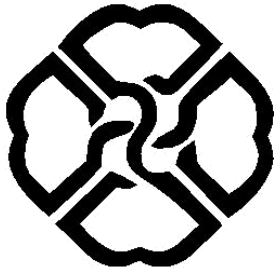


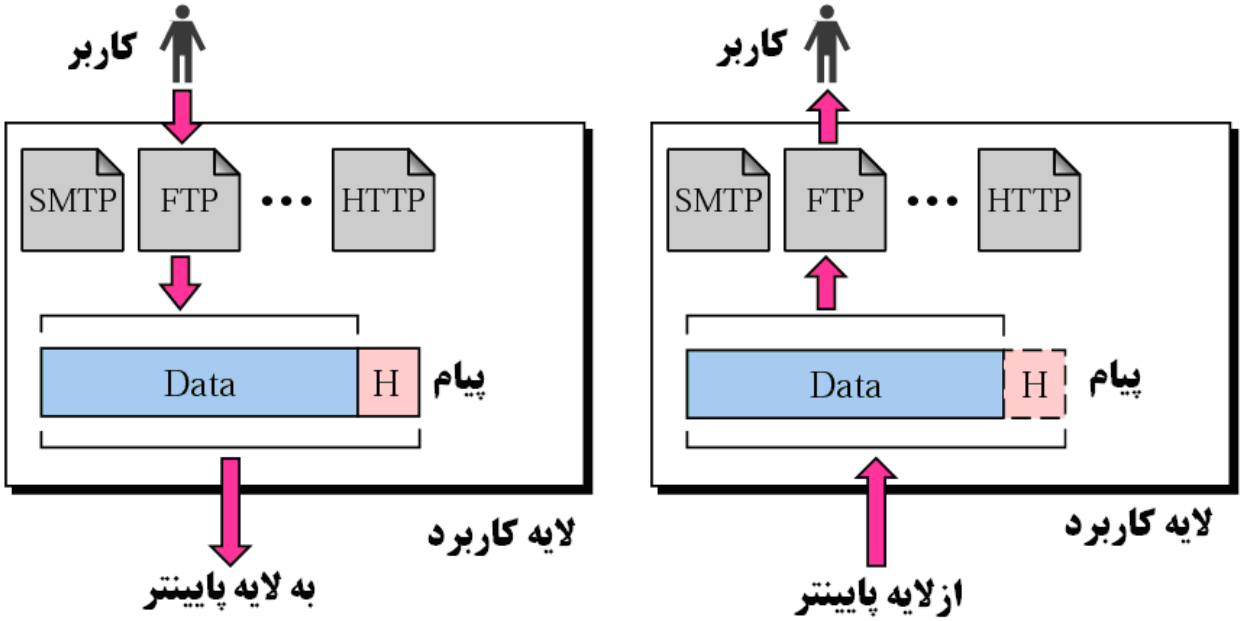
Department of Computer and IT Engineering
University of Kurdistan

Advanced Computer Networks

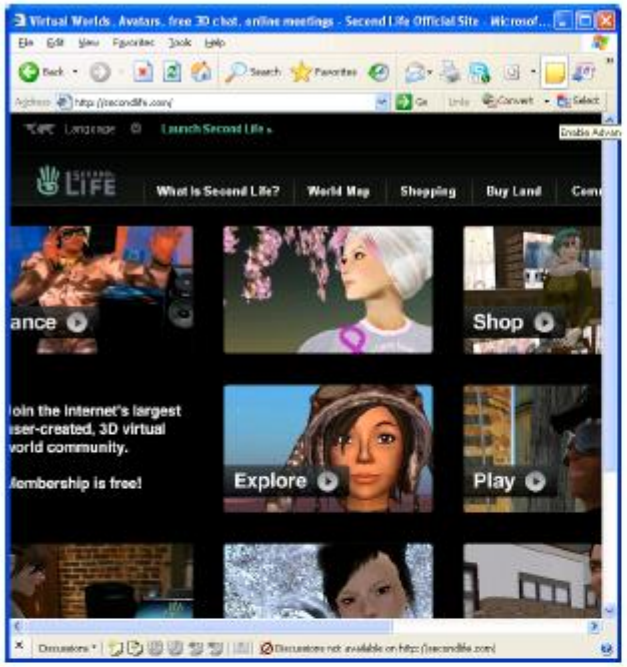
Application Layer

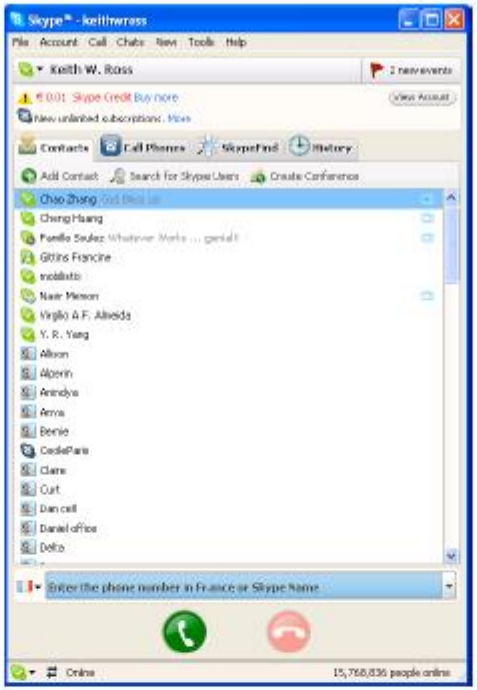
By: Dr. Alireza Abdollahpouri

Application Layer



لایه کاربرد مستقیما با کاربر (برنامه ها یا اشخاص) در ارتباط است. این لایه از طریق پروتکلهای مختلفی که در اختیار دارد، خدمات مورد نیاز کاربران را فراهم میآورد. هر کدام از پروتکلهای این لایه بسته به نوع و ماهیت آنها از یکی از پروتکلهای TCP یا UDP در لایه پایینتر استفاده میکنند.



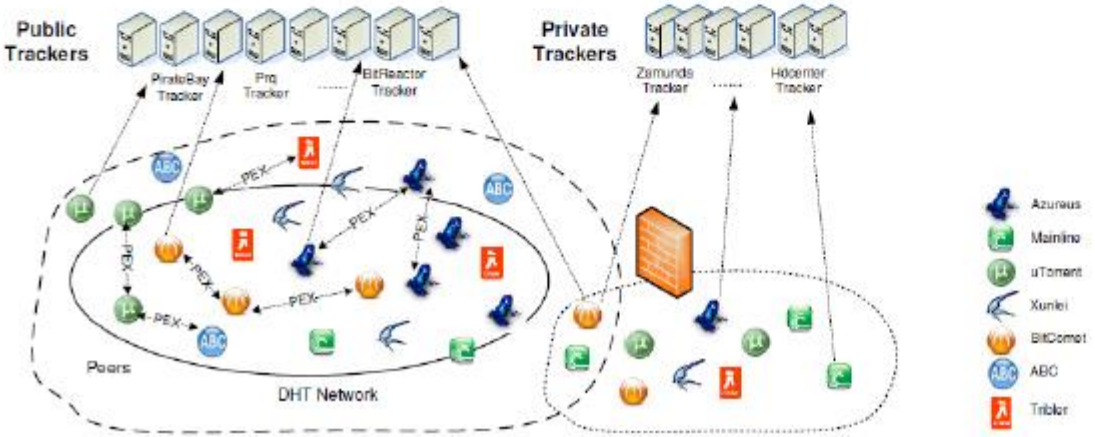


Public Discovery Sites

Private Discovery Sites

Public Trackers

Private Trackers



Public Torrents

Private Torrents

Application-Layer Protocols

مشهورترین پروتکلهای این لایه عبارتند از:

FTP: پروتکلی برای انتقال فایل
HTTP: پروتکلی برای دسترسی به صفحات وب
DNS: پروتکلی برای ترجمه نامهای نمادین به آدرسهای IP
Telnet: پروتکلی برای ورود به سیستم از راه دور
SMTP و POP3: پروتکلهایی برای ارسال و دریافت E-mail

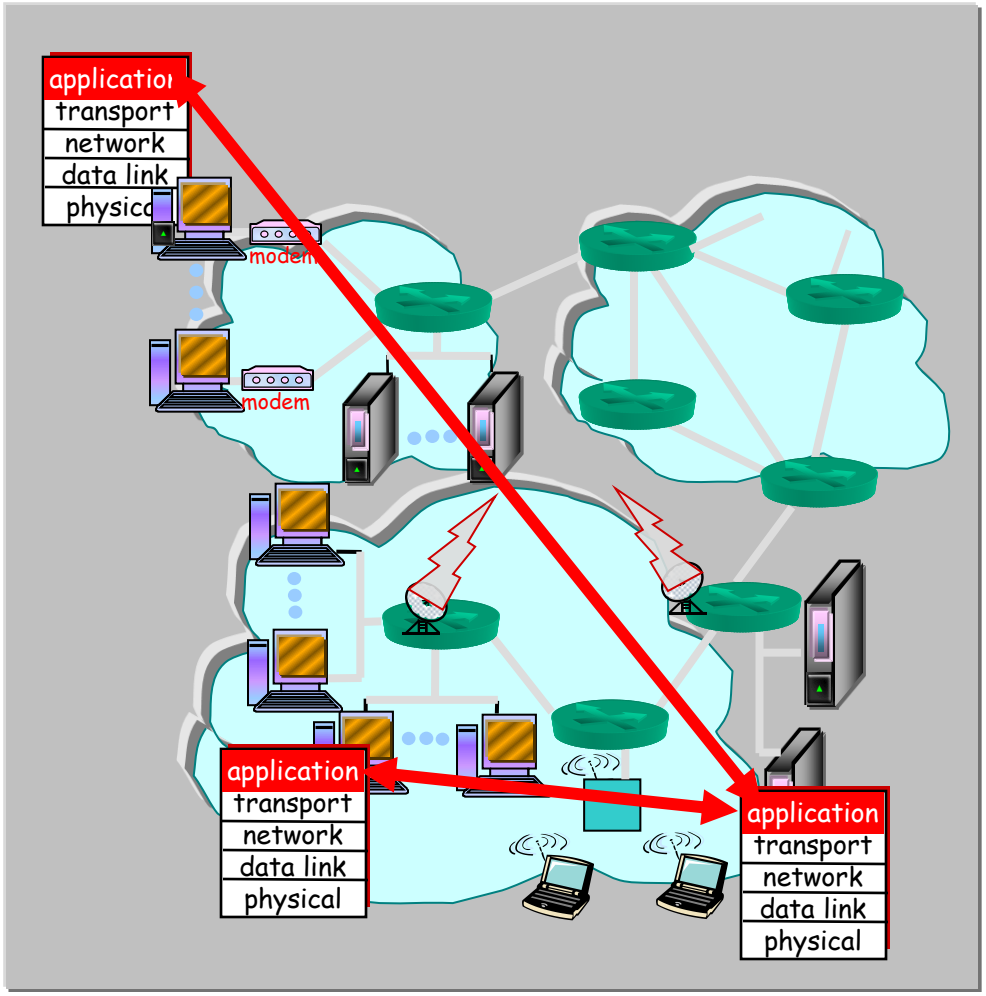
Applications and Application-Layer Protocols

Application: communicating, distributed processes

- e.g., e-mail, Web, P2P file sharing, instant messaging
- running in end systems (hosts)
- exchange messages to implement application

Application-layer protocols

- one “piece” of an app
- define messages exchanged by apps and actions taken
- use communication services provided by lower layer protocols (TCP, UDP)



Client-Server Paradigm

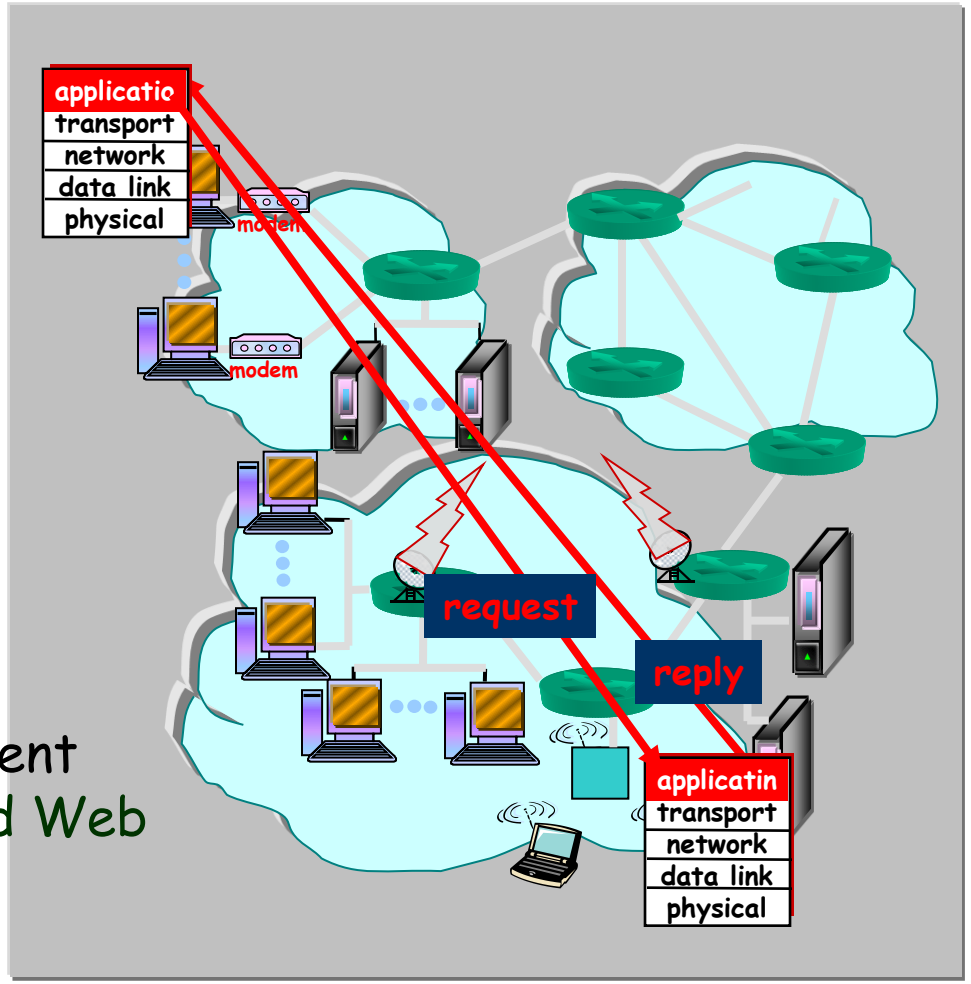
Typical network app has two pieces: *client* and *server*

Client:

- initiates contact with server ("speaks first")
- typically requests service from server,
- Web: client implemented in browser; e-mail: in mail reader

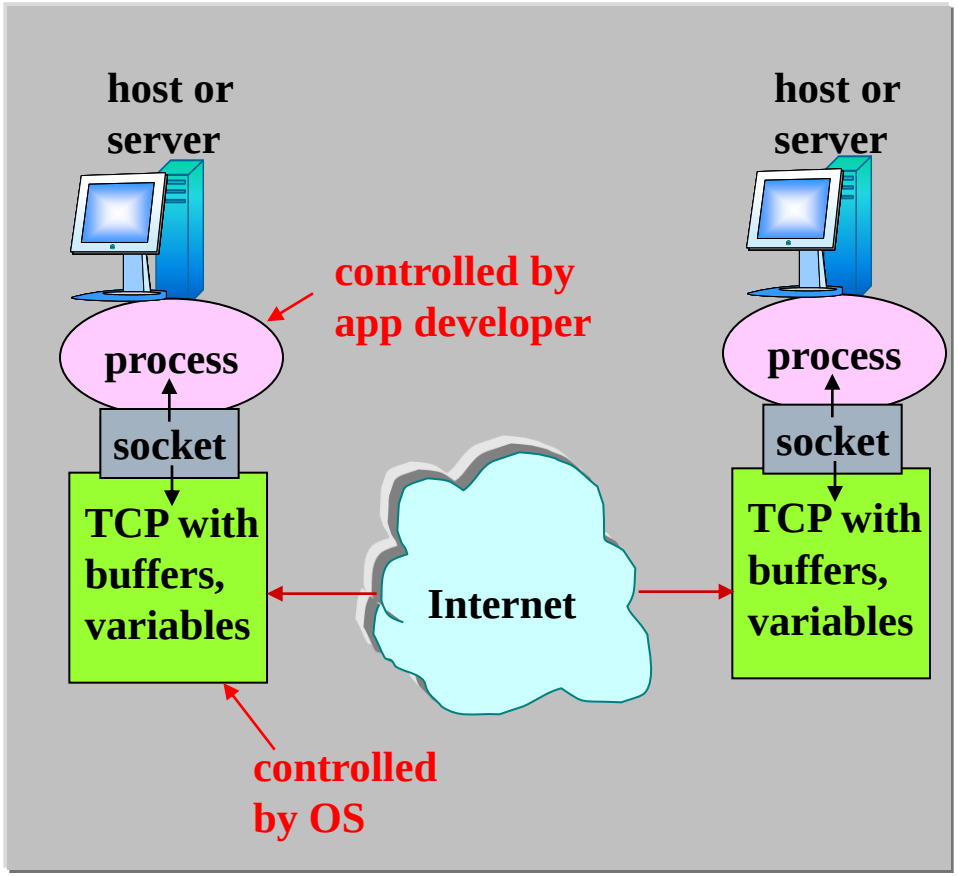
Server:

- provides requested service to client
- e.g., Web server sends requested Web page, mail server delivers e-mail



Processes Communicating Across Network

- process sends/receives messages to/from its socket
- socket analogous to door
  - sending process pushes message out door
  - sending process assumes transport infrastructure on other side of door which brings message to socket at receiving process



What transport service does an app 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

Requirements of Selected Network Applications

Application	Data loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
web documents	no loss	elastic (few kbps)	no
real-time audio/video	loss-tolerant	audio: few kbps-1Mbps video:10kbps-5Mbps	yes, 100s of msec
stored audio/video	loss-tolerant	same as above	yes, few sec
interactive games	loss-tolerant	few kbps-10kbps	yes, 100s of msec
instant messaging	no loss	elastic	yes and no

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
- *no guarantee on:* timing, minimum bandwidth

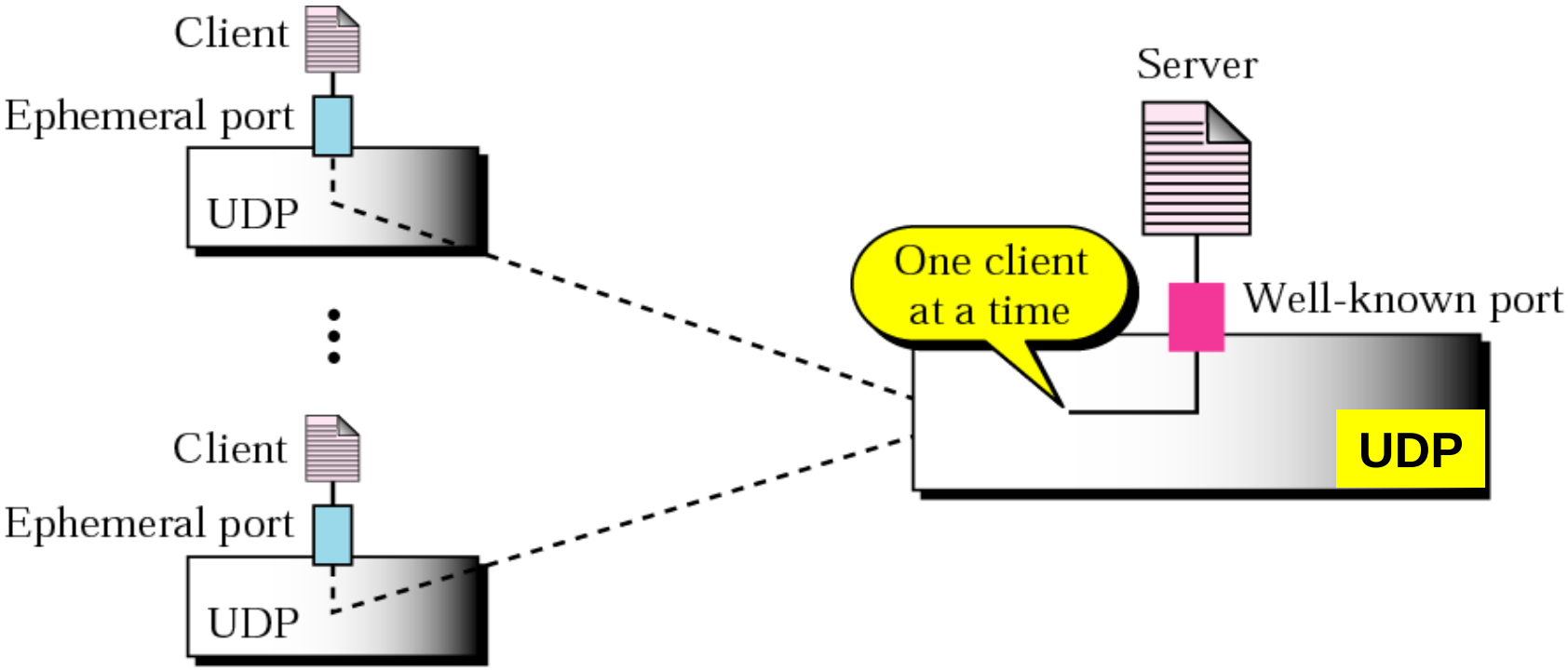
UDP service:

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

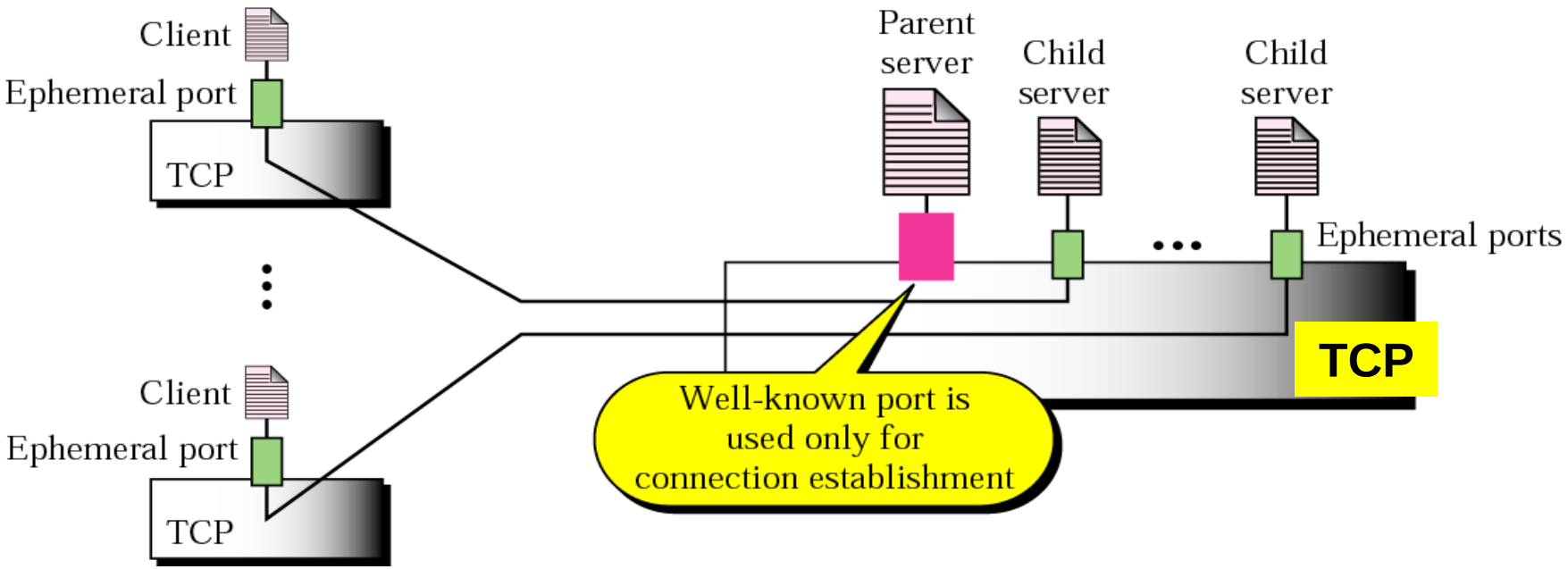
Internet apps: application, transport protocols

applications	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
Name server	DNS [RFC 1034]	UDP or TCP
streaming multimedia	proprietary (e.g., youtube)	Typically UDP

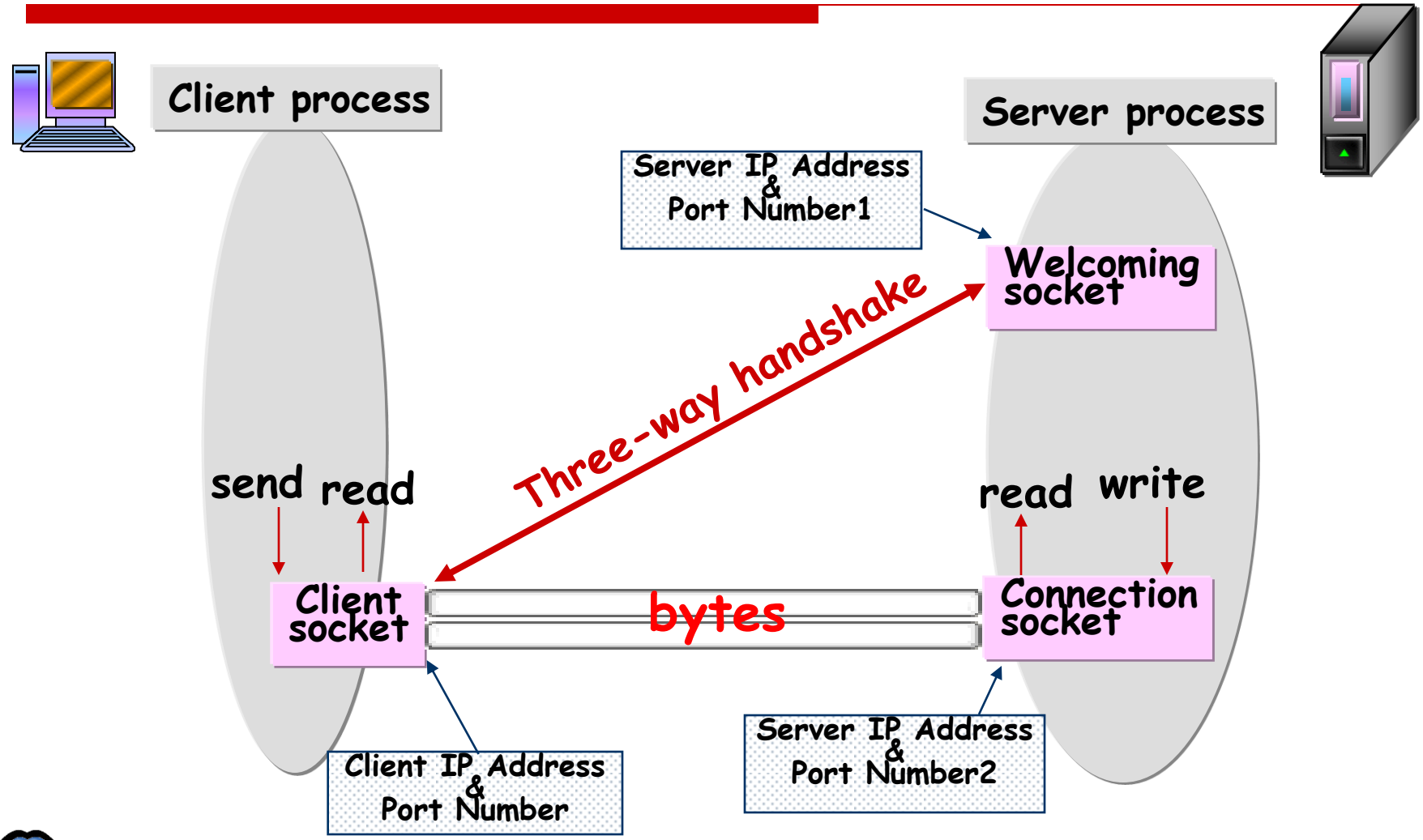
Connectionless iterative server



Connection-oriented concurrent server



Sockets


Client process
Server process
Server IP Address & Port Number1
Welcoming socket
Three-way handshake
send
read
read
write
Client socket
bytes
Connection socket
Client IP Address & Port Number
Server IP Address & Port Number2

Domain Name System (DNS)

همانطور که می‌دانیم در اینترنت از آدرس **IP** برای شناسایی یک میـزبان استفـــاده می‌گردد. اما آنچه مسلم است این است که برای کاربران کار با اسامی بسیار آسانتر از کار با اعداد است. به عنوان مثال به خاطر سپاری آدرس به فرم www.google.com بسیار راحت‌تر از به‌خاطرسپاری همان آدرس به فرم ۶۴.۲۳۳.۱۶۹.۹۹ می‌باشد. بنابراین احتیاج به سیستمی داریم که بتواند یک نام را به آدرس **IP** آن تبدیل نماید.

در اولین سالهای راه اندازی شبکه اینترنت، راه حل بسیار ساده ای برای ترجمه نامهای نمادین به آدرسIP وجود داشت و آن تعریف تمام نامها و آدرسهایIP معادل، در یک فایل بنام hosts.txt بود. این فایل دارای دو ستون بود که در یک طرف آدرس نمادین و در طرف دیگر آدرسIP معادل آن نوشته شده بود. به دلیل اینکه در آن تاریخ تعداد آدرسها زیاد نبود، حجم چنین فایلی چندان بزرگ نمی‌شد و هر ماشین میزبان می توانست یک نسخه از این فایل را در اختیار داشته باشد و ساعت ۲٤ هر شب این فایل را از روی فایل مرجع تازه‌سازی و به‌روز می‌کرد تا هر گونه تغییر احتمالی و تعریف آدرسهای جدید اعمال شود. بدیهی است که امروزه با حجم میلیونی آدرسها در اینترنت، داشتن یک فایل متمرکز و قرار دادن تمام آدرسها و معادل آدرسIP در آن، امکان پذیر نیست.

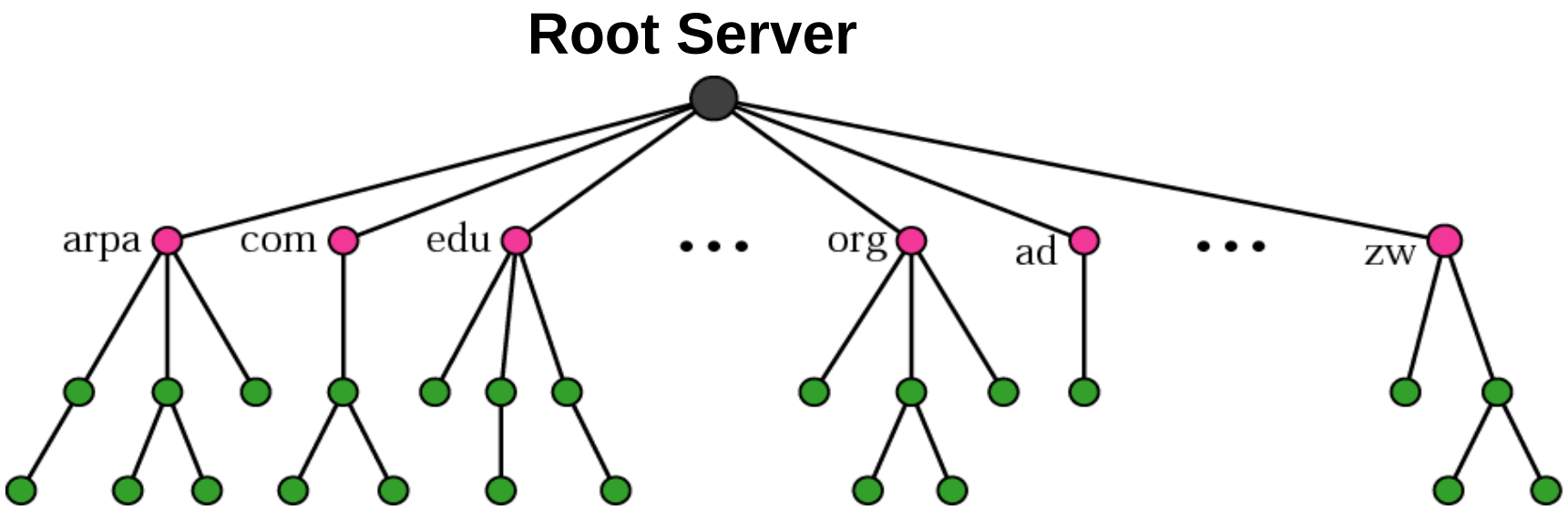
راه حل: استفاده از یک پایگاه داده سلسله مراتبی توزیع شده

DNS System

- اساساً یک پایگاه داده توزیع شده از نگاشت های نام به آدرس IP در سطح گسترده است.
- اهداف :

