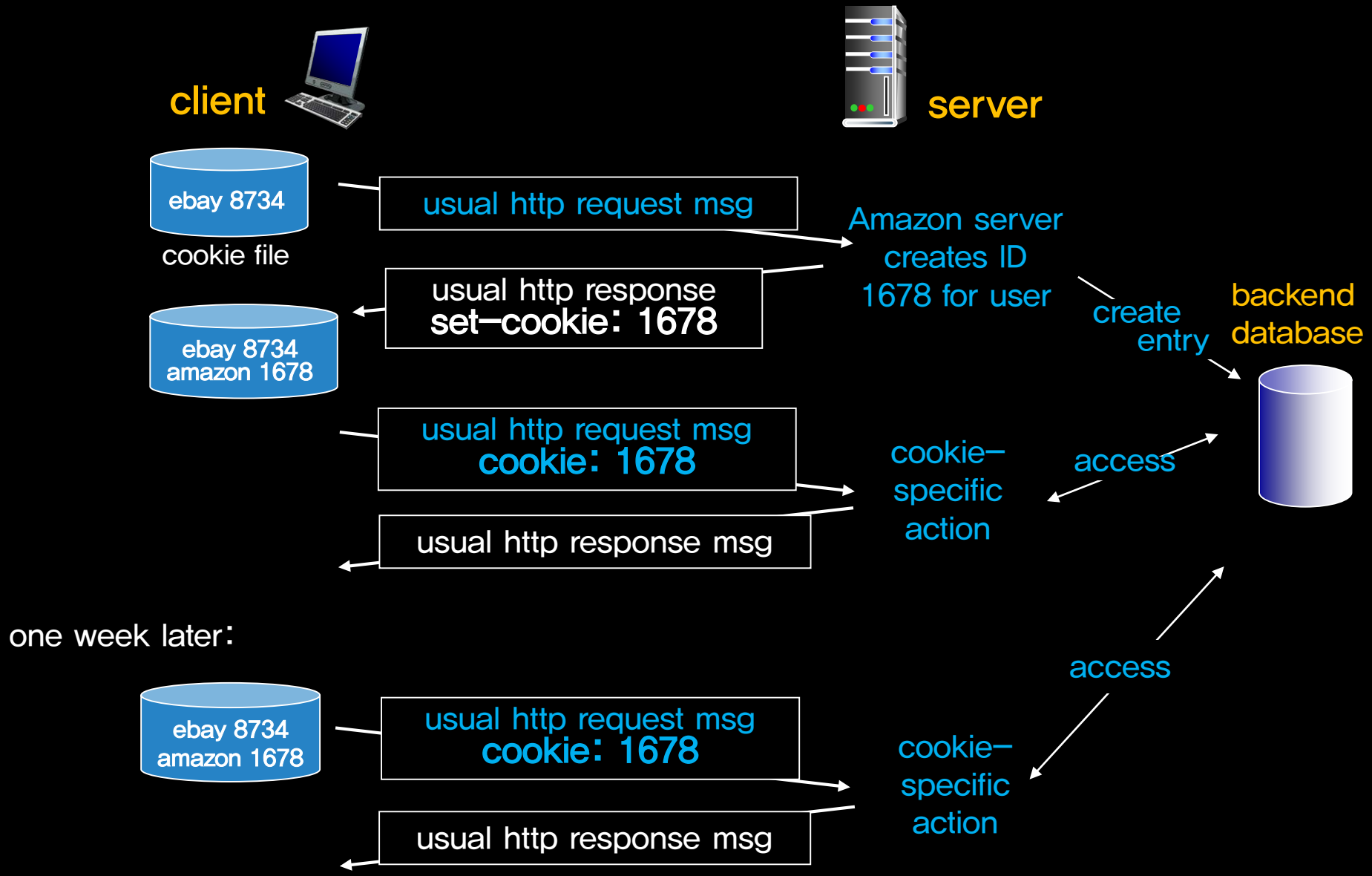
User Experience Enhancement on Online

출처 – http://osxdaily.com/2016/05/08/automatic-login-mac/

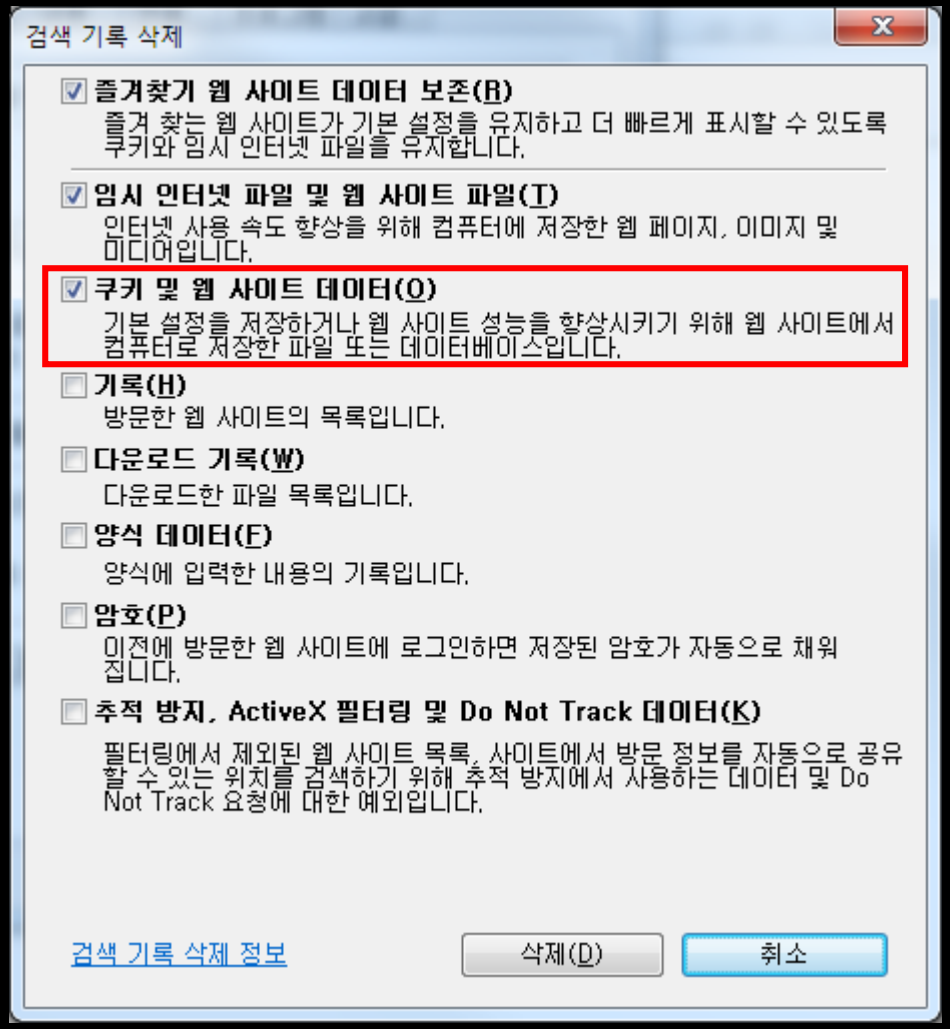
출처 – https://willscullypower.wordpress.com/tag/online-shopping/

- Automatic user authorization and item keeping in cart on online shopping
- HTTP is stateless, then how the server can know the user?
- Cookies!!!

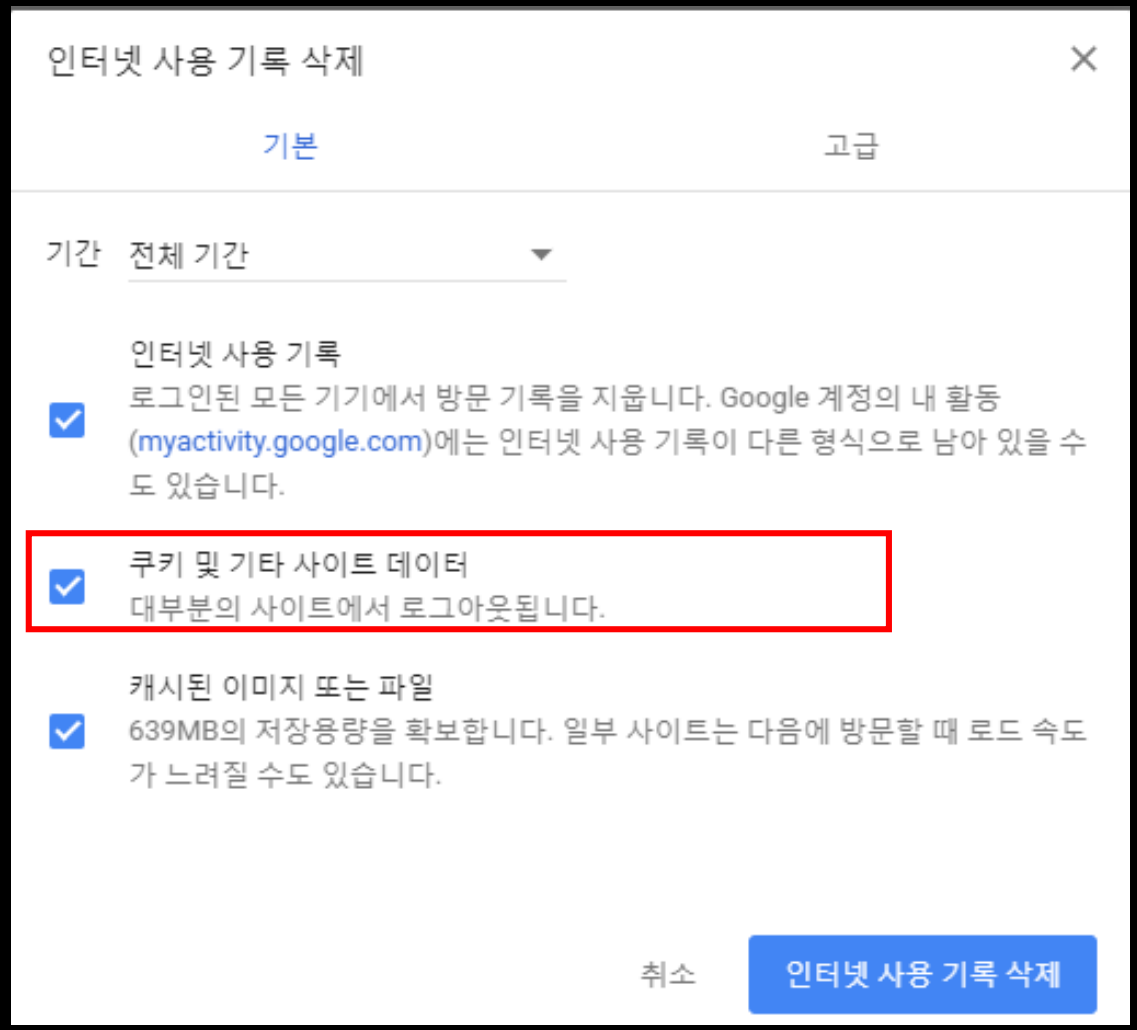
Cookies: Keeping "State"



Cookies on Browsers



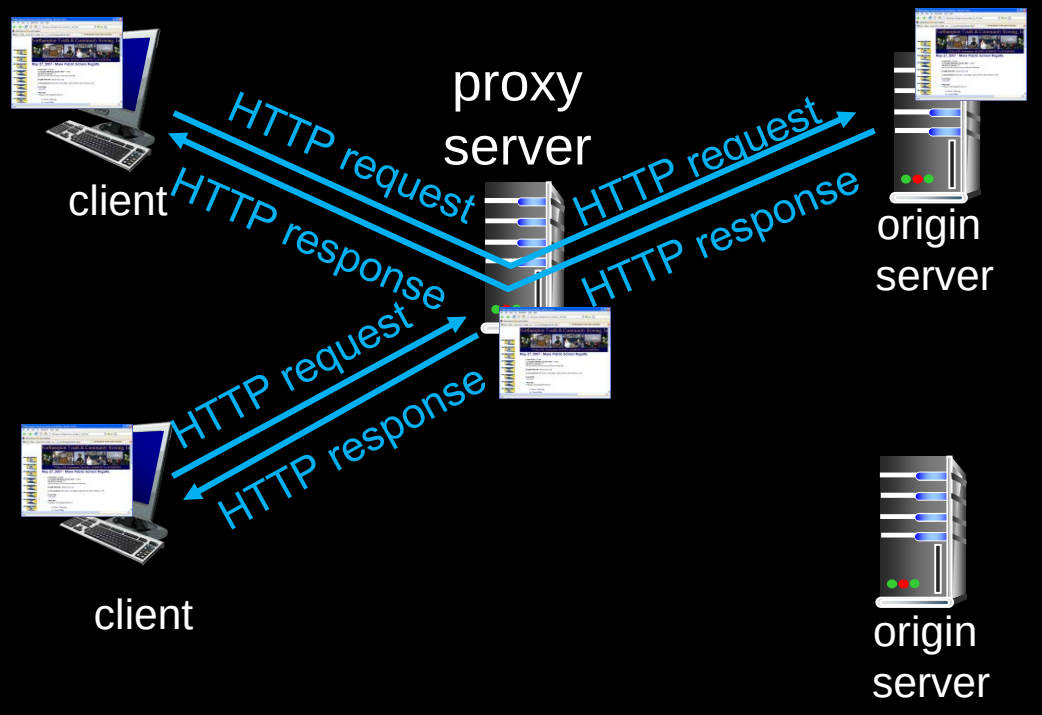
Explorer



Chrome

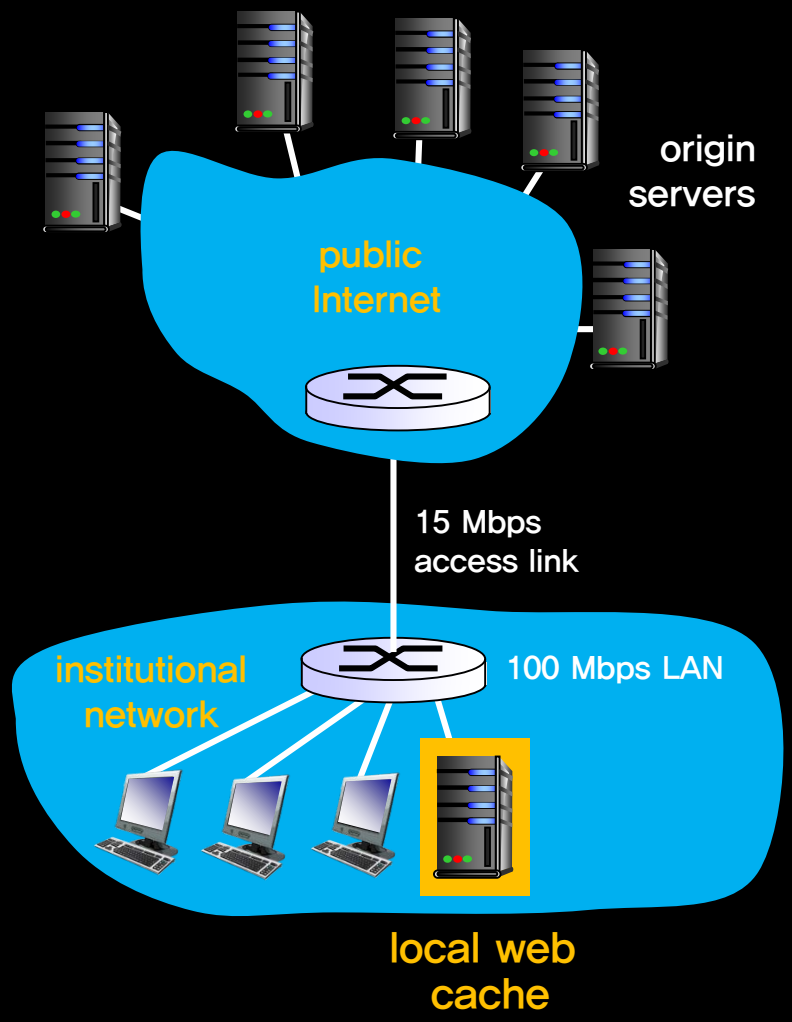
Web Caching (Proxy Server)

- Browser set to access Web via cache
- All HTTP requests sent to cache
  - if the requested object in cache: cache returns the object to the client
  - otherwise: cache requests the object from the origin server, then returns it to the client
- Cache acts as both client and server



Web Caching

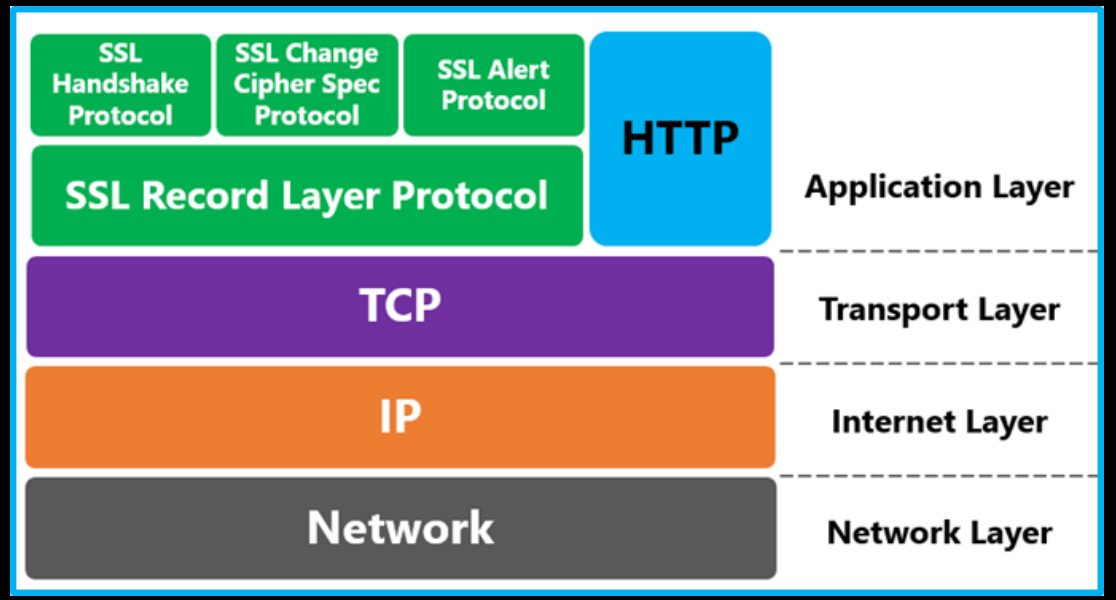
- Effect of Web caching
  - for client, response time reduction
  - for server (content provider), more user support
  - for local ISP, efficient usage of access link bandwidth by reducing traffic to external server



04. SSL/TLS

Securing TCP

- Transport layer protocols: TCP, UDP
  - no encryption
  - even passwords traversed Internet in cleartext
- SSL/TLS
  - provides encrypted TCP connection at the application layer
  - data integrity
  - end-point authentication



출처 – https://linuxacademy.com/community/posts/show/topic/14103-lecture-elb-amp-ec2-logging-puzzled-about-http-vs-tcp/

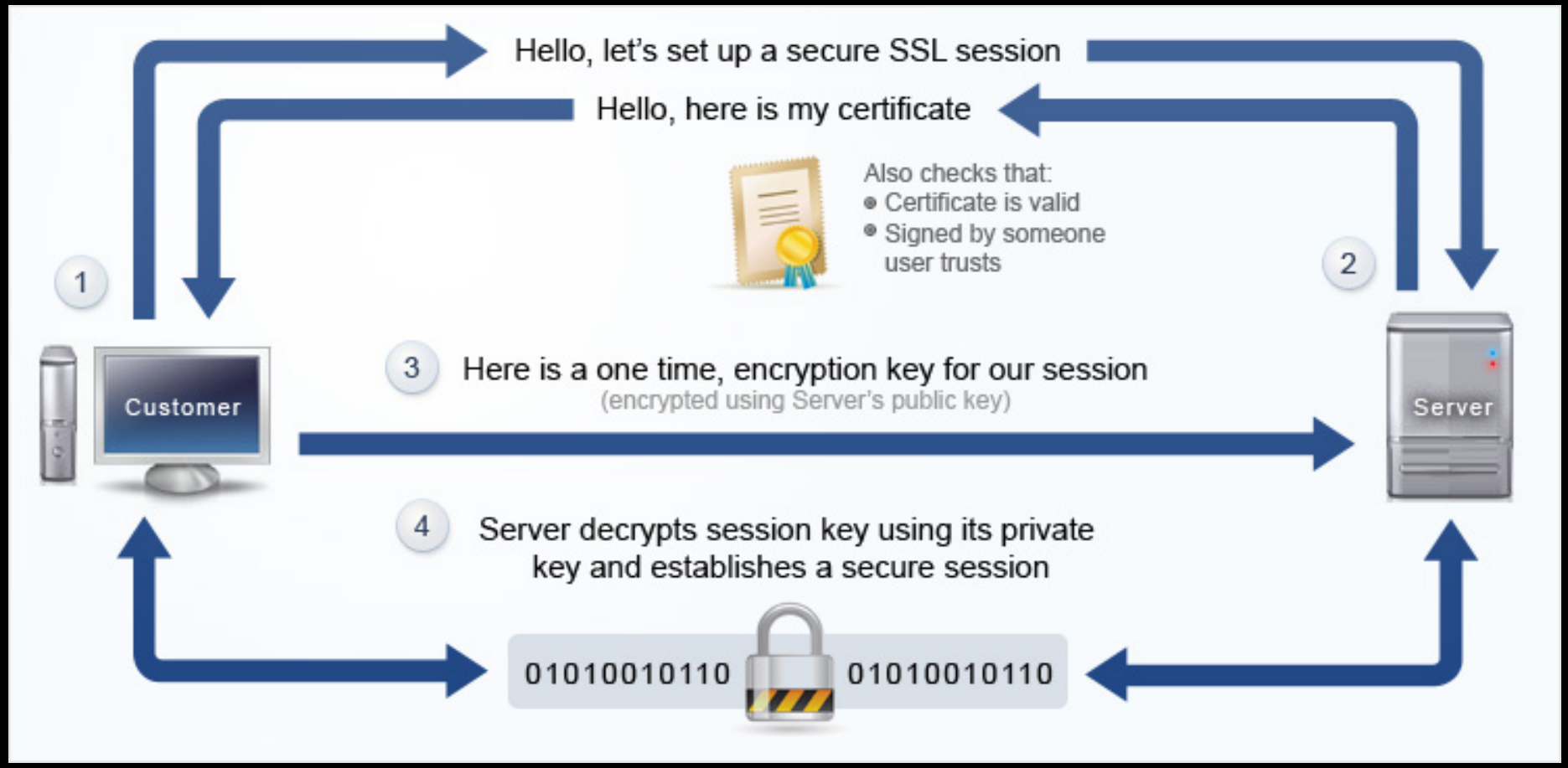
SSL/TLS

- SSL (Secured Socket Layer)
  - SSL v2.0 and v3.0: released in 1995 and 1996
- TLS (Transport Layer Security)
  - the improved version of SSL v3.0
  - more secure but little slower due to the two-step communication processes, i.e., server authentication and actual data transfer