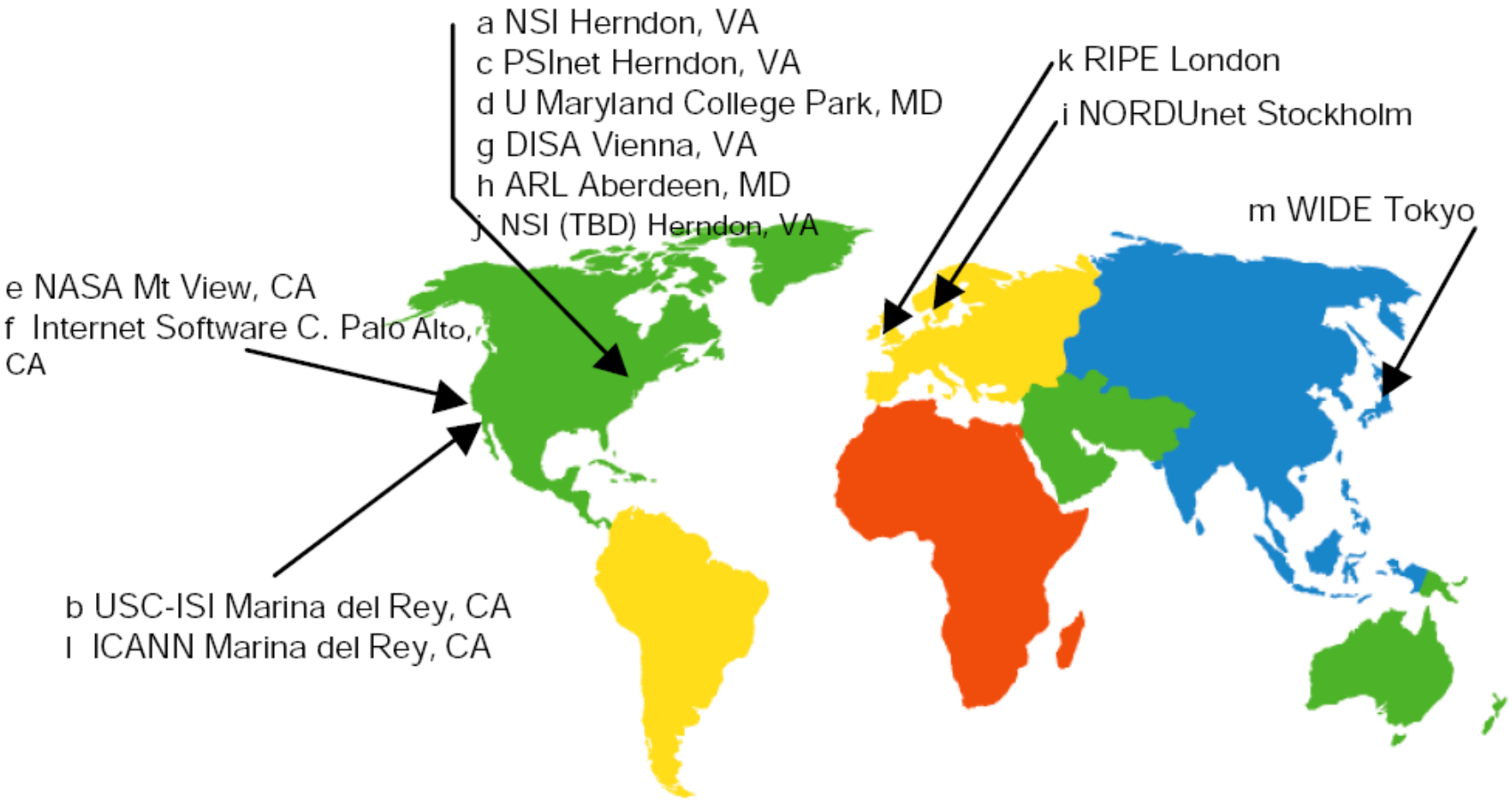
- قابلیت مقیاس پذیری
- عدم وابستگی به یک سرور مرکزی
- استحکام

Hierarchical Distributed Name Space



تعداد ۱۳ عدد سرویس دهنده ریشه وجود دارد

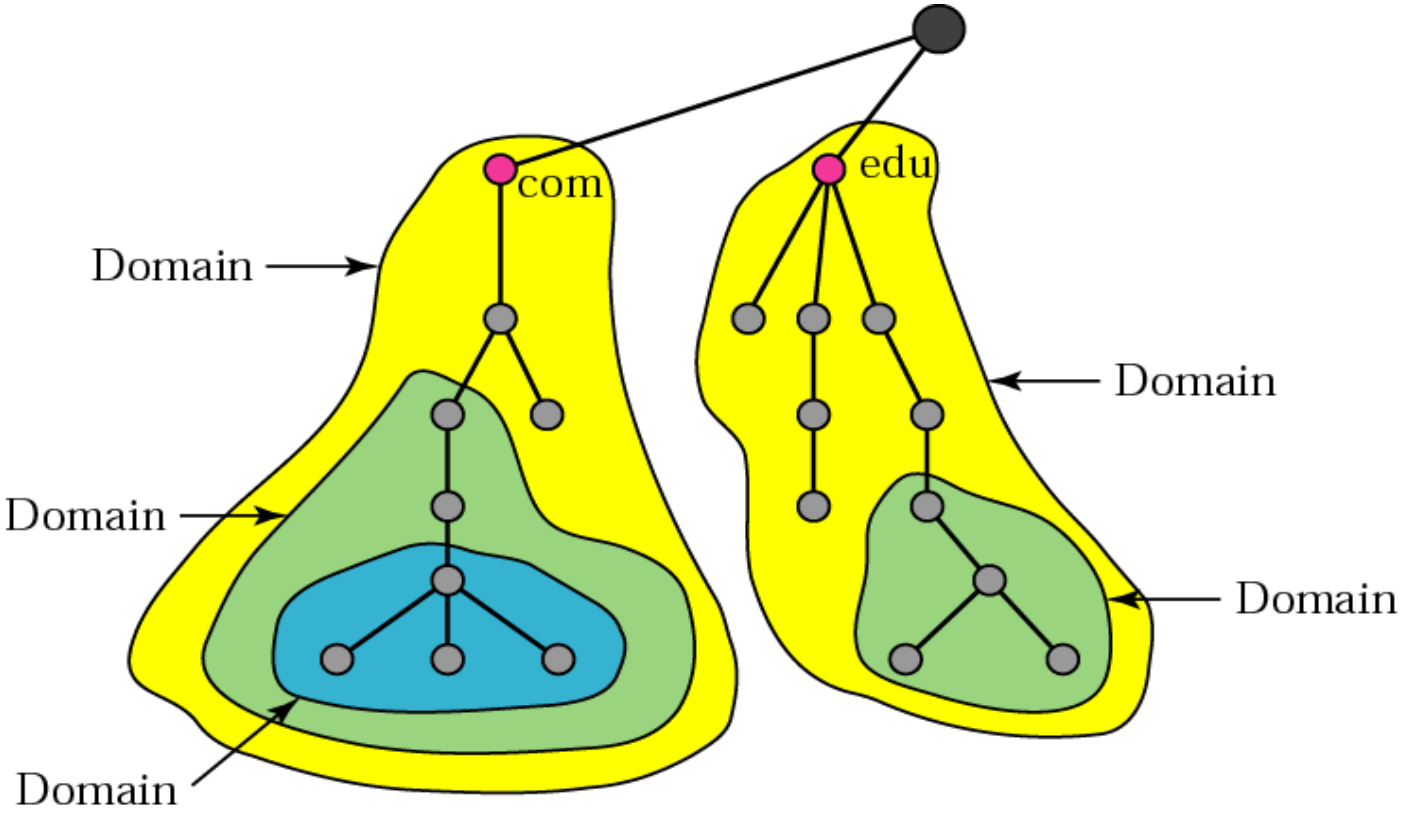
DNS root Servers



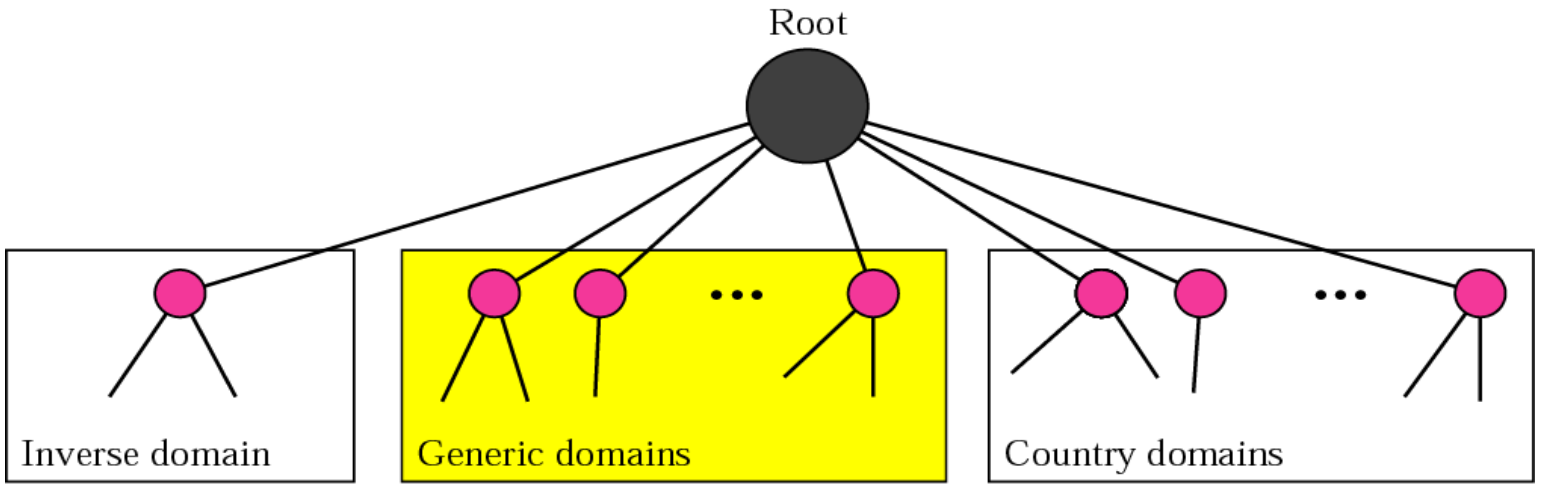
DNS root Servers

Server	Operator	Cities	IP Addr
A	VeriSign Global Registry Services	Herndon VA, US	198.41.0.4
B	Information Sciences Institute	Marina Del Rey CA, US	128.9.0.107
C	Cogent Communications	Herndon VA, US	192.33.4.12
D	University of Maryland	College Park MD, US	128.8.10.90
E	NASA Ames Research Center	Mountain View CA, US	192.203.230.10
F	Internet Software Consortium San Francisco CA, US	Palo Alto CA, US; IPv6: 2001:500::1035	IPv4: 192.5.5.241
G	U.S. DOD Network Information Center	Vienna VA, US	192.112.36.4
H	U.S. Army Research Lab	Aberdeen MD, US	128.63.2.53
I	Autonomica	Stockholm, SE	192.36.148.17
J	VeriSign Global Registry Services	Herndon VA, US	192.58.128.30
K	Reseaux IP Europeens—Network Coordination Centre	London, UK	193.0.14.129
L	Internet Corporation for Assigned Names and Numbers	Los Angeles CA, US	198.32.64.12
M	WIDE Project	Tokyo, JP	202.12.27.33

Domains

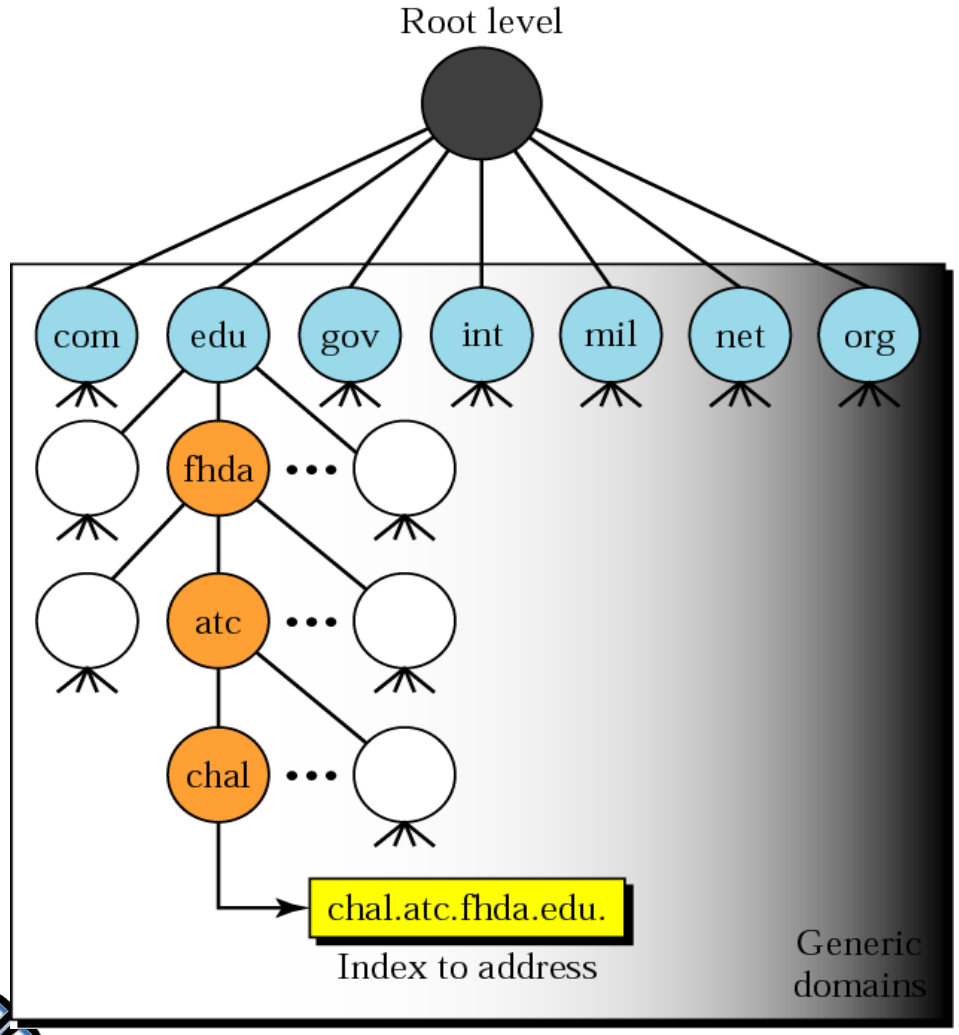


DNS in the Internet



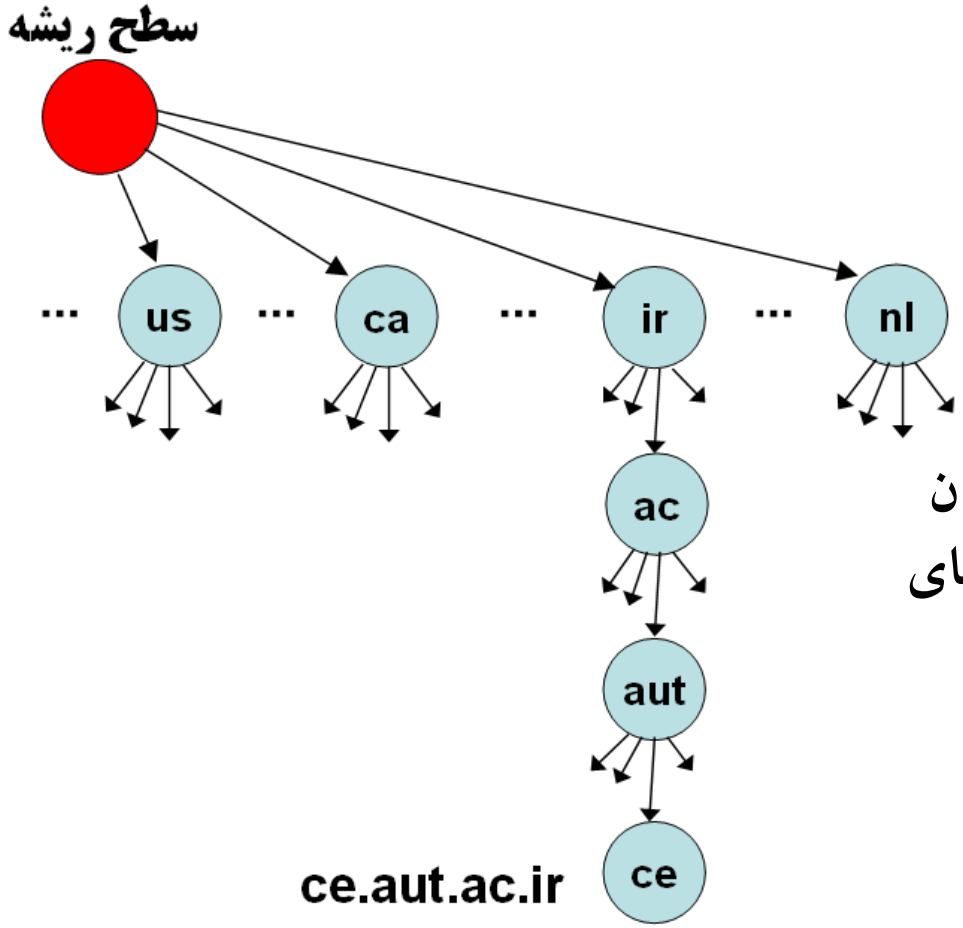
The inverse domain is used to map an IP address to a name

Generic domains



	Description
com	Commercial organizations
edu	Educational institutions
gov	Government institutions
int	International organizations
mil	Military groups
net	Network support centers
org	Nonprofit organizations