출처 – https://www.quora.com/What-is-SSL-TLS-SSH-Secure-DNS-and-HTTPS

SSL/TLS Principle

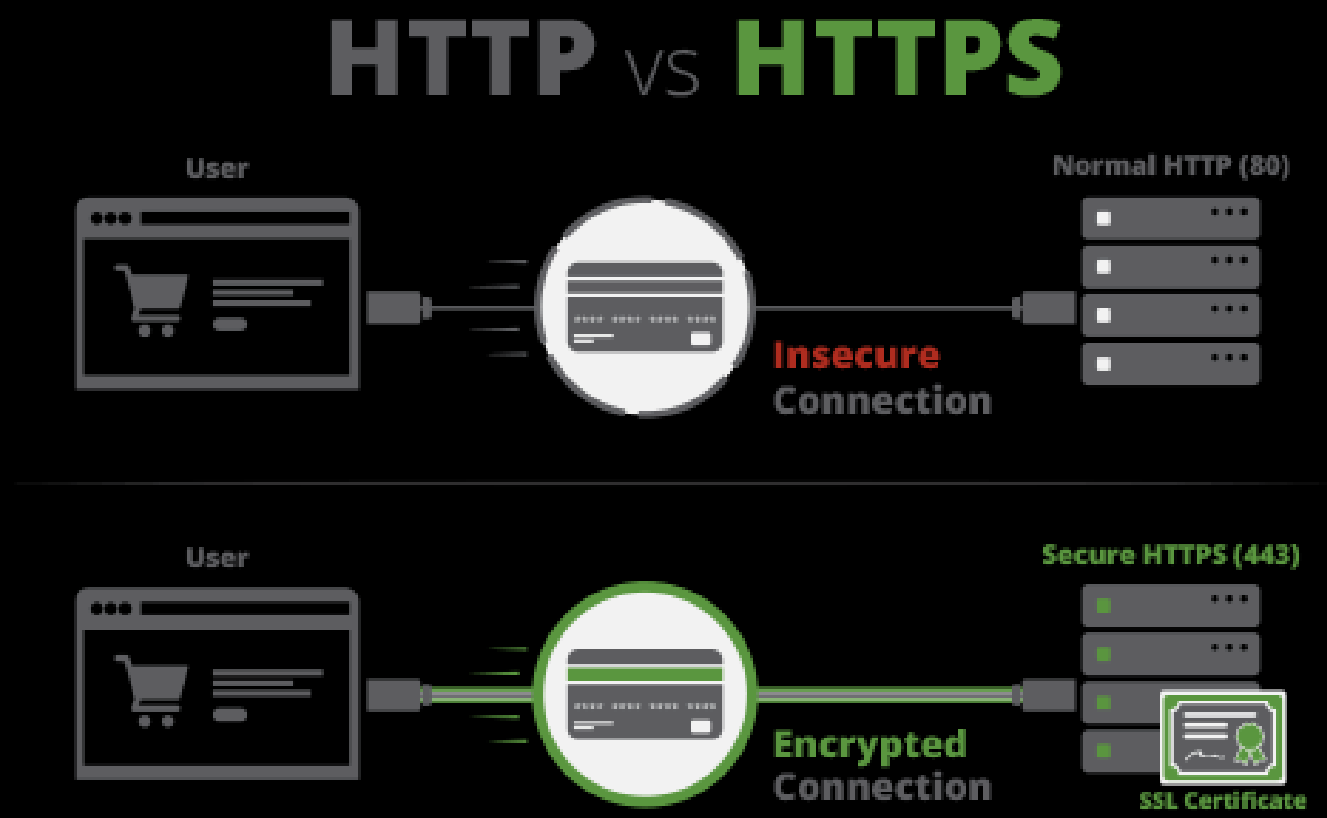
- Usage of the public–private (asymmetric) key pair system



출처 – https://www.msctrustgate.com/ssl_id.php

HTTPS = HTTP + SSL/TLS

출저 – http://blog.getpostman.com/2017/12/05/set-and-view-ssl-certificates-with-postman/

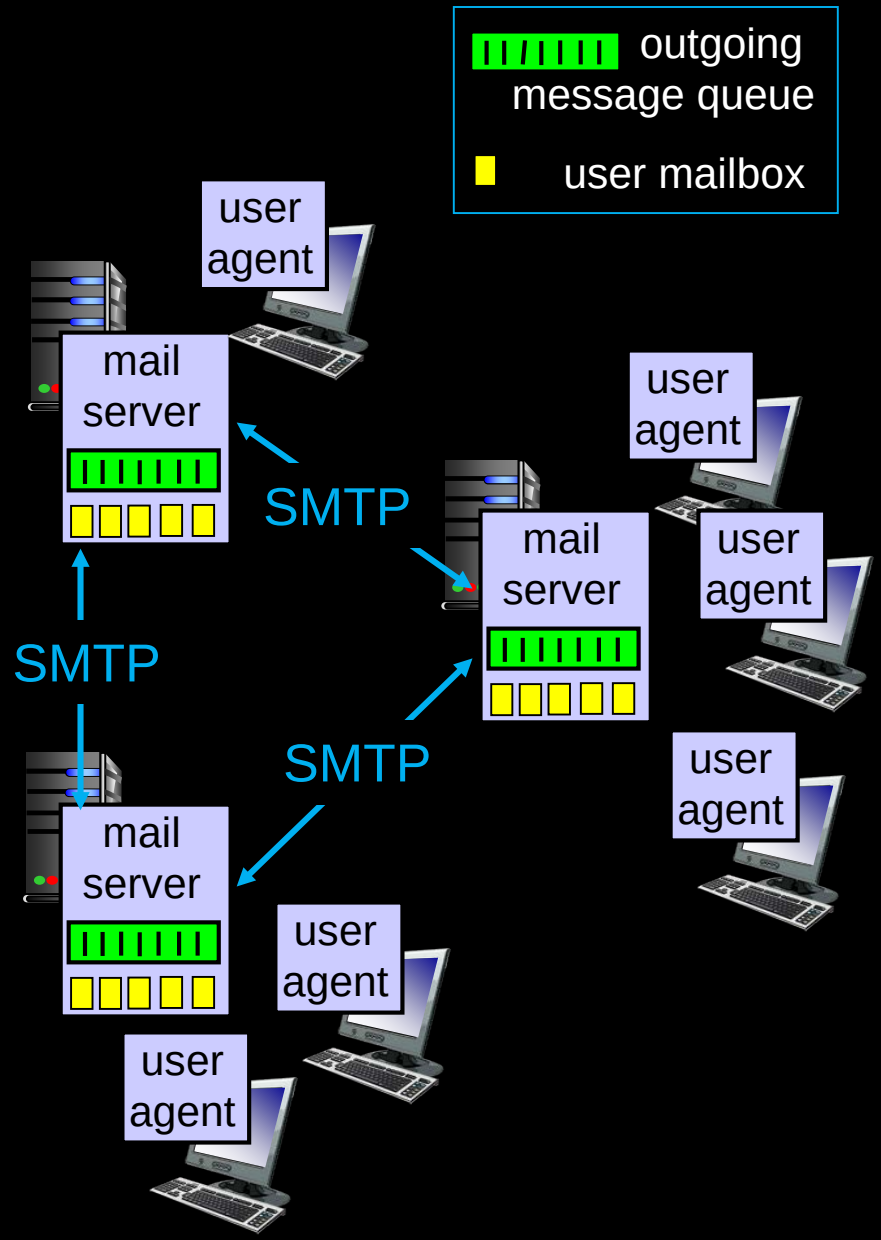


출처 – https://sucuri.net/guides/how-to-install-ssl-certificate

05. Electronic Mail

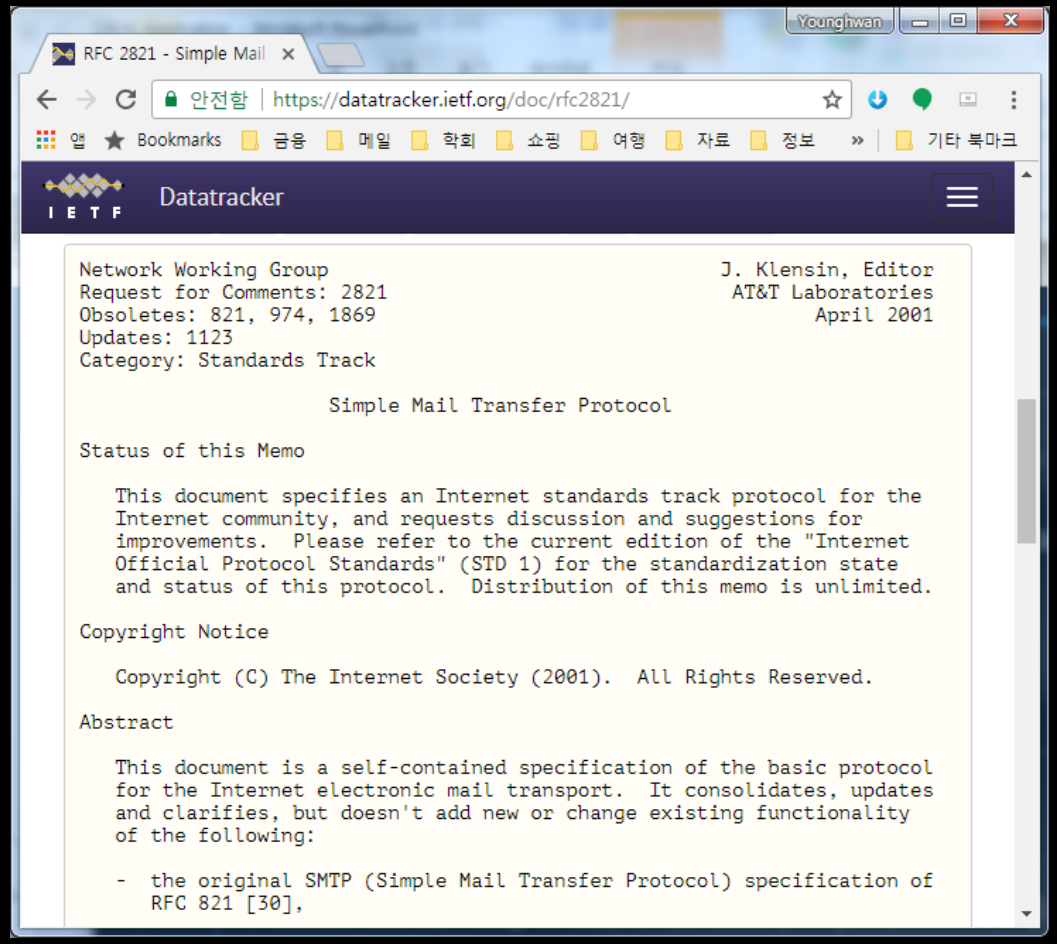
Electronic Mail

- Three components
  - user agents (clients): editing, reading
  - mail servers
  - protocols: SMTP, POP3, IMAP, …
- Components of mail servers
  - mailbox for incoming message
  - message queue for outgoing message
  - SMTP (Simple Mail Transfer Protocol)
    - Sending out email from a user
    - Exchanging between mail servers

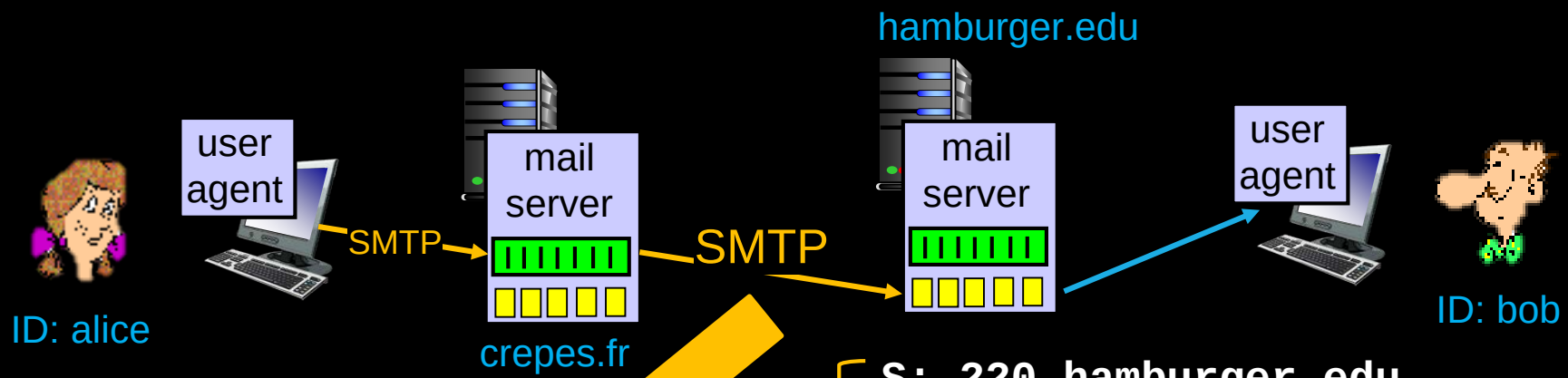


SMTP [RFC 2821]

- Uses TCP as the transport layer protocol for reliable email delivery from sending server to receiving server
- Three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- Command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase



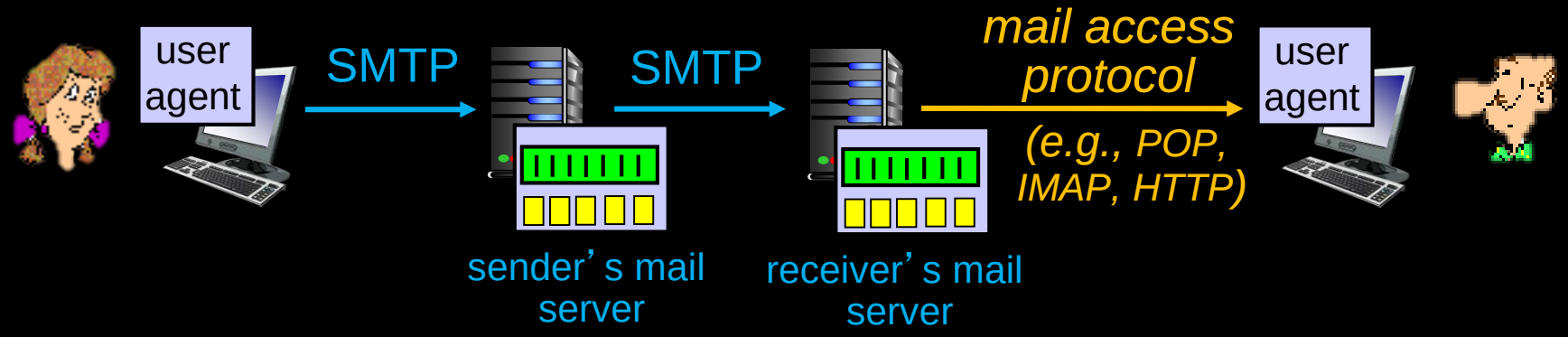
SMTP Example



After the TCP connection is established,

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Mail Access Protocols



POP3

- Post Office Protocol 3
- By default, deletes messages from the server after retrieving
- Disconnects from the server after download

IMAP

- Internet Mail Access Protocol
- Keeps all messages at server and allows user to organize message folders =〉 synchronization across devices
- Stays connected until the mail client app is closed and downloads messages on demand

HTTP

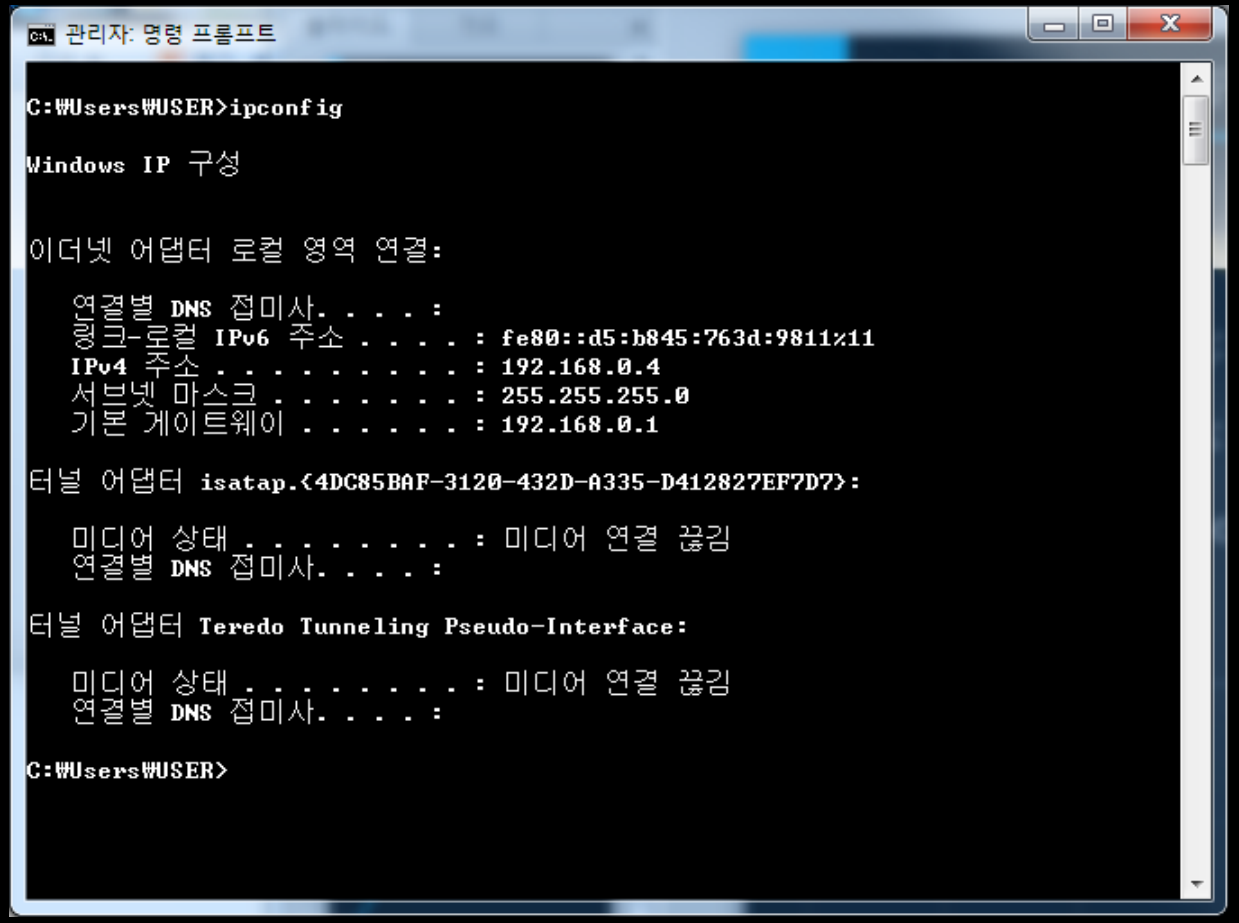
- Web-based email
- Used between browser and server (user-to-server, server-to-user)
- Hotmail in the mid 1990s
- Google, Yahoo!, etc.

06. Domain Name System

IP Address

- `ipconfig` command

- IP address is 32-bit long, represented by 4 numbers between 0~255
- Hard to remember the IP address of a server by the numbers
- Instead, people use the name of servers such as www.pusan.ac.kr



Domain Name System

- Actually, the most important part of the Internet for internetworking
- Brief view of the DNS operation

출처 – https://help.ubnt.com/hc/en-us/articles/115005817467-Intro-to-Networking-Domain-Name-System-DNS-

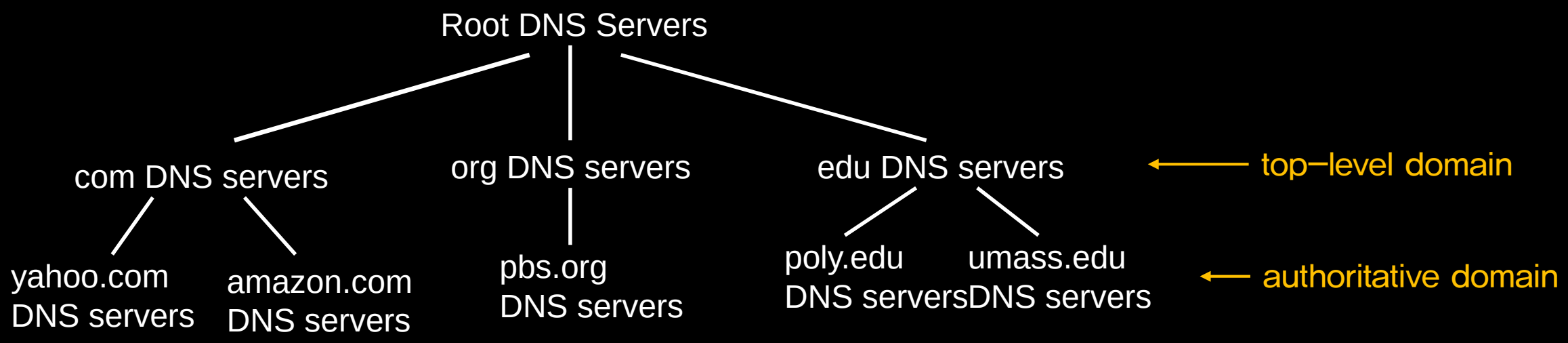
Domain Name System

- DNS services
  - hostname to IP address translation
  - load distribution
    - replicated Web servers: many IP addresses correspond to one name

- Distributed database system
- Why not centralize DNS?
  - single point of failure
  - traffic volume
  - distant centralized database

} Not scalable!!!