Country domains



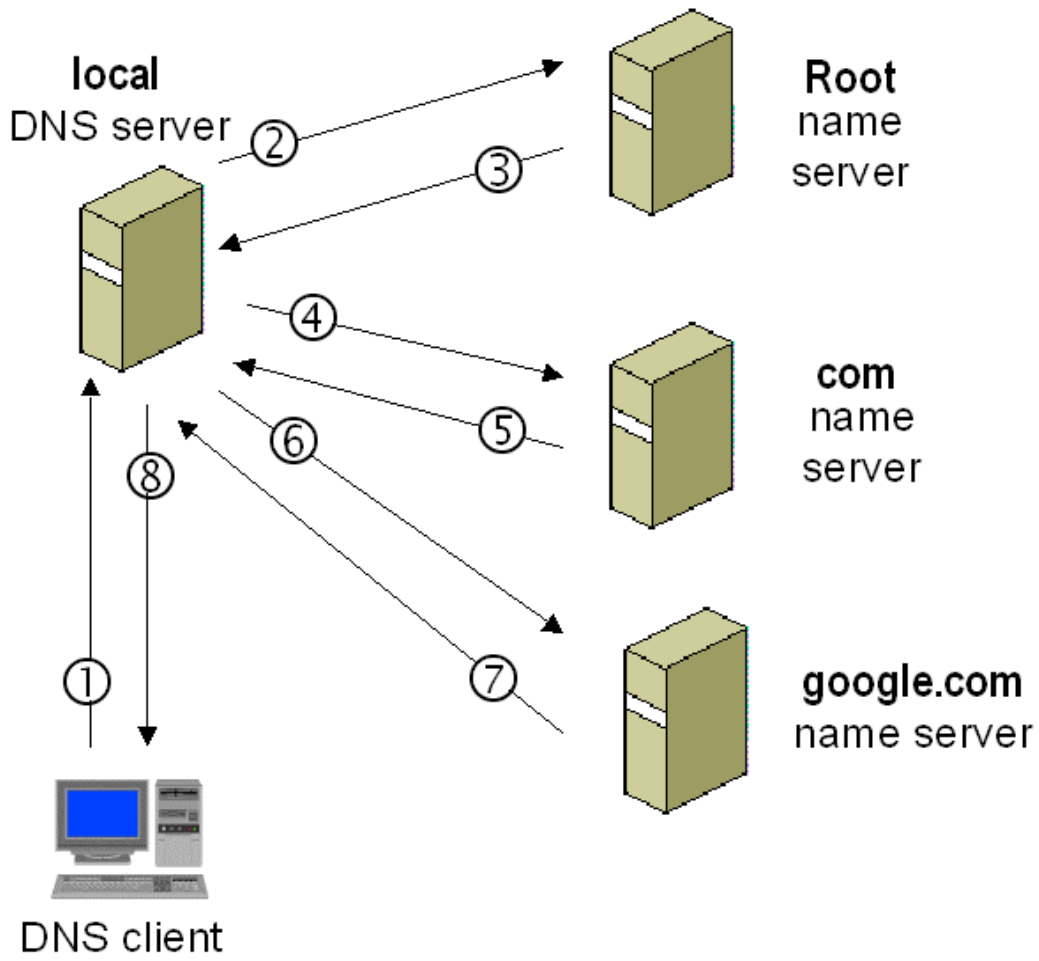
پژوهشگاه دانش‌های بنیادی در تهران مسئولیت کنترل دامنه ir و زیر دامنه های آن را به عهده دارد

http://www.nic.ir/

Local DNS Server

- هر ISP (ISP محلی، شرکت، دانشگاه) یک خدمتگزار نام محلی دارد.
  - که به آن «خدمتگزار نام پیش فرض» نیز گفته می‌شود.
- زمانی که میزبان یک جستجوی DNS داشته باشد، جستجو را به خدمتگزار DNS محلی ارسال می کند.
  - خدمتگزار DNS محلی به صورت یک میانجی، جستجو را به سلسله مراتب می‌فرستد.
  - برای جستجوی نام میزبان های متداول (مثل گوگل) سرعت جستجو به میزان قابل ملاحظه‌ای افزایش می‌یابد.

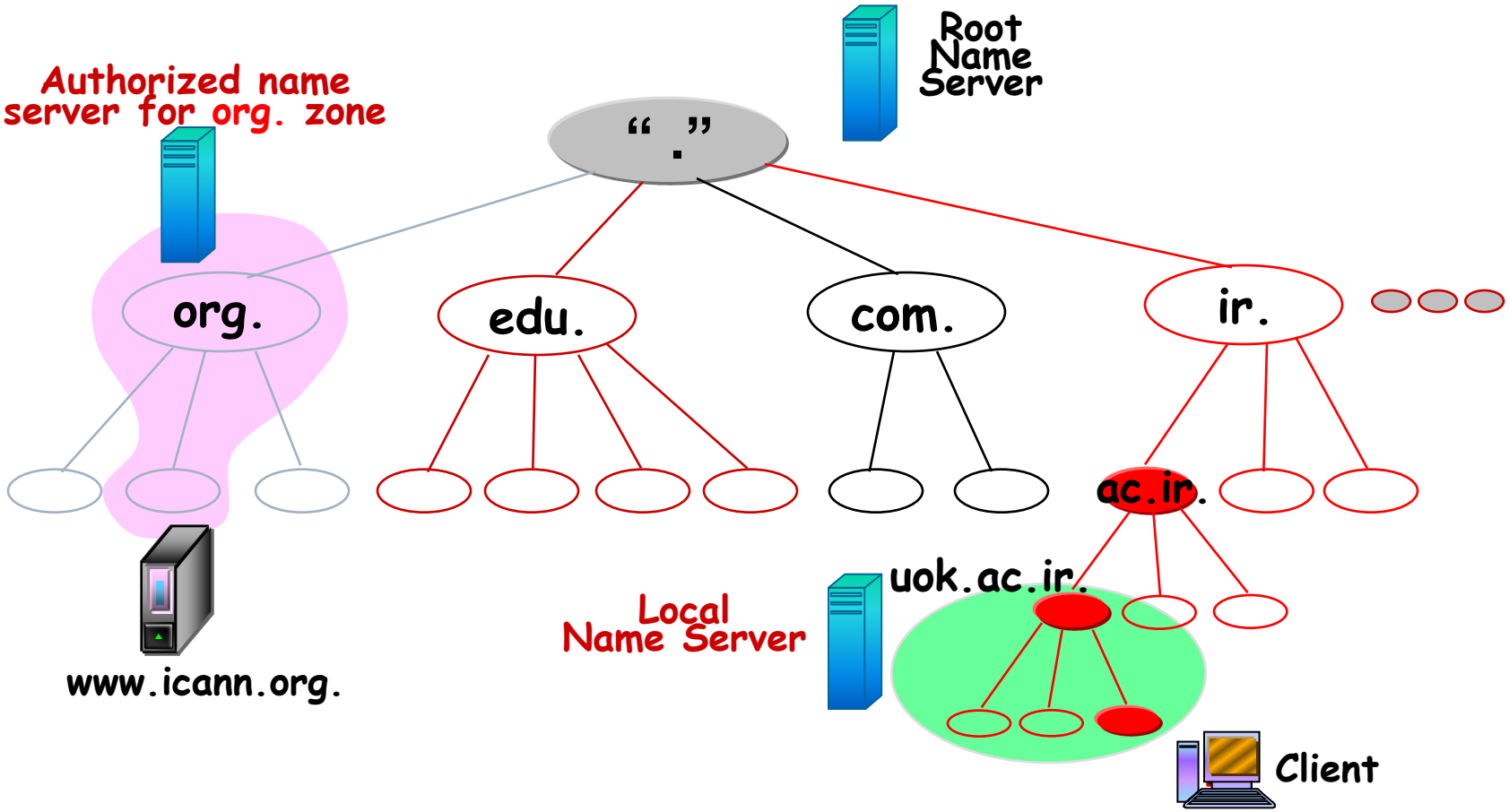
Iterative resolution



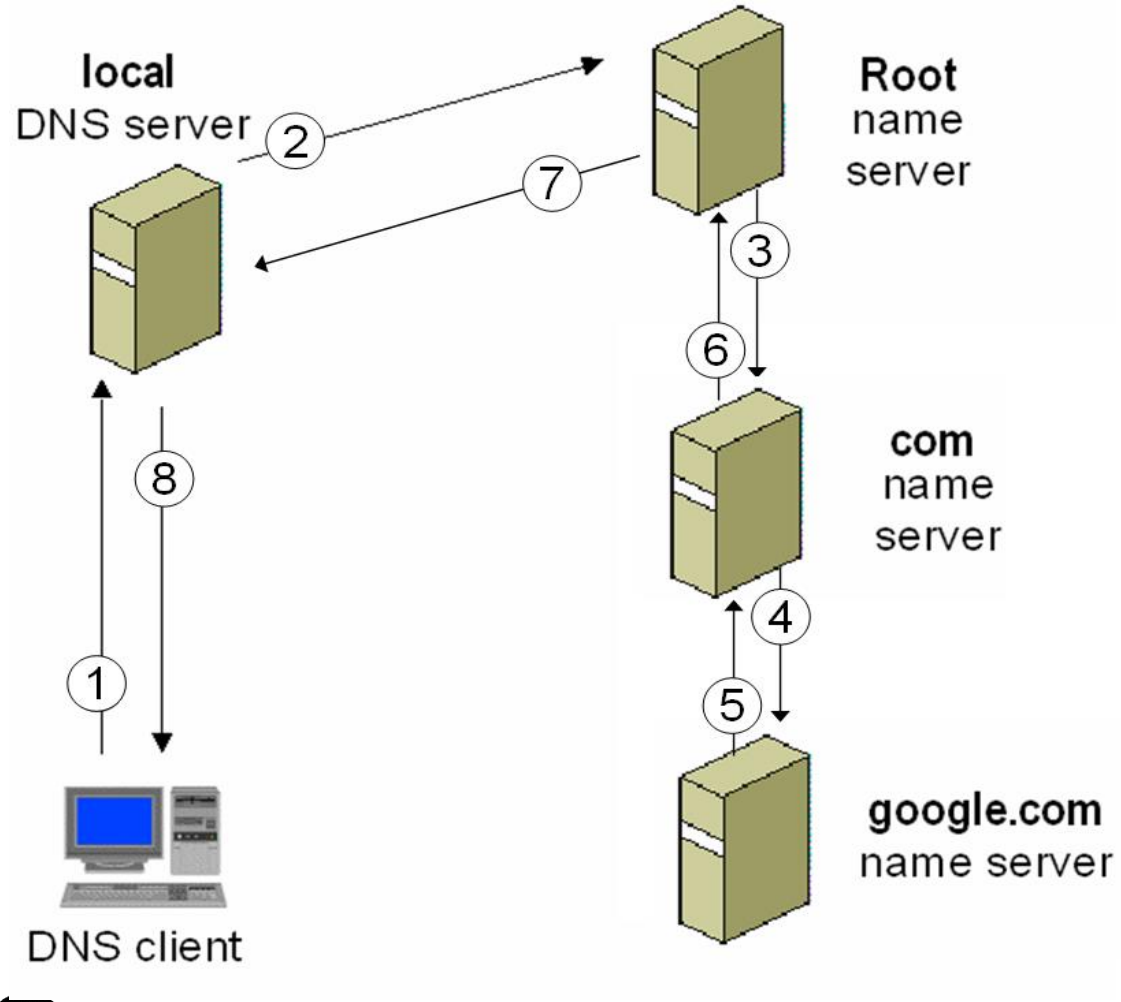
پرس و جوی تکراری

قسمت اعظم تلاش برای تبدیل یک نام برعهده **سرویس دهنده محلی** است

DNS Protocol: Forward Lookup Query

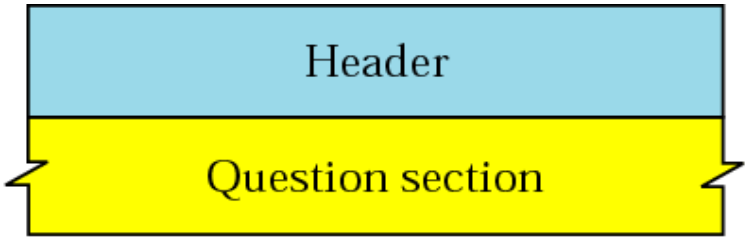


Recursive resolution

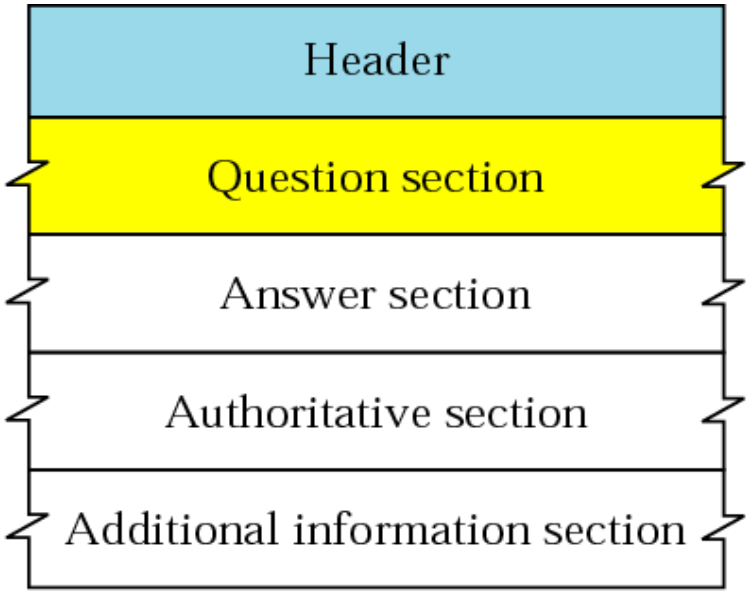


پرس و جوی بازگشتی

Query and response messages



a. Query

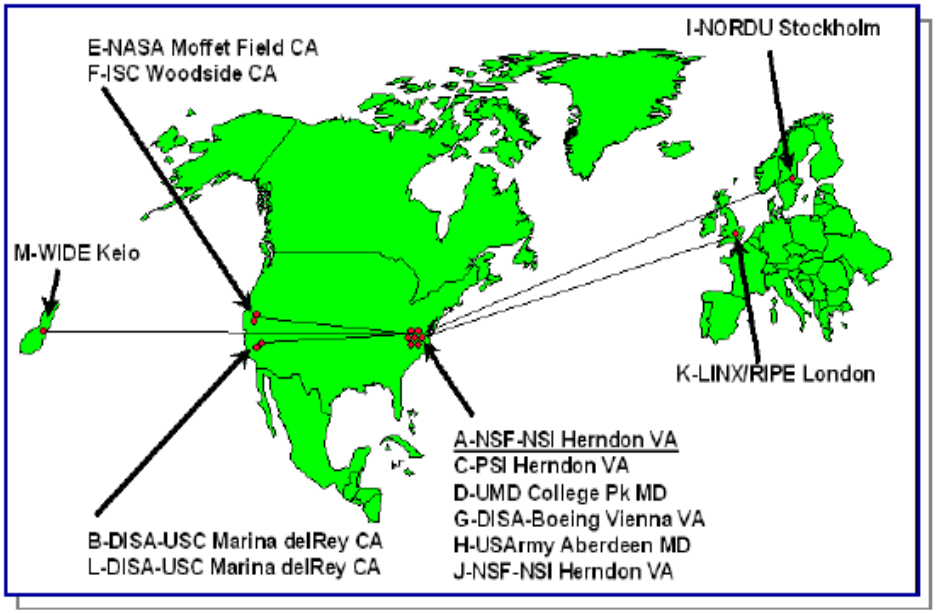


b. Response

DNS can use the services of UDP or TCP, using the well-known port 53.

خلاصه

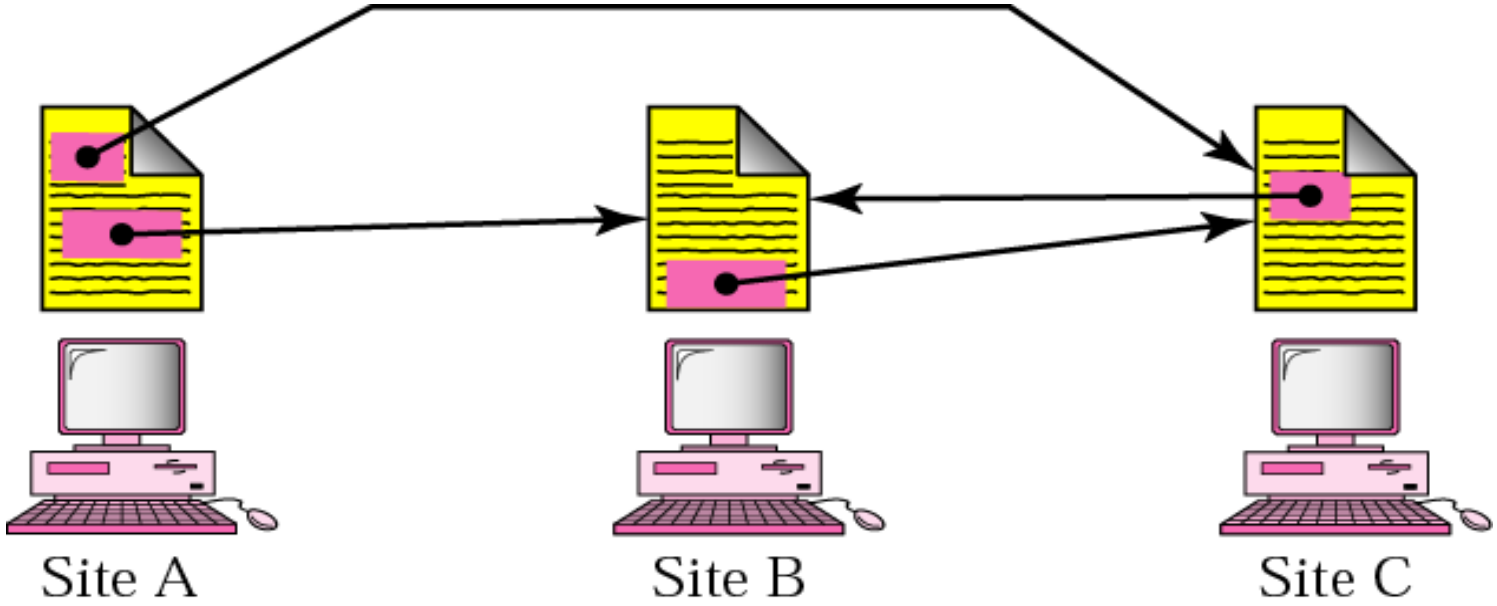
- F روزانه +270,000,000سوال را جواب می‌دهد.
  - دیگرسرورها بار مشابهی دارند
- TLD serverها در هر روز ۵۰۰۰۰۰۰۰۰۰۰ پرسش را جواب می دهند
- واضح است DNS بدون موارد زیر از کار می‌افتد:
  - **سلسله مراتب**
  - **پردازش توزیع شده**
  - **حافظه نهان**
- اگر DNS ناموفق باشد ، خدمات اینترنت از کار می افتند!