DNS: Distributed & Hierarchical Database

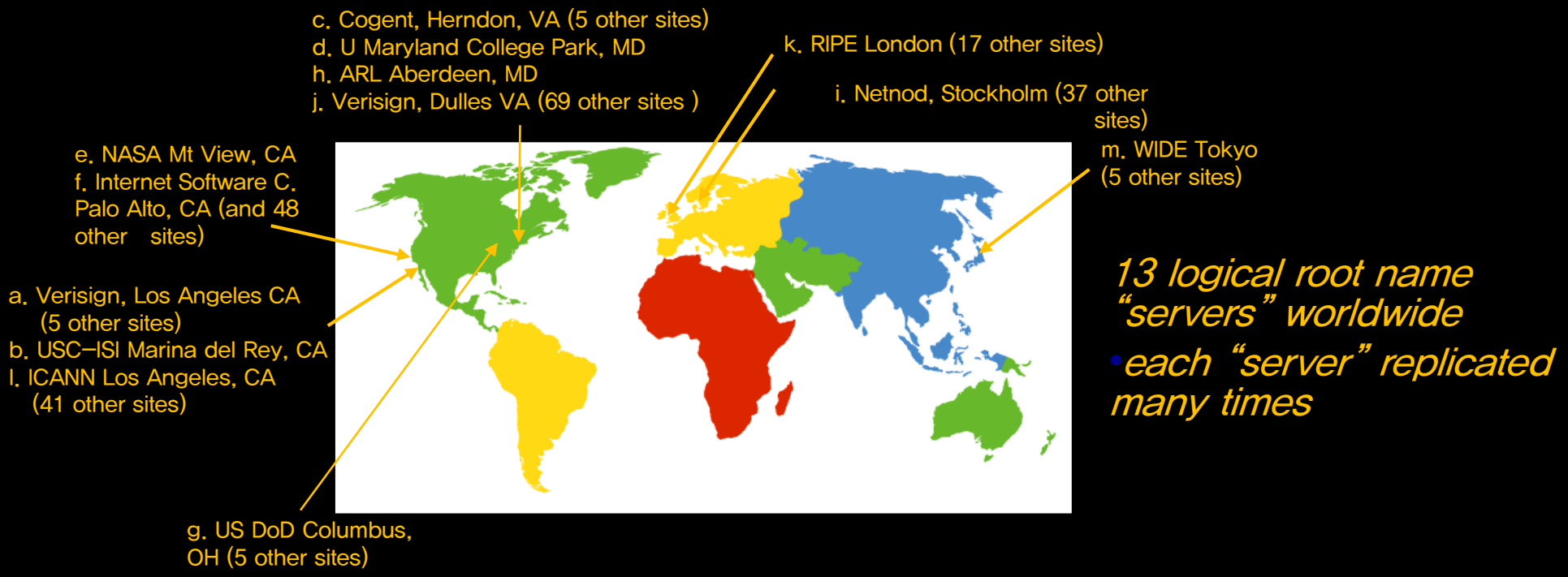


Client wants IP for www.amazon.com; 1st approximation:

1. Client queries Root server to find com DNS server
2. Client queries com DNS server to get amazon.com DNS server
3. Client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: Root Name Servers

- Contacted by local name server that can not resolve name
- Location of root name servers



13 logical root name "servers" worldwide

- *each "server" replicated many times*

DNS: Top-Level Domain (TLD) Servers

- Responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp, kr
- "Any language possible besides English alphabet" (June 20th, 2011)
- Network Solutions maintains servers for .com TLD
- Educause for .edu TLD

출처 – https://youcanbefound.com/do-top-level-domains-tlds-matter-for-seo-com-vs-brand/

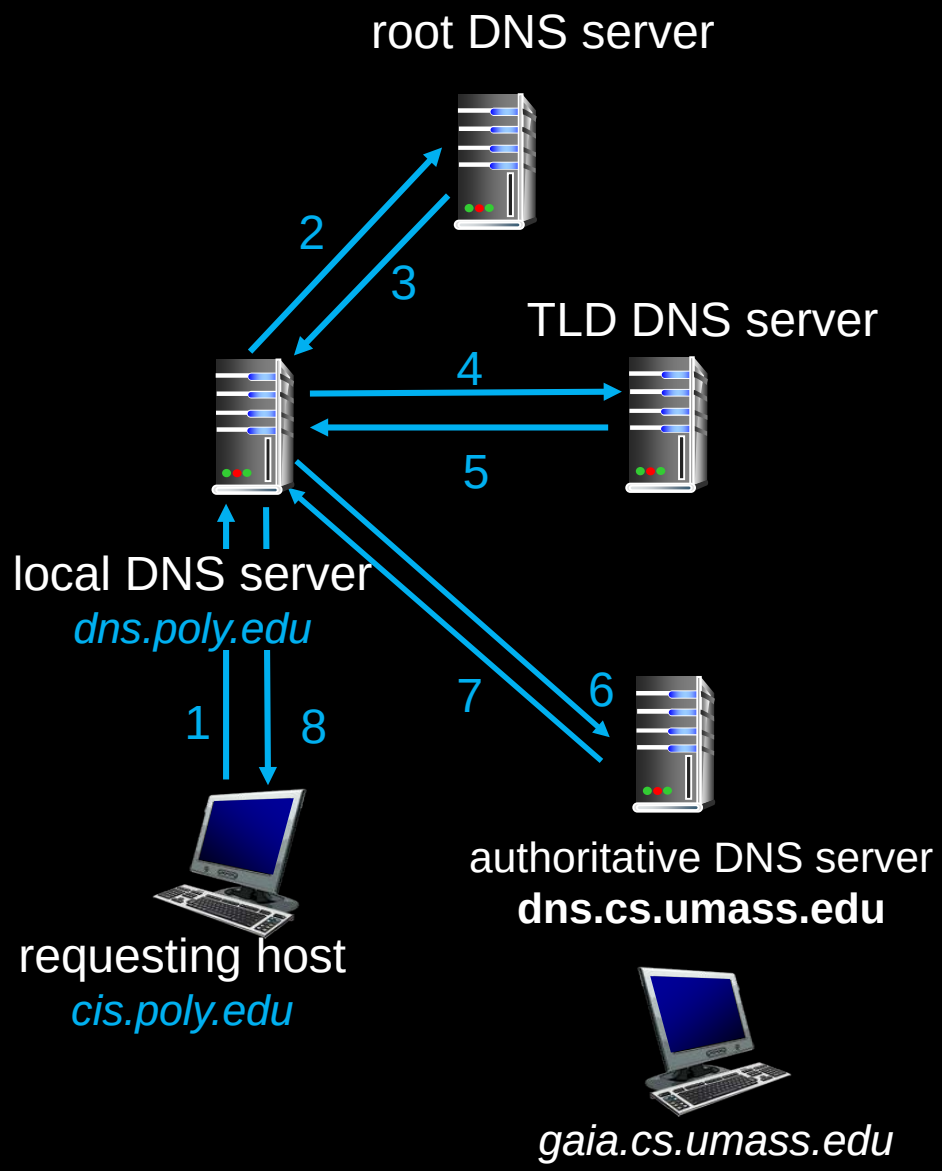
DNS: Authoritative Servers & Local Servers

- Authoritative Servers
  - organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
  - can be maintained by organization or service provider
- Local DNS servers
  - does not strictly belong to hierarchy
  - each ISP (residential ISP, company, university) has one
    - also called "default name server"
  - when host makes DNS query, query is sent to its local DNS server
    - has local cache of recent name–to–address translation pairs
    - acts as proxy, forwards query into hierarchy

DNS Name Resolution (1/2)

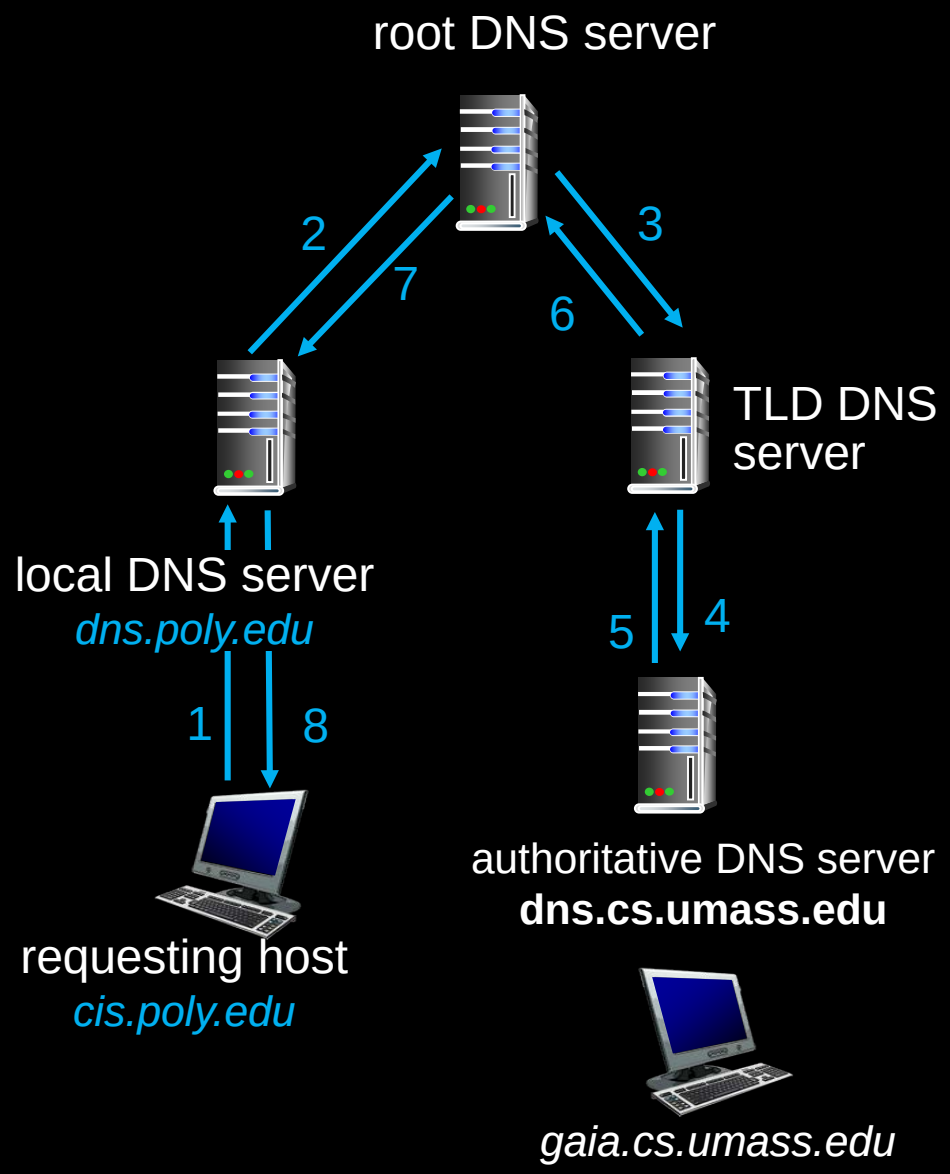
- Situation: "host at cis.poly.edu wants IP address for gaia.cs.umass.edu"

- Iterated query
  - contacted server replies with name of server to contact
  - "I don't know this name, but ask this server"



DNS Name Resolution (2/2)

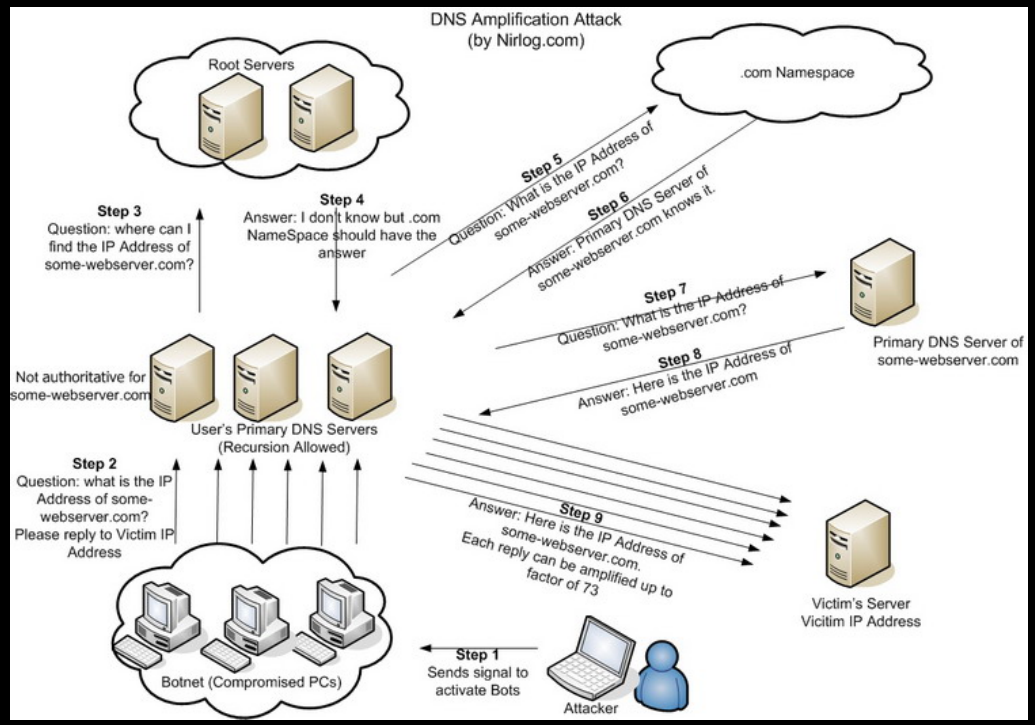
- Recursive query
  - puts burden of name resolution on contacted name server
  - not recommended due to
    - heavy load at upper levels of hierarchy
    - a security issue



Attacking DNS (1/2)

- DDoS attacks
  - bombard root servers with traffic
    - not successful to date
    - traffic filtering
    - local DNS servers cache IPs of TLD servers, allowing root server bypass
  - bombard TLD servers
    - potentially more dangerous

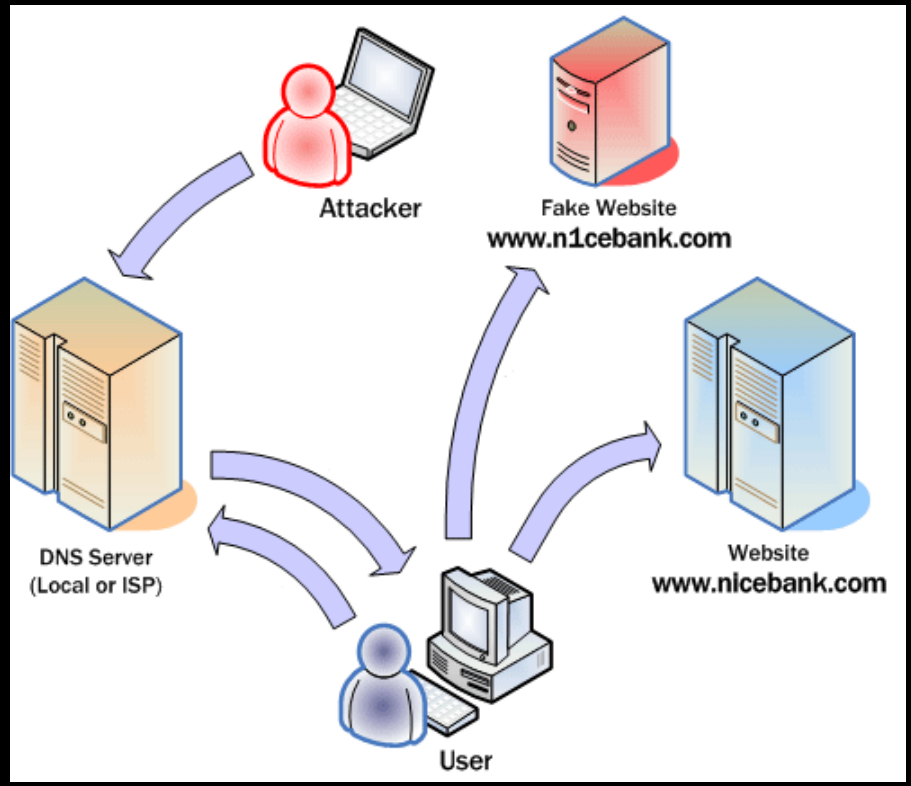
- Amplification attacks
  - exploit DNS for DDoS



출처 – http://nirlog.com/2006/03/28/dns-amplification-attack/

Attacking DNS (2/2)

- Pharming attacks
  - Private data + Farming
    - domain hijacking
    - DNS poisoning: registration of bogus sites