HTTP and WWW

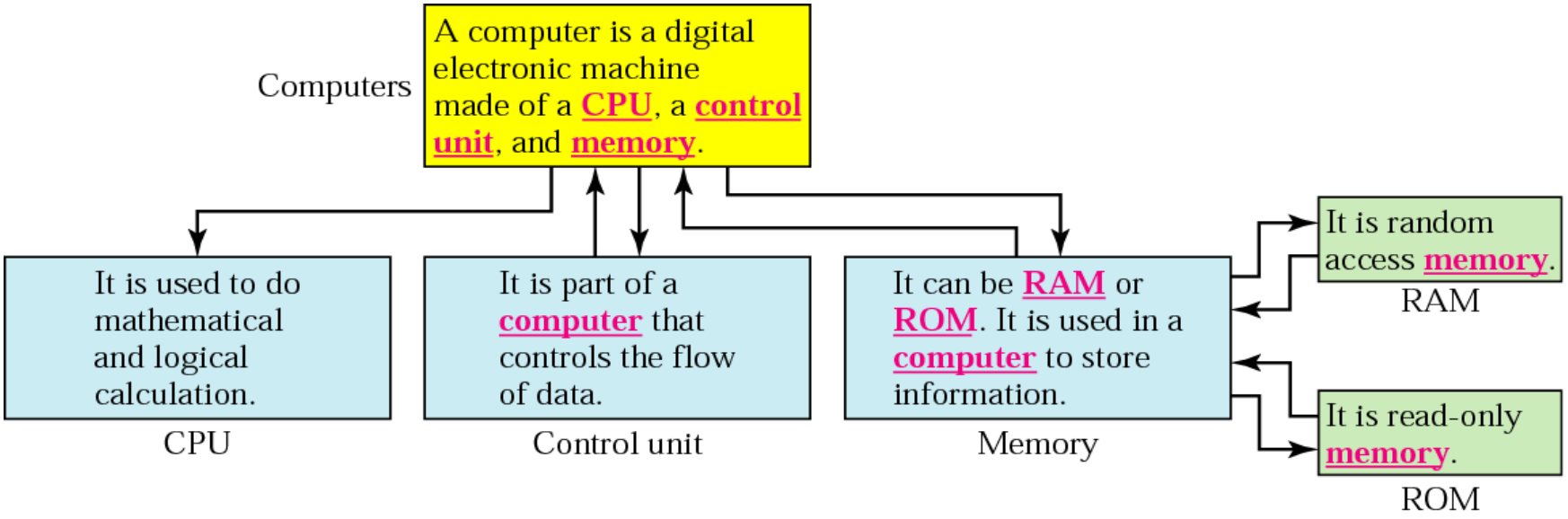
وب جهان گستر و پروتکل انتقال صفحات ابرمتن

World Wide Web

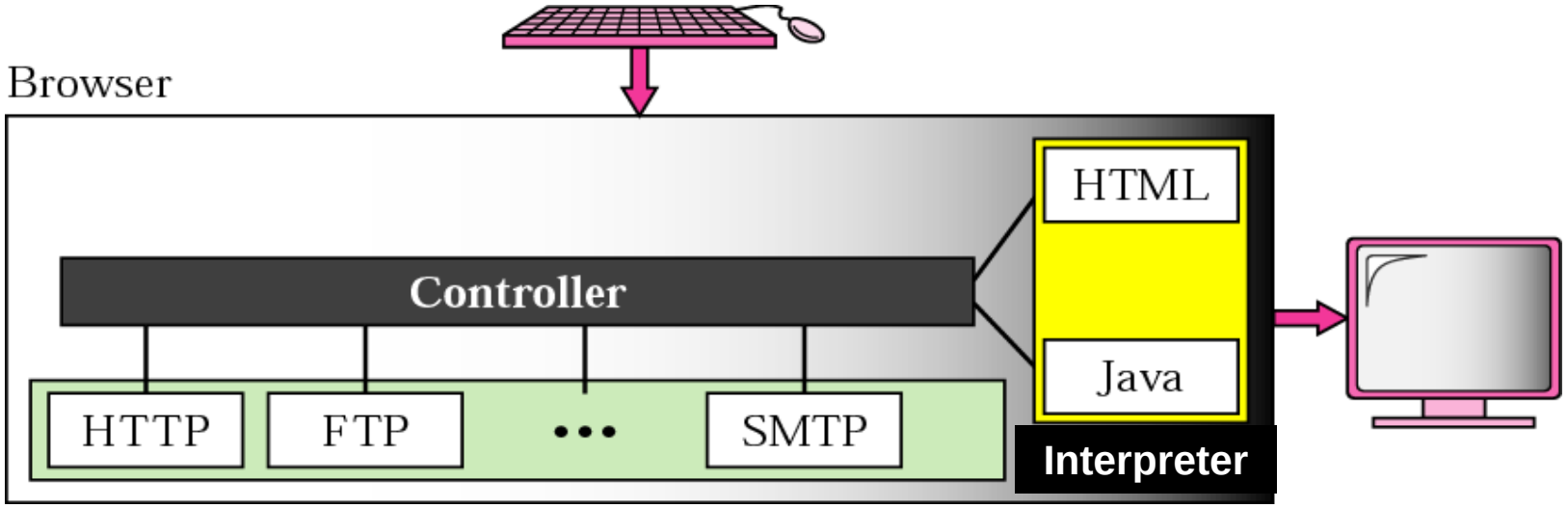


Distributed services

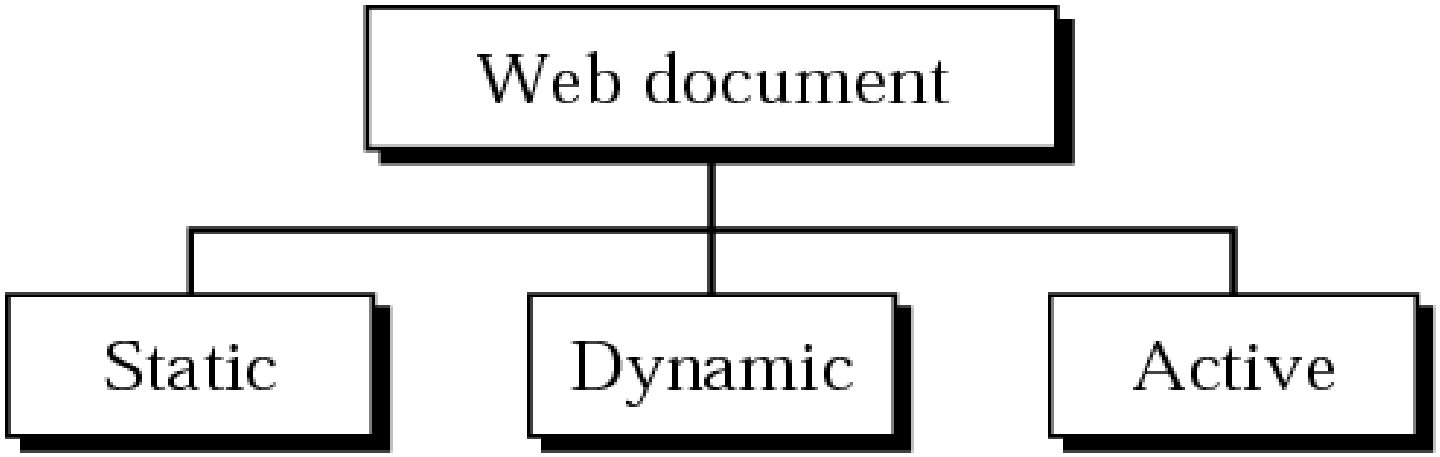
Hypertext


Computers
A computer is a digital electronic machine made of a CPU, a control unit, and memory.
It is used to do mathematical and logical calculation.
CPU
It is part of a computer that controls the flow of data.
Control unit
It can be RAM or ROM. It is used in a computer to store information.
Memory
It is random access memory.
RAM
It is read-only memory.
ROM

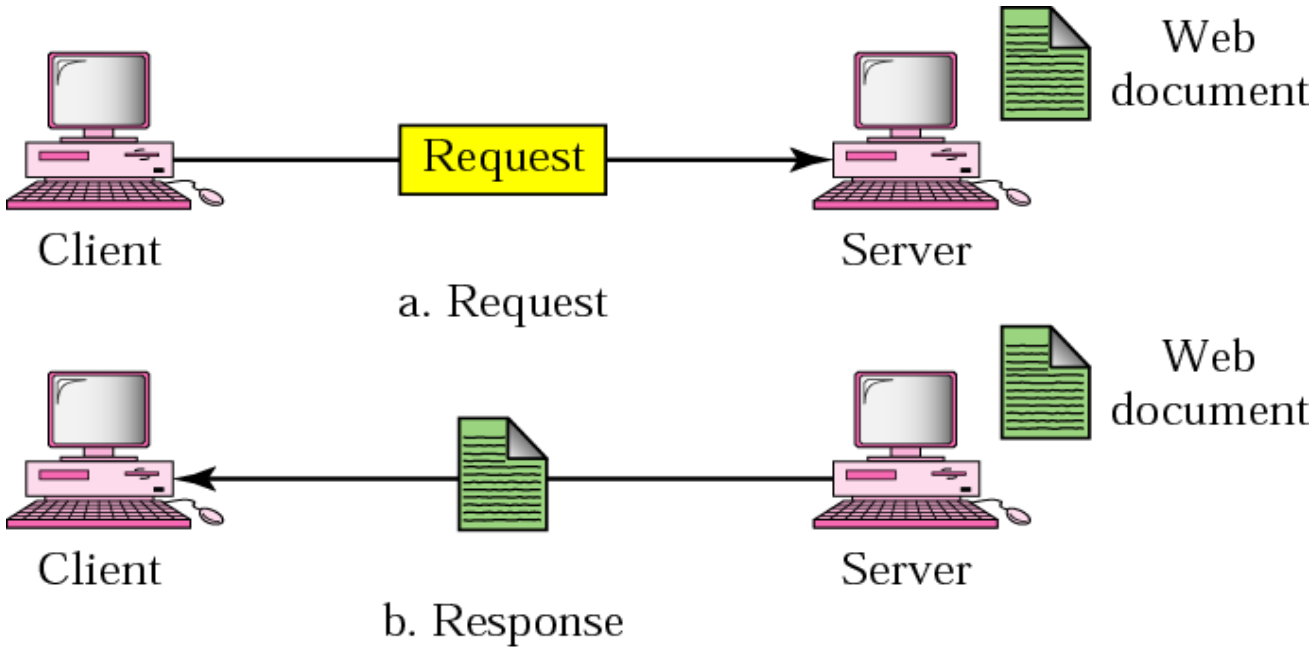
Browser architecture



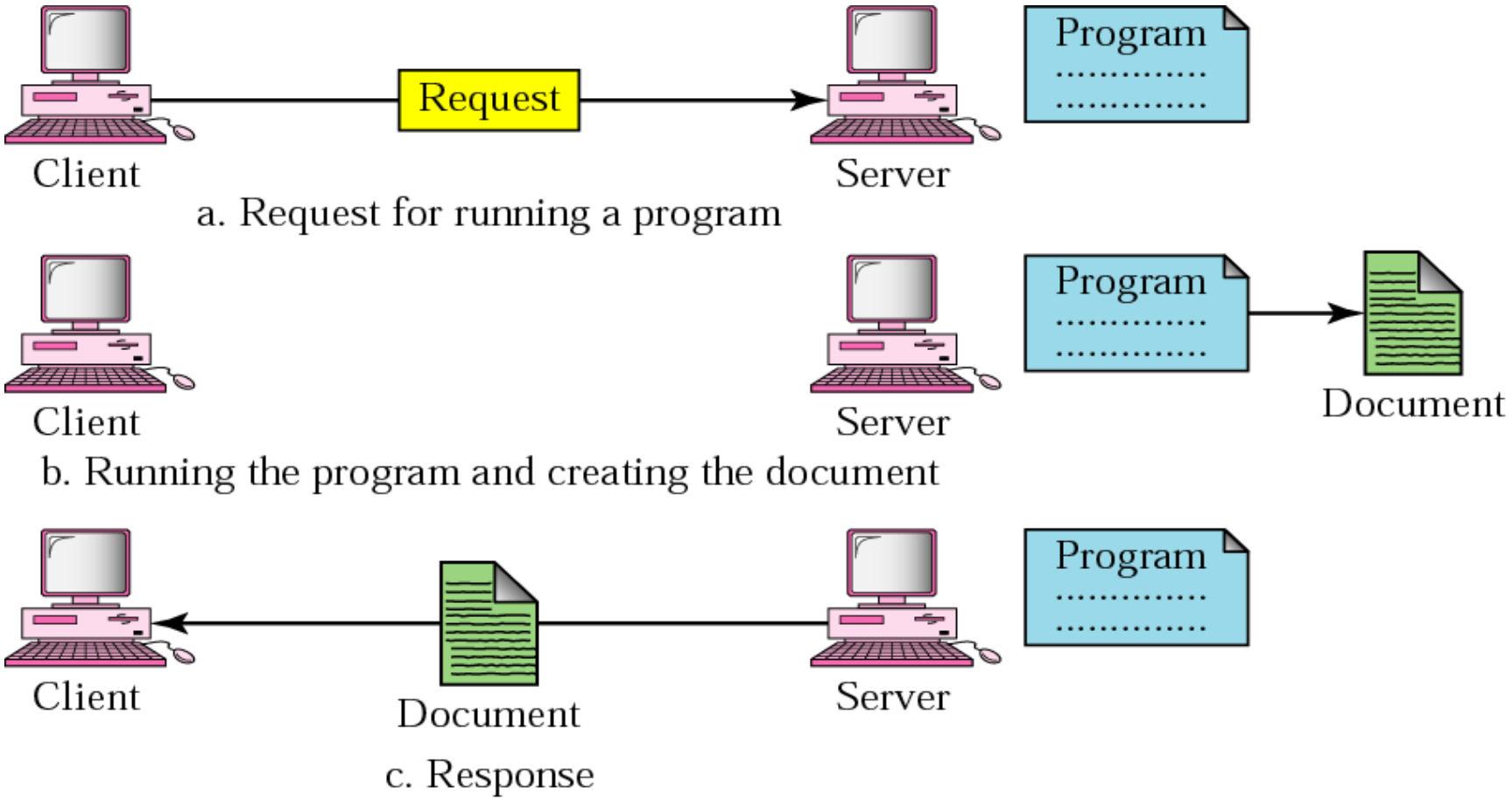
Categories of Web documents



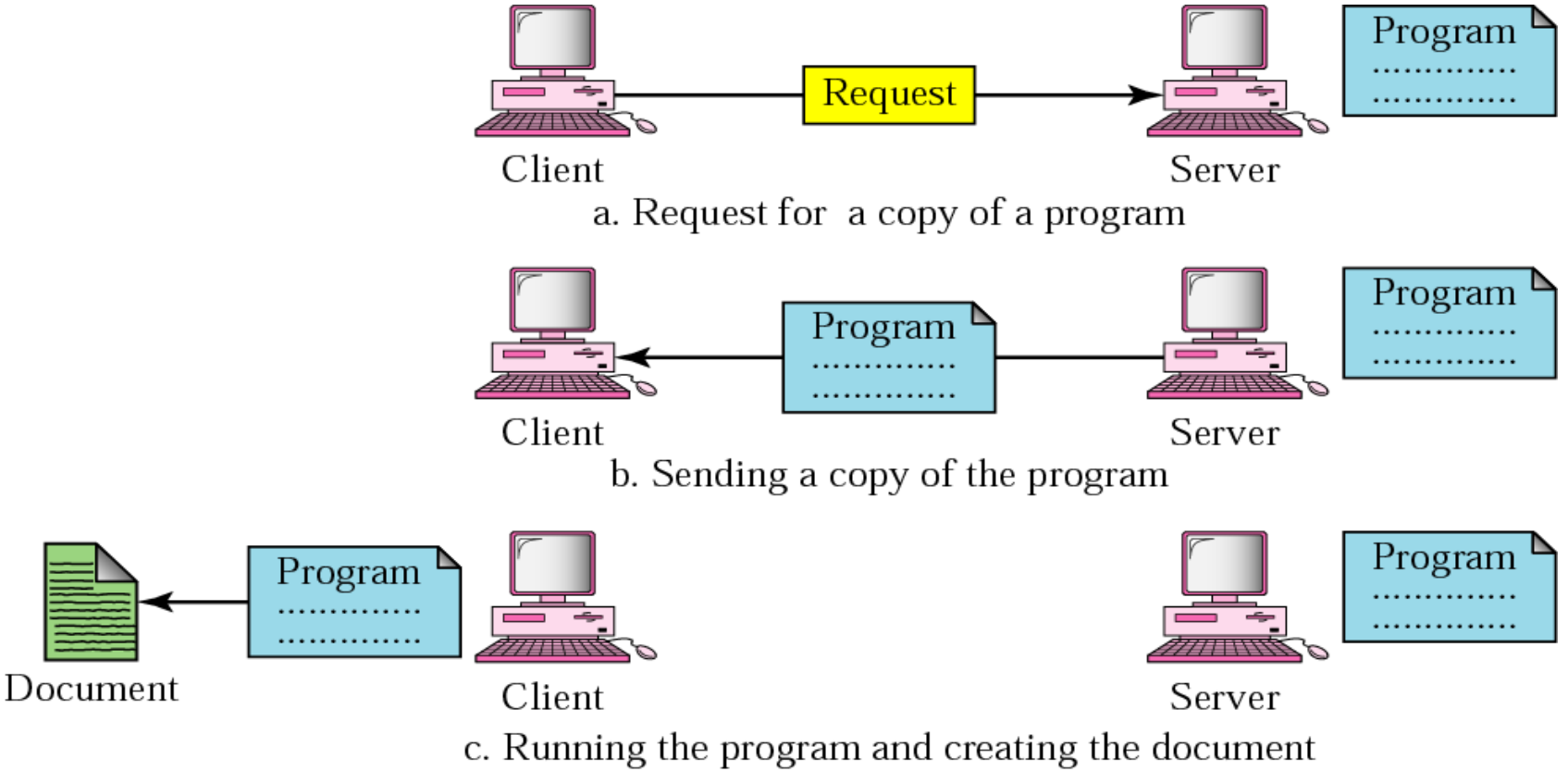
Static document



Dynamic document



Active document



Web and HTTP

- 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
- Example URL:

HTTP Overview

HTTP: hypertext transfer protocol

- Web's app layer protocol
- client/server model
  - *client:* browser that requests, receives, "displays" Web objects
  - *server:* Web server sends objects in response to requests
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068

HTTP uses the services of TCP on well-known port 80.