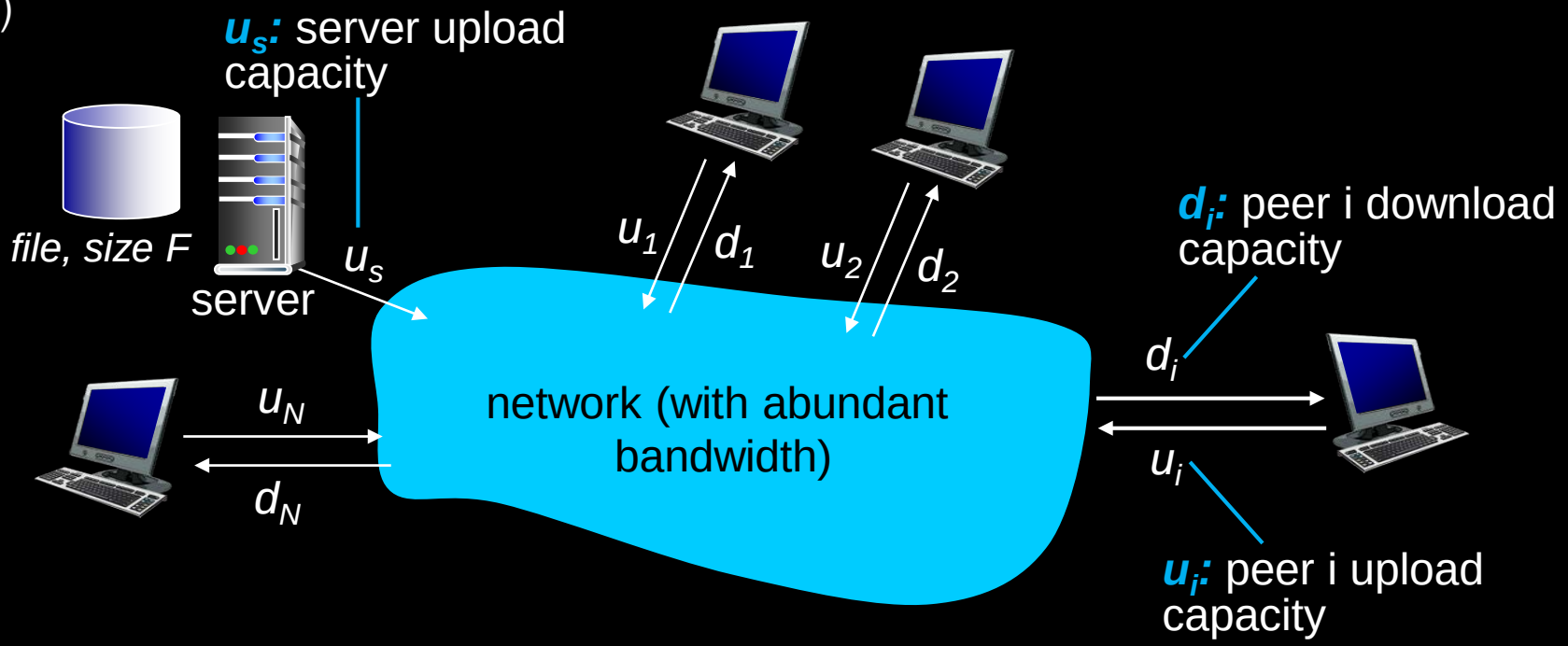


출처 – http:// www.petervaldivia.com/technology/networks/dns.php

07. Peer-to-Peer Application

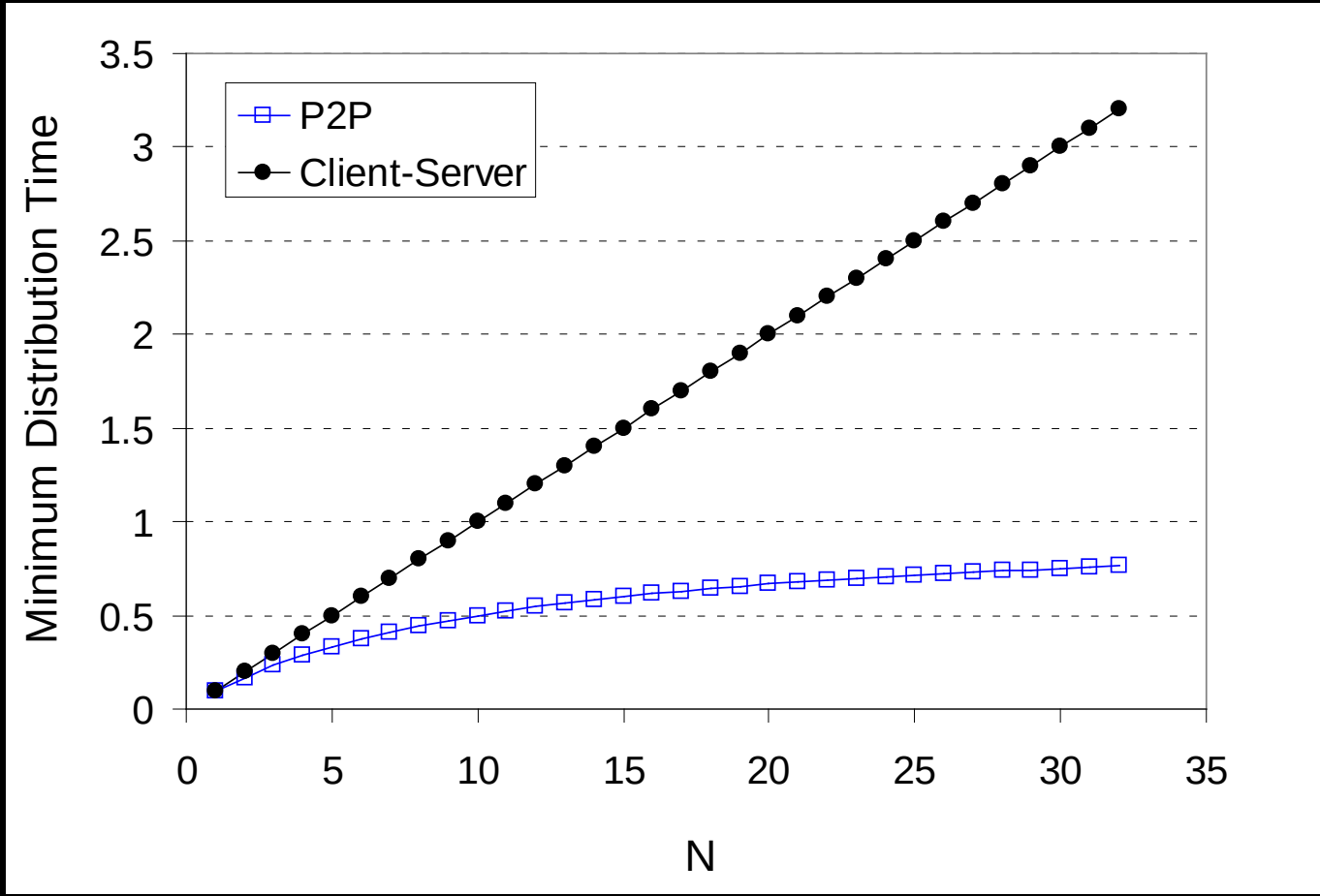
P2P Architecture

- No always–on server
- Arbitrary end systems directly communicate
- Peers are intermittently connected and change IP addresses
- Examples:
  - file distribution (BitTorrent)
  - streaming (KanKan)
  - VoIP (Skype)



File Distribution: Client–Server vs P2P

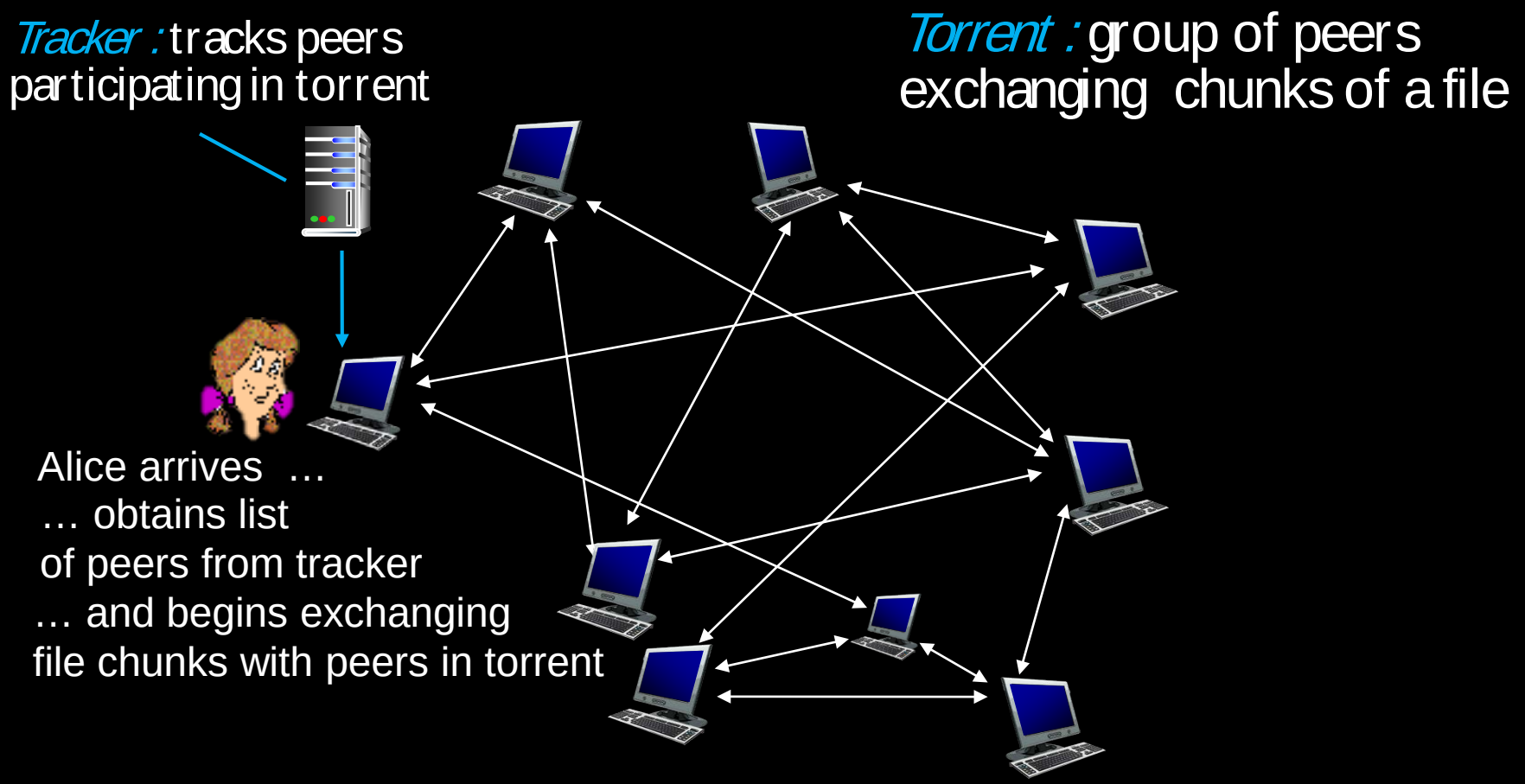
- *Question:* "How much time to distribute file (size *F*) from one server to *N* peers?"



"P2P is scalable!!!"

Case Study: BitTorrent

- File divided into 256 kb chunks
- Peers in torrent send/receive file chunks



BitTorrent: Receiving and Sending Chunks

- Chunk receiving
  - at any given time, different peers have different subsets of file chunks
  - periodically, Alice asks each peer for list of chunks that they have
  - Alice requests missing chunks from peers, rarest first

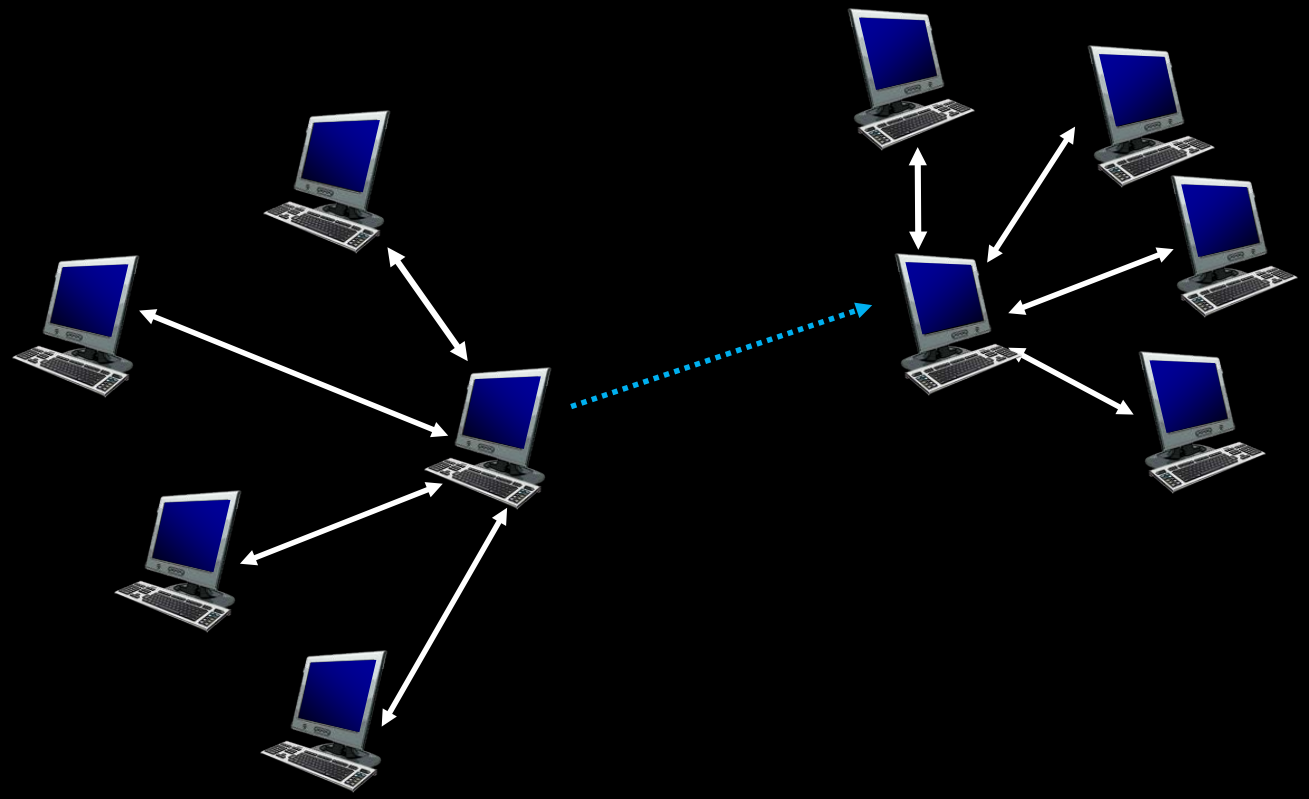
Free-rider

peer usually wants to receive file without sending file to others

- Chunk sending: **tit-for-tat**
  - Alice sends chunks to those four peers currently sending her chunks at highest rate
    - other peers are choked by Alice (do not receive chunks from her)
    - re-evaluate top 4 every 10 secs
  - every 30 secs: randomly select another peer, starts sending chunks
    - "optimistically unchoke" this peer
    - newly chosen peer may join top 4

BitTorrent：tit–for–tat

1） Alice “optimistically unchokes” Bob

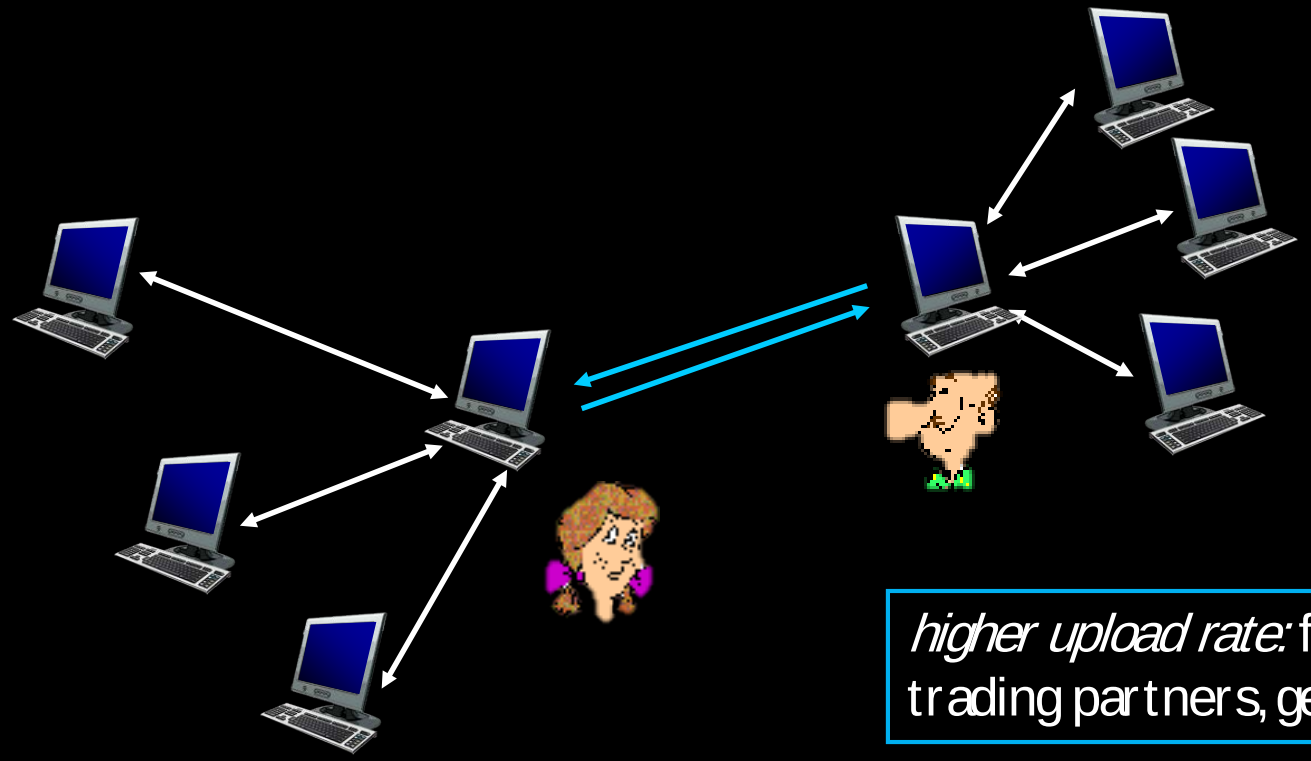


BitTorrent: tit-for-tat

1) Alice "optimistically unchokes" Bob

2) Alice becomes one of Bob's top-four providers; Bob reciprocates

3) Bob becomes one of Alice's top-four providers

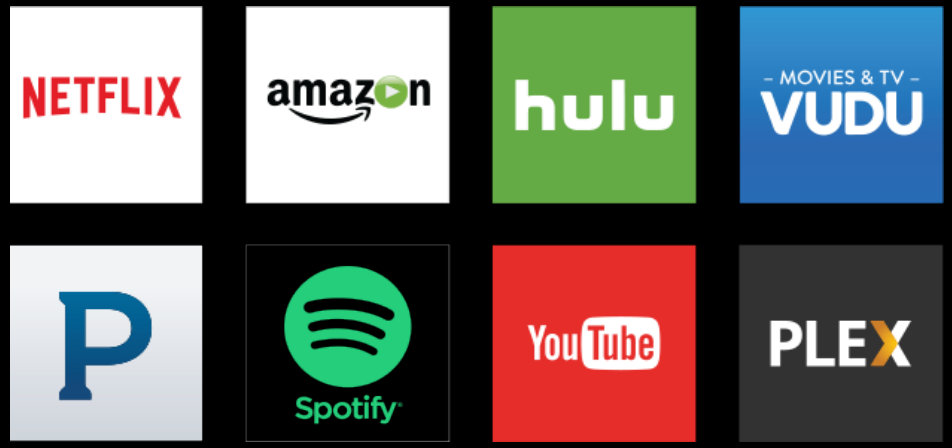


higher upload rate: find better trading partners, get file faster !

08. Video Streaming and CDNs

Video Traffic

- Video traffic: major consumer of Internet bandwidth
  - Netflix, YouTube: 37%, 16% of downstream residential ISP traffic
  - ~1B YouTube users, ~75M Netflix users



출처 – https://www.tivo.com/OTA-antenna-DVR-recorder

- Challenge: Scalability
  - single mega-video server issues
    - single point of failure
    - point of network congestion
    - long path to distant clients
    - multiple copies of video sent over outgoing link

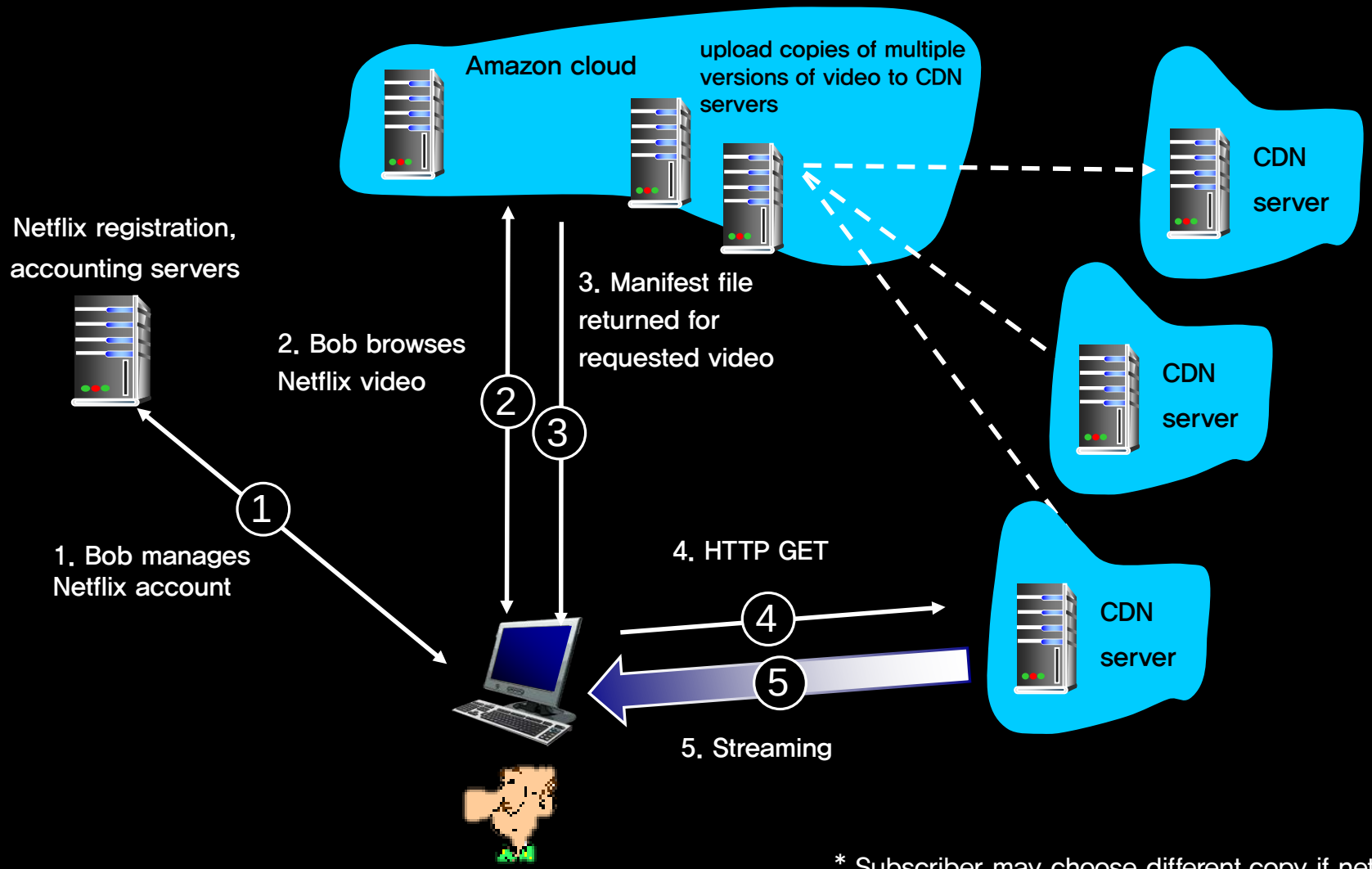
출처 – https://orpical.com/should-you-use-a-content-delivery-network/

Content Distribution Networks

- Store/serve multiple copies of videos at multiple geographically distributed sites

출처 – https://orpical.com/should-you-use-a-content-delivery-network/

Case Study: Netflix



* Subscriber may choose different copy if network path congested

Summary

Summary (1/2)

01 Principles of application
- network architecture: client–server vs. P2P
- application protocol layer

02 Web and HTTP
- HTTP (HyperText Transfer Protocol)
- HTTP history

03 Cookies and web–caching
- cookies: user convenience vs. privacy
- web–caching: latency reduction and service scalability

04 SSL/TLS
- security over transport layer protocol
- HTTPS (HTTP Secure)

Summary (2/2)

05 Electronic mail

- SMTP (Simple Mail Transfer Protocol)
- mail access protocols: POP, IMAP, HTTP

06 Domain name system

- the most important part for Internet internetworking
- hierarchical and distributed database

07 Peer-to-peer application

- efficient for service scalability
- case study: BitTorrent

08 Video streaming and CDNs

- Video traffic is the most major consumer of Internet traffic
- CDNs: multiple geographically distributed servers for scalability