HTTP Overview (cont.)

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

HTTP Connections

Nonpersistent HTTP

- At most one object is sent over a TCP connection.
- HTTP/1.0 uses nonpersistent HTTP

Persistent HTTP

- Multiple objects can be sent over single TCP connection between client and server.
- HTTP/1.1 uses persistent connections in default mode

Non-persistent HTTP

Suppose user enters URL
`www.uok.ac.ir/computer/home.html`

(contains text, references to 10 jpeg images)

time

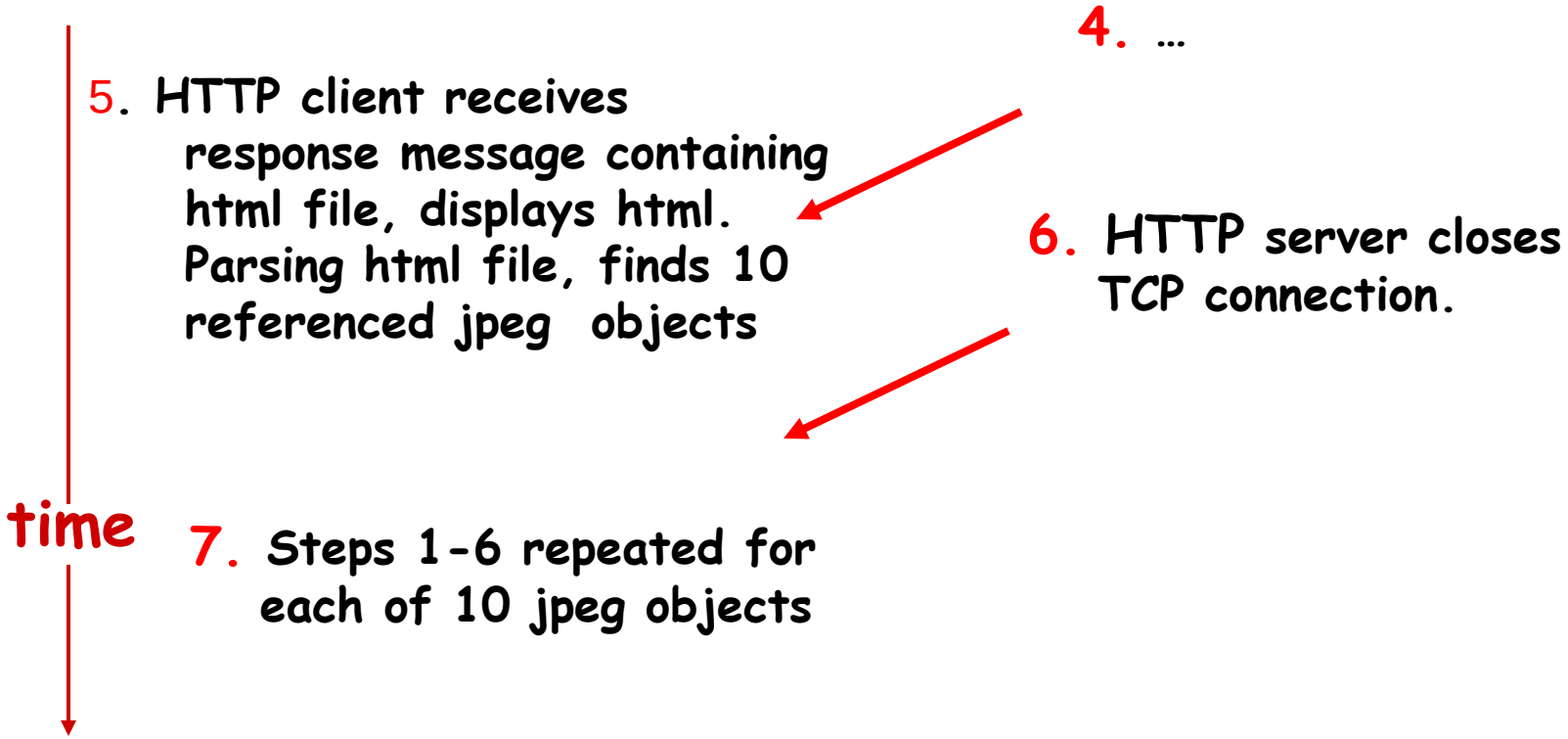
1. HTTP client initiates TCP connection to HTTP server (process) at www.uok.ac.ir on port 80

2. HTTP server at host `www.uok.ac.ir` waiting for TCP connection at port 80. "accepts" connection, notifying client

3. HTTP client sends HTTP *request message* (containing URL) into TCP connection socket. Message indicates that client wants object `Computer/home.html`

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

Non-persistent HTTP (cont.)

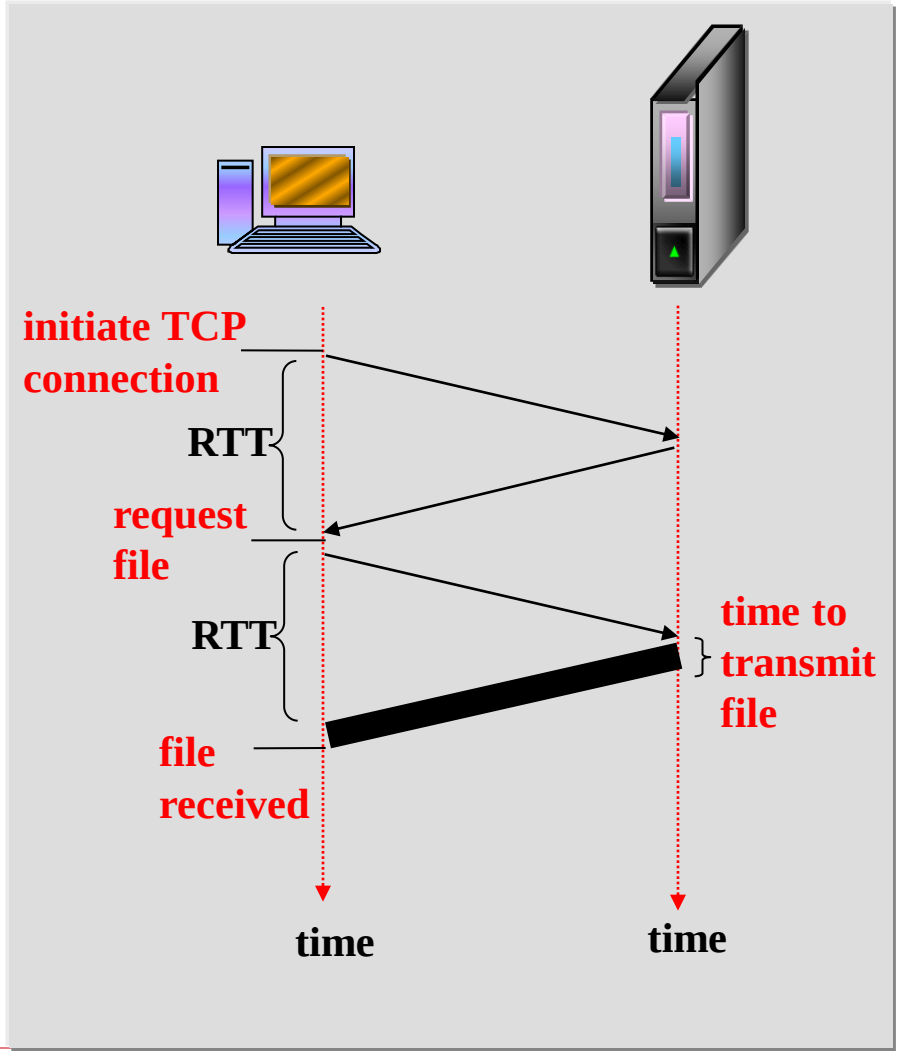


Response Time Modeling

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 = 2RTT+transmit time



Persistent HTTP

Nonpersistent HTTP issues:

- requires 2 RTTs per object
- OS must work and allocate host resources for each TCP connection
- but browsers often open parallel TCP connections to fetch referenced objects

Persistent HTTP

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server are sent over connection

Persistent without pipelining:

- client issues new request only when previous response has been received
- one RTT for each referenced object

Persistent with pipelining:

- default in HTTP/1.1
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

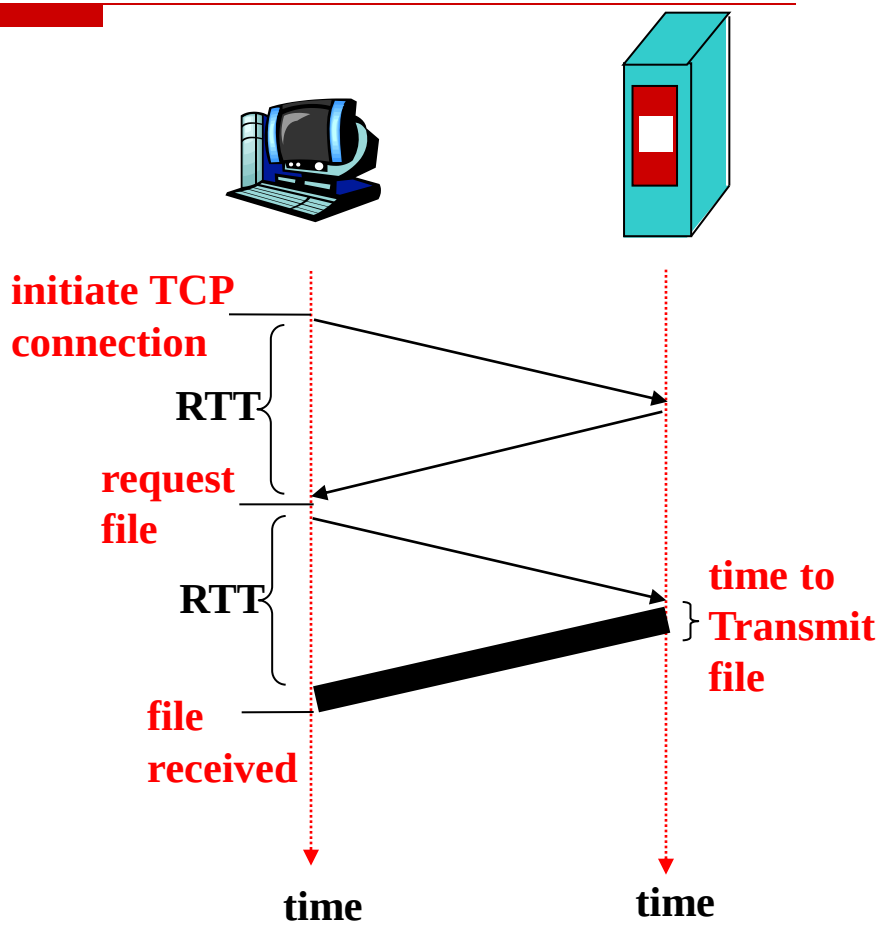
Response time modeling

Definition of RRT: time to send a small packet to travel from client to server and back.

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 = 2RTT+transmit time



Persistent HTTP

Nonpersistent HTTP issues:

- requires 2 RTTs per object
- OS must work and allocate host resources for each TCP connection
- but browsers often open parallel TCP connections to fetch referenced objects

Persistent HTTP

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server are sent over connection

Persistent without pipelining:

- client issues new request only when previous response has been received
- one RTT for each referenced object

Persistent with pipelining:

- default in HTTP/1.1
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

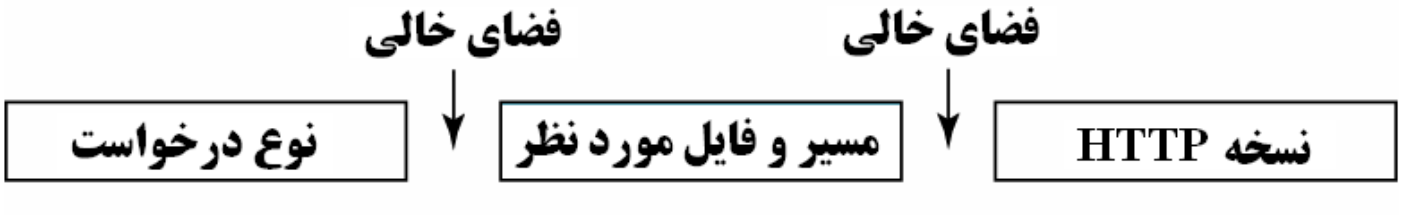
HTTP Modeling

- Assume Web page consists of:
  - *1* base HTML page (of size *O* bits)
  - *M* images (each of size *O* bits)
- Non-persistent HTTP:
  - *M+1* TCP connections in series
  - *Response time = (M+1)O/R + (M+1)2RTT + sum of idle times*
- Persistent HTTP:
  - *2 RTT* to request and receive base HTML file
  - *1 RTT* to request and receive M images
  - *Response time = (M+1)O/R + 3RTT + sum of idle times*
- Non-persistent HTTP with X parallel connections
  - Suppose M/X integer.
  - 1 TCP connection for base file
  - M/X sets of parallel connections for images.
  - *Response time = (M+1)O/R + (M/X + 1)2RTT + sum of idle times*

HTTP request and response format

پیغام درخواست

پیغام پاسخ

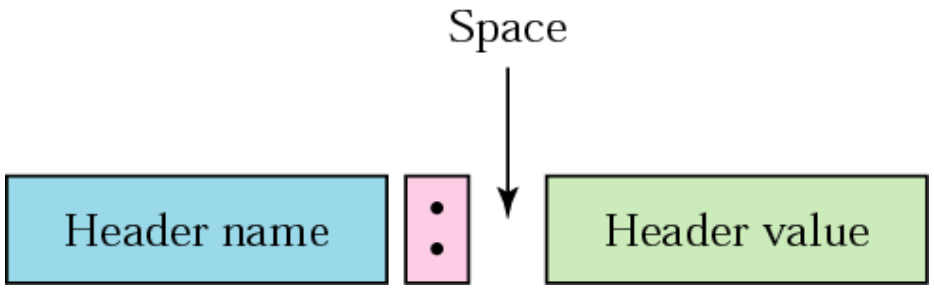


Request line

خط درخواست

Status line

خط وضعیت



هر سطر سرآیند به این فرمت است

نام فرمان	توضیـــح
GET	تقاضا برای دریافت یک صفحه وب از سرویس‌دهنده
HEAD	تقاضا برای دریافت سرآیند یک صفحه وب
PUT	تقاضا برای ذخیره کردن یک صفحه وب روی یک سرویس‌دهنده
POST	تقاضا برای ضمیمه کردن اطلاعاتی به یک منبع (مثل فایل یا صفحه وب)
DELETE	تقاضا برای حذف یک صفحه وب

انواع دیگر درخواست:

OPTIONS, PATCH, COPY, MOVE, LINK, UNLINK, TRACE,

HTTP Request Message - example

HTTP request message: ASCII (human-readable format)

request line (GET, POST, HEAD commands)

header lines

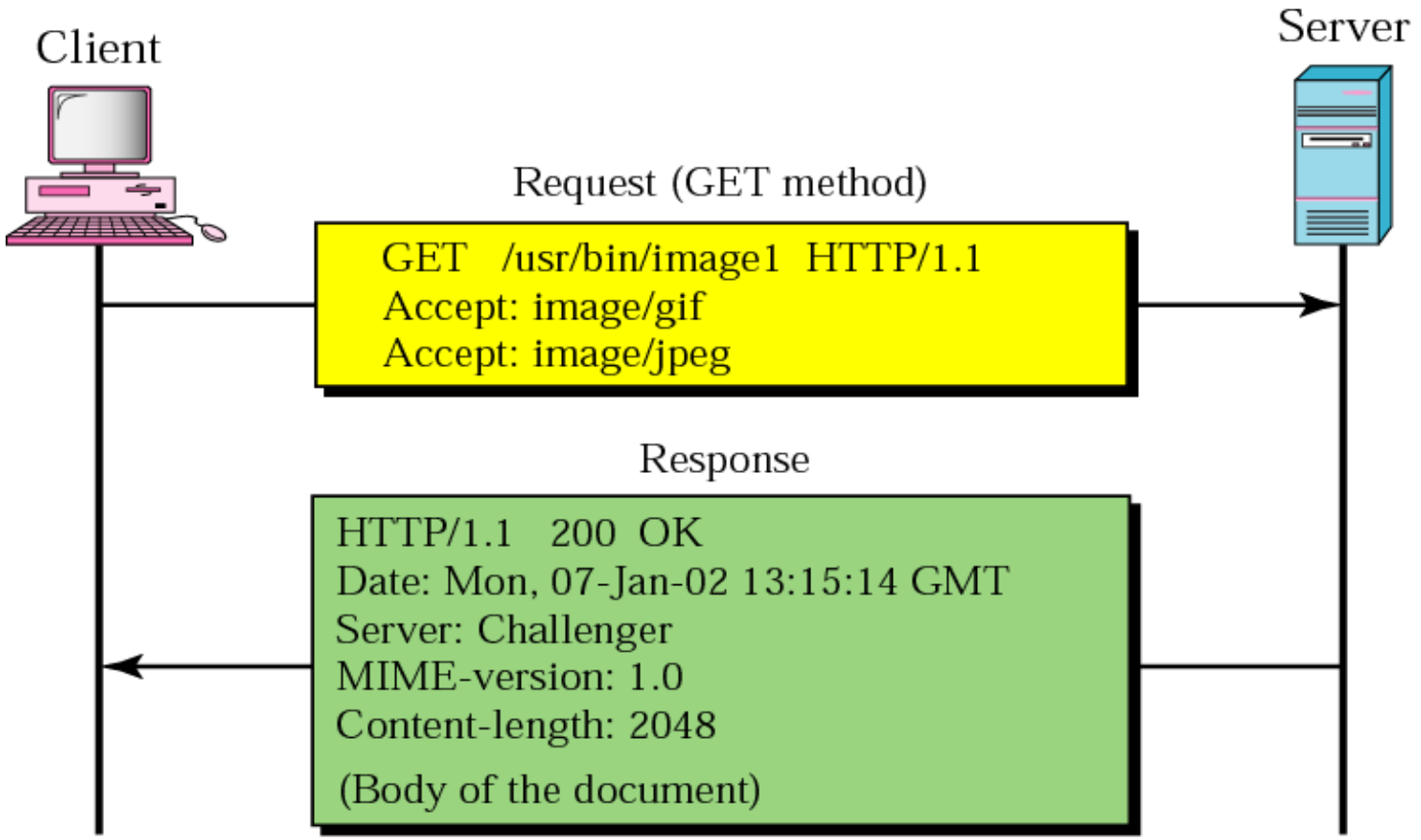
Carriage return, line feed indicates end of message

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language:fr
If-modified-since:Sat, 2 Nov 2002 13:45:12
(carriage return, line feed)
```

Example 1

This example retrieves a document. We use the GET method to retrieve an image with the path /usr/bin/image1. The request line shows the method (GET), the URL, and the HTTP version (1.1). The header has two lines that show that the client can accept images in GIF and JPEG format. The request does not have a body. The response message contains the status line and four lines of header. The header lines define the date, server, MIME version, and length of the document. The body of the document follows the header (see next slide).

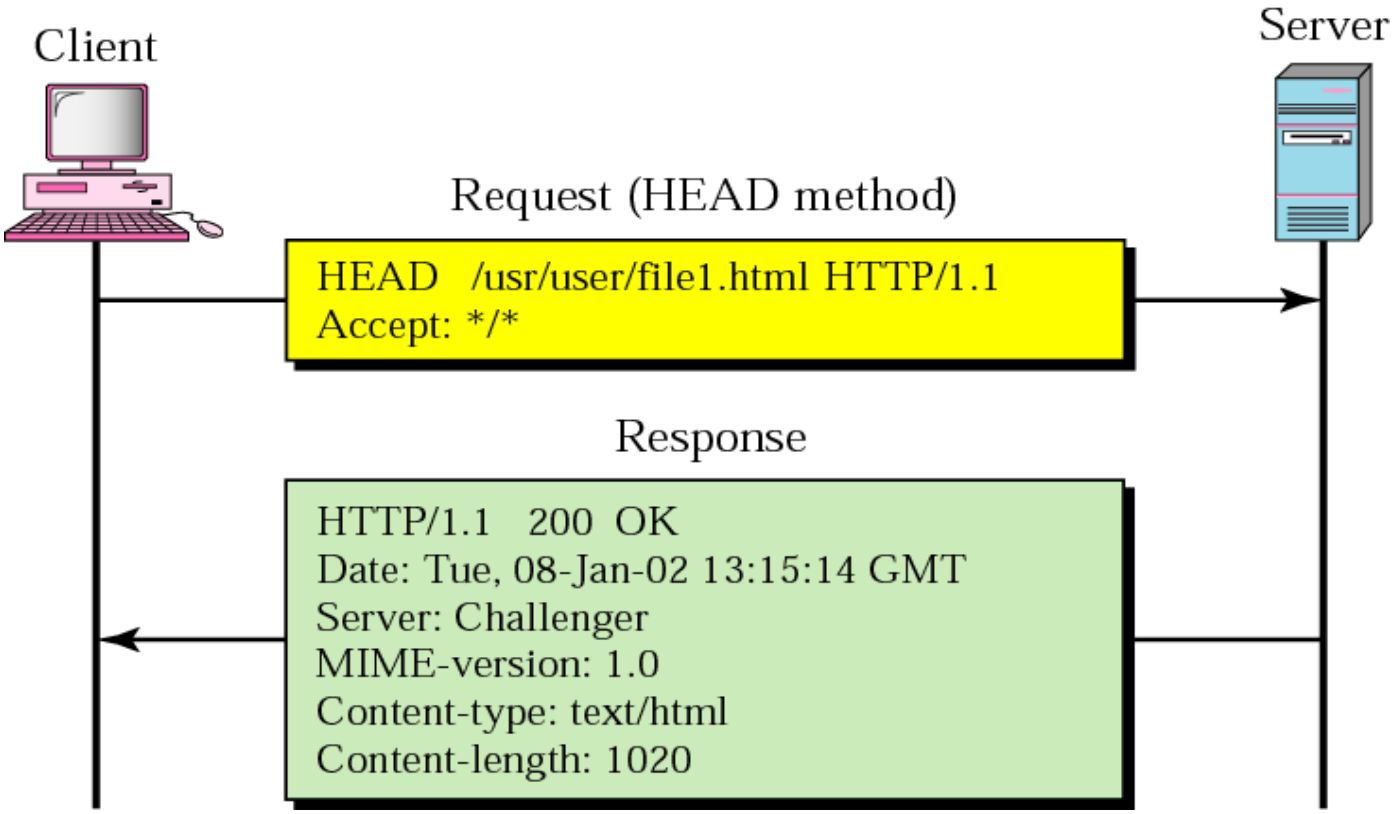
Example 1


Client
Server
Request (GET method)
GET /usr/bin/image1 HTTP/1.1
Accept: image/gif
Accept: image/jpeg
Response
HTTP/1.1 200 OK
Date: Mon, 07-Jan-02 13:15:14 GMT
Server: Challenger
MIME-version: 1.0
Content-length: 2048
(Body of the document)

Example 2

This example retrieves information about a document. We use the HEAD method to retrieve information about an HTML document. The request line shows the method (HEAD), URL, and HTTP version (1.1). The header is one line showing that the client can accept the document in any format (wild card). The request does not have a body. The response message contains the status line and five lines of header. The header lines define the date, server, MIME version, type of document, and length of the document (see next slide). Note that the response message does not contain a body.

Example 2


Client
Server
Request (HEAD method)
HEAD /usr/user/file1.html HTTP/1.1
Accept: */*
Response
HTTP/1.1 200 OK
Date: Tue, 08-Jan-02 13:15:14 GMT
Server: Challenger
MIME-version: 1.0
Content-type: text/html
Content-length: 1020

فرض کنید کاربر، آدرس زیر را در مرورگر خود وارد می‌کند:

http://www.w3.org/hyper/www/project.html

مرورگر با تحلیل آدرس متوجه میشود که باید تقاضای فایلی را طبق پروتکل HTTPبه سمت سرویس دهنده بفرستد. مراحلی که اتفاق می افتند به شرح زیر خواهند بود:

۱) مرورگر آدرس را تحلیل کرده و قسمتهای پروتکل، آدرس نام حوزه، شاخه و نام فایل را از آن استخراج میکند.
۲) مرورگر یک اتصال UDP با پورت ۵۳ سرویس دهنده DNS برقرار نموده و تقاضای ترجمه آدرس نام حوزه را به آن ارسال می نماید تا آدرس IP ماشین سرویس دهنده بدست آید. در این مثال مرورگر تقاضای ترجمه نام www.w3.org را به DNS ارسال میکند.

۳) DNS در پاسخ، آدرس IP معادل با نام حوزه را برمیگرداند. فرض کنید در این مثال DNS آدرس IP را 128.30.52.31برگردانده است.
٤) مرورگر یک ارتباط TCP با آدرس 128.30.52.31و پورت۸۰ برقرار میکند.
٥) پس از برقراری ارتباط، یک پیغام درخواست به صورت زیر به سمت سرویس دهنده ارسال میشود:

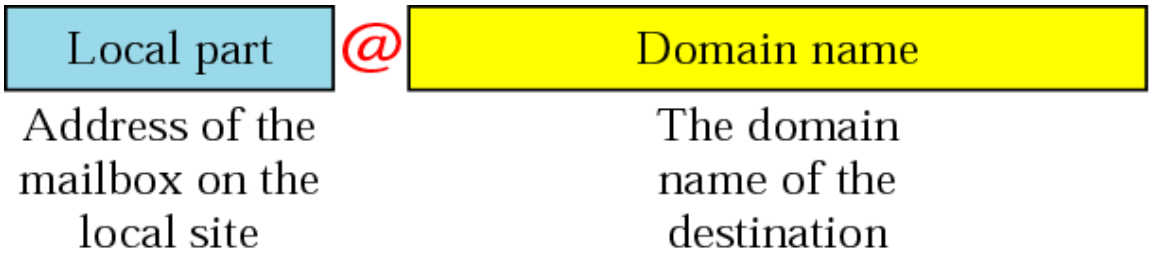
“GET /hyper/www/project.html http/1.1”

٦) سرویس دهنده این رشته را دریافت و پس از پردازش آن، فایل project.htmlرا از شاخه /hyper/www/ استخراج کرده و برای مرورگر ارسال می نماید.
۷) مرورگر فایل را دریافت کرده و پس از خاتمه دریافت ارتباط TCPرا قطع میکند.
۸) مرورگر فایل ابرمتنی را تفسیر کرده و آنرا روی خروجی نمایش میدهد.
۹) اگر فایل ابرمتنی در جایی دارای صدا یا تصویر باشد به ازای تک تک آنها مراحل ۱ تا ۸ را تکرار نموده و آنها را بترتیب دریافت می کند (با فرض persistent http)

SMTP and POP3

پروتکلهای انتقال و دریافت ایمیل

Email address



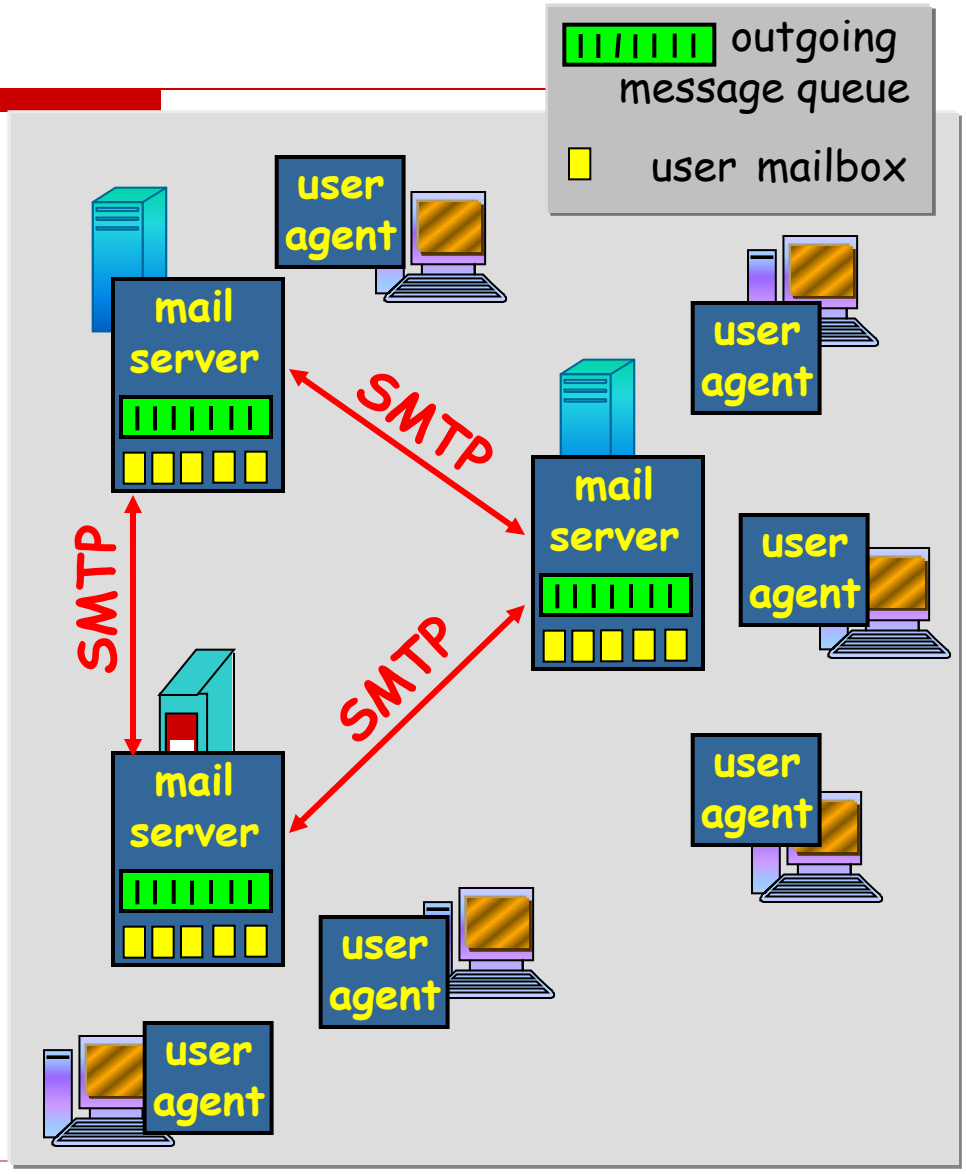
Electronic Mail

Three major components:

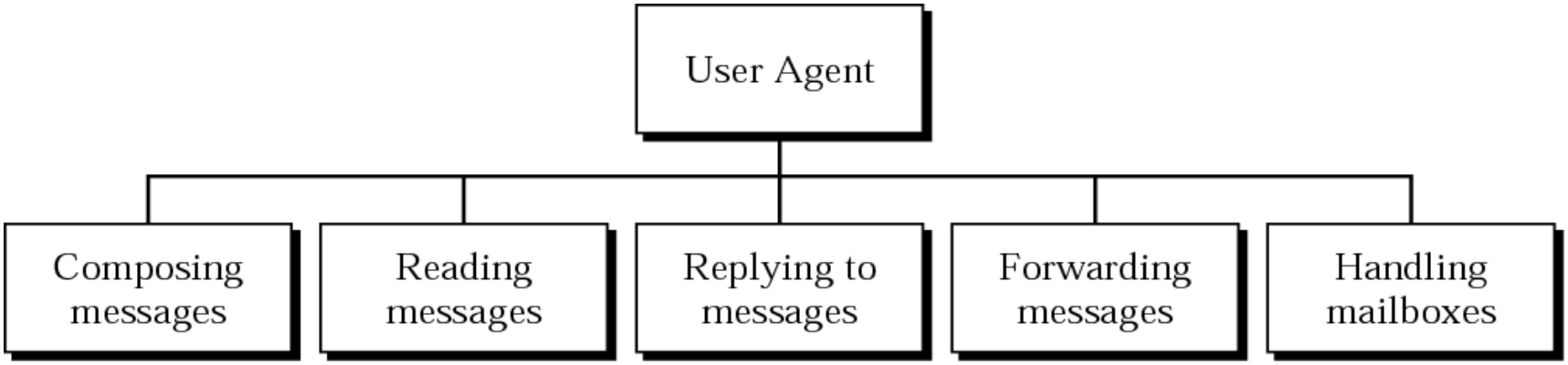
- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent

- Sometimes is called:
  "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Netscape Messenger
- outgoing, incoming messages stored on server



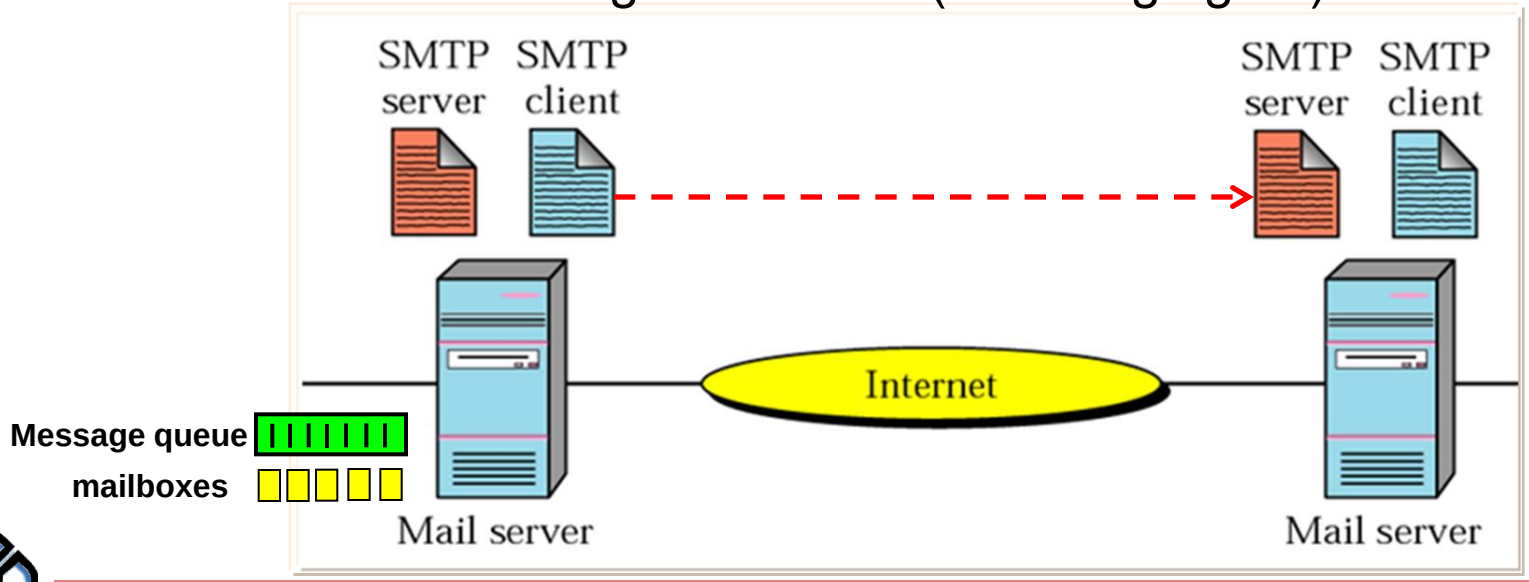
User agent


User Agent
Composing messages
Reading messages
Replying to messages
Forwarding messages
Handling mailboxes

Electronic Mail: Mail Servers

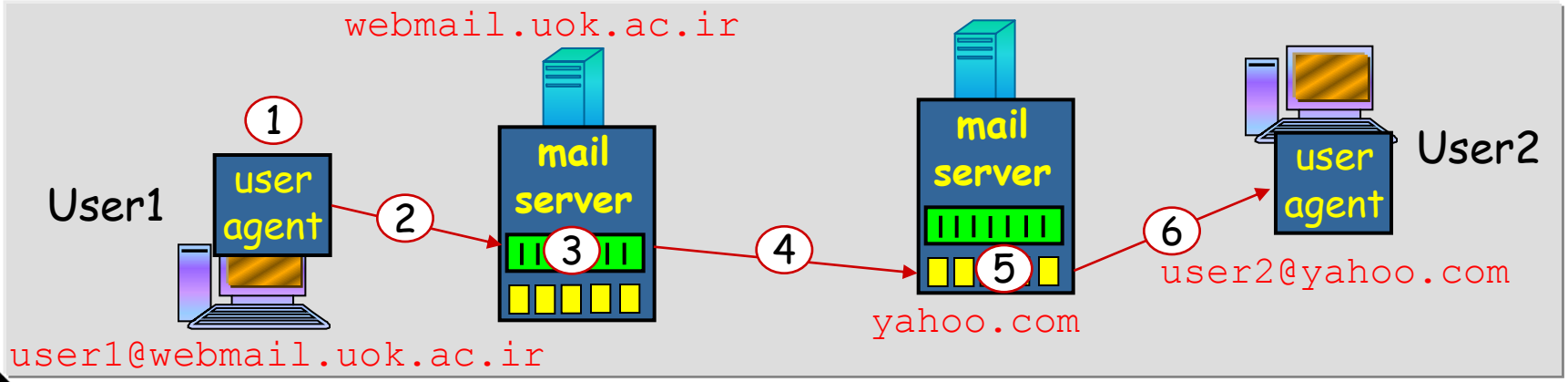
Mail Servers

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - "client": sending mail server (sending agent)
  - "server": receiving mail server (receiving agent)



Example: User1 sends message to User2

1) User1 (sender) uses UA to compose message to `user2@yahoo.com` .
2) User1's UA sends message to his mail server; message placed in message queue.
3) Client side of SMTP opens TCP connection with User2's mail server.
4) SMTP client sends User1's message over the TCP connection.
5) User2's mail server places the message in User2's mailbox.
6) User2 invokes his/her user agent to read message.



SMTP commands

- **HELO**: برنامه سرویس گیرنده را معرفی می‌کند.
- **MAIL FROM**: فرستنده پیام را مشخص می‌کند.
- **RCPT TO**: گیرنده را مشخص می‌کند.
- **DATA**: بدنه نامه را مشخص می‌کند.
- **QUIT**: ارتباط را قطع می نماید.
- **HELP**: در رابطه با دستورات توضیحات لازم را ارائه می‌نماید.

C:\> telnet www.uok.ac.ir 25

Connecting to www.uok.ac.ir ...

========== برقراري اتصال==========

220 PARSDATA Mail Server (IMail 8.00 2586-5) NT-ESMTP Server X1

HELO PARSDATA

250 hello PARSDATA Mail Server

========== پوشش نامه ==========

MAILFROM: Abdollahpour@uok.ac.ir

250 ok

RCPT TO: Abdollahpour@uok.ac.ir

250 ok deliver to alternate

========== سرآیند و بدنه نامه ==========

DATA

354 ok, send it; end with <CRLF>.<CRLF>

FROM: Abdollahpour

TO: myself

Hi this is a sample e-mail to show SMTP in action.

.

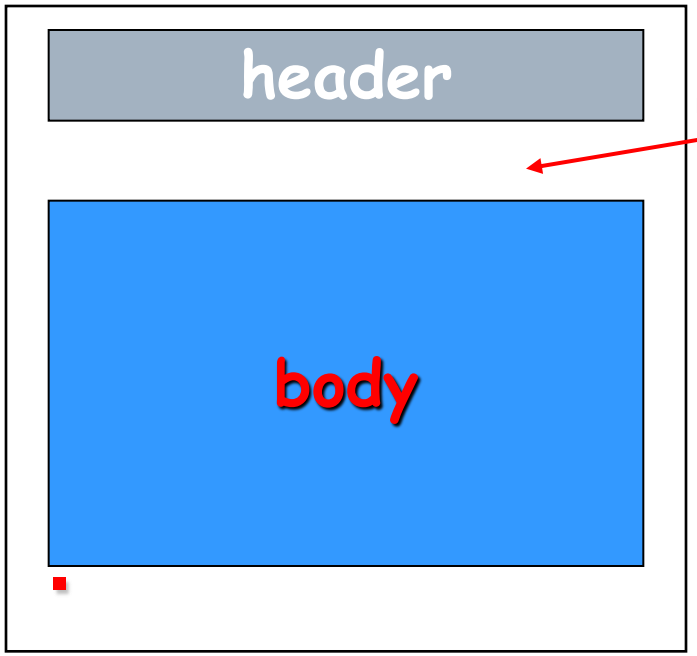
========== خاتمه اتصال==========

250 Message queued

QUIT

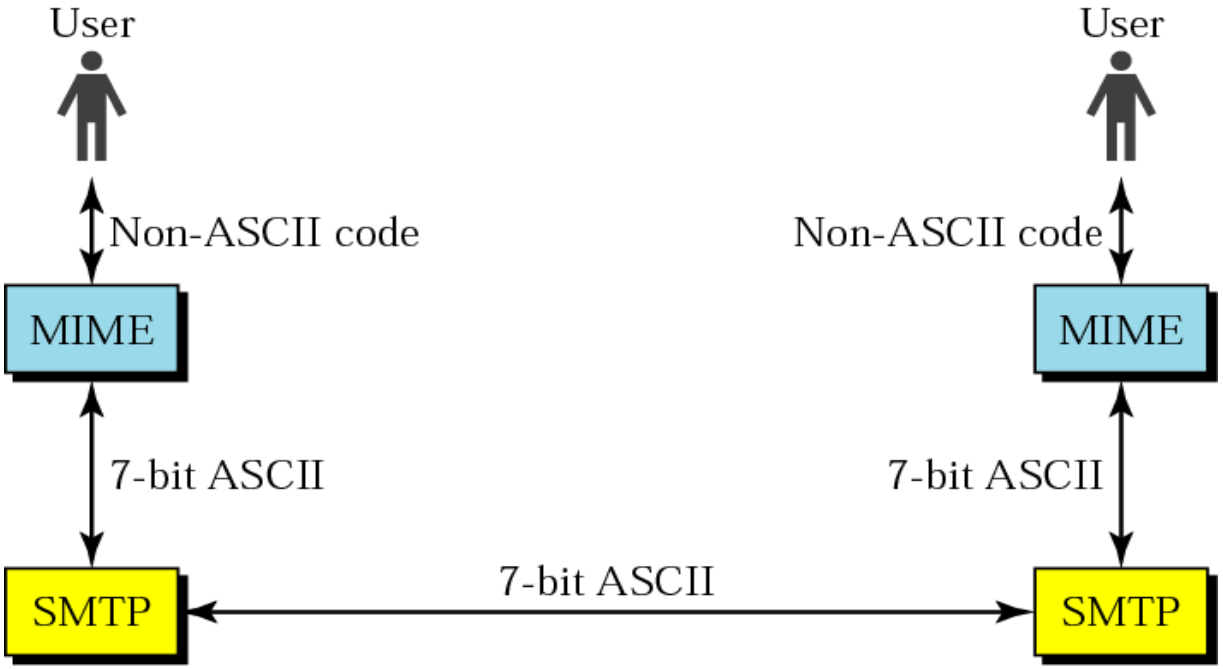
221 Goodbye

Connection to host lost



ارسال ایمیل به صورت مستقیم با دستورات SMTP

MIME



MIME یک پروتکل تکمیلی است که اجازه می دهد کاراکترهای غیرASCII نیز از طریقSMTP منتقل شوند. دقت کنید کهMIME یک پروتکل جایگزین برایSMTP نیست و قادر به ارسال نامه نمی باشد بلکه توسعه ای برای پروتکلSMTP محسوب می شود. میتوان MIME را به صورت برنامه ای تصور نمود که داده غیرASCII را به دادهASCII و بالعکس ترجمه میکند

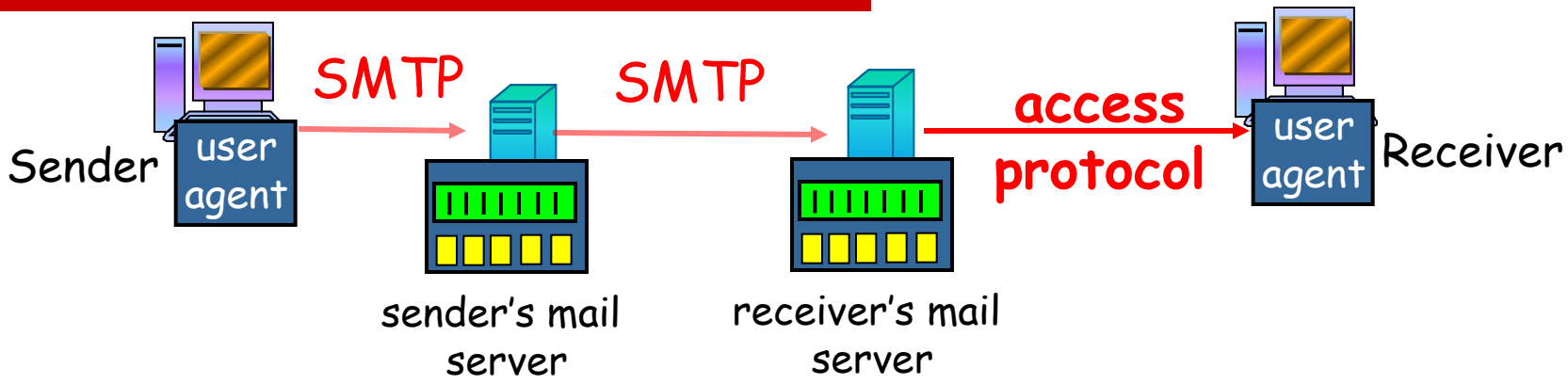
MIME header

```
From: User1@mail.uok.ac.ir
To: User2@yahoo.com
Subject: Picture of iust
cc: User3@tu.ac.ir
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data

.
```

Examples of other Content-Types:

- **Image/gif**
- **Text/plain**
- **Text/html**
- **Audio/basic**
- **Video/mpeg**
- **…**

Mail Access Protocols



- SMTP: delivery/storage to receiver's server (PUSH)
- Mail access protocol: retrieval from server (PULL)
  - POP: Post Office Protocol [RFC 1939]
    - authorization (agent <-->server) and download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - more features (more complex)
    - manipulation of stored messages on server
  - HTTP: Hotmail , Yahoo! Mail, etc.

POP3 Protocol

authorization phase

- client commands:
  - **`user:`** declare username
  - **`pass:`** password
- server responses
  - **`+OK`**
  - **`-ERR`**

transaction phase, client:

- **`list:`** list message numbers
- **`retr:`** retrieve message by number
- **`dele:`** delete
- **`quit`**

```
S: +OK POP3 server ready
C: user USER1
S: +OK
C: pass zxcdvf
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

FTP

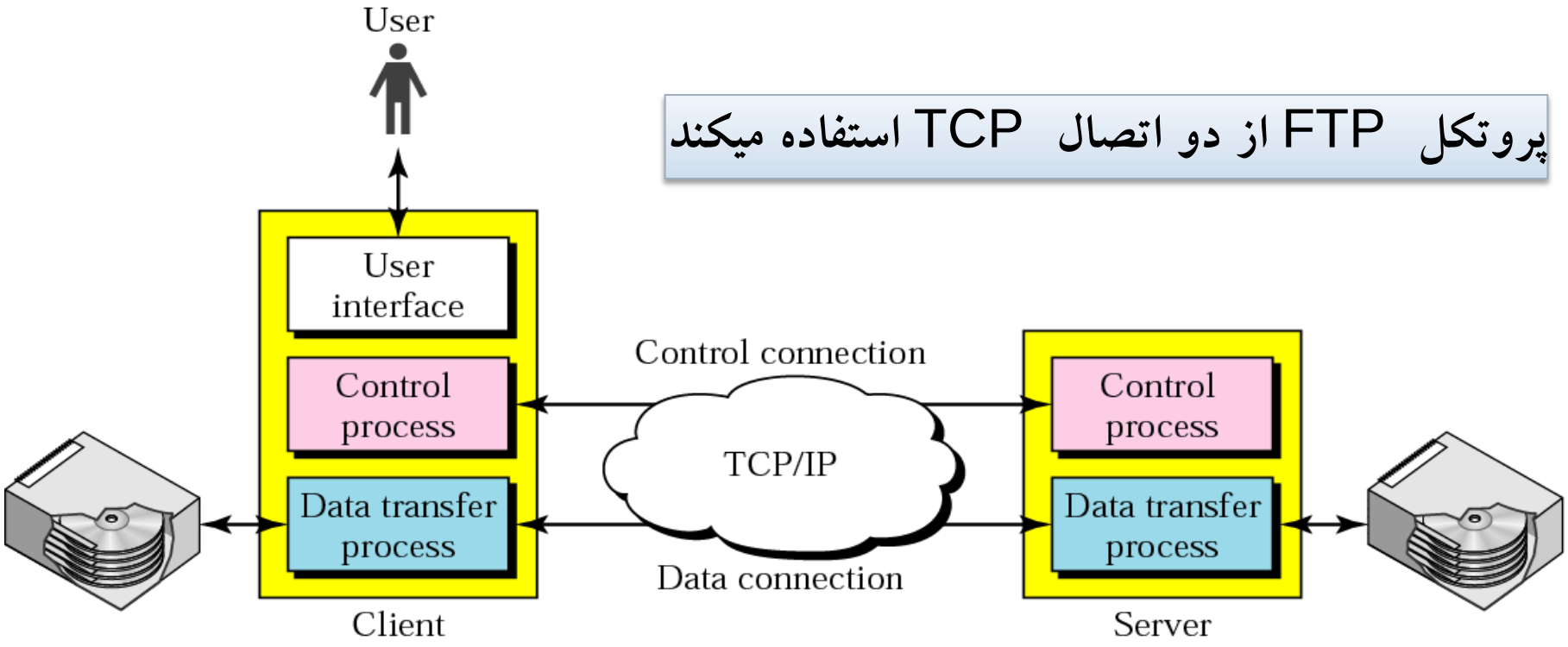
پروتکل انتقال فایل

Note:

FTP uses the services of TCP. It needs two TCP connections. The well-known port 21 is used for the control connection, and the well-known port 20 is used for the data connection.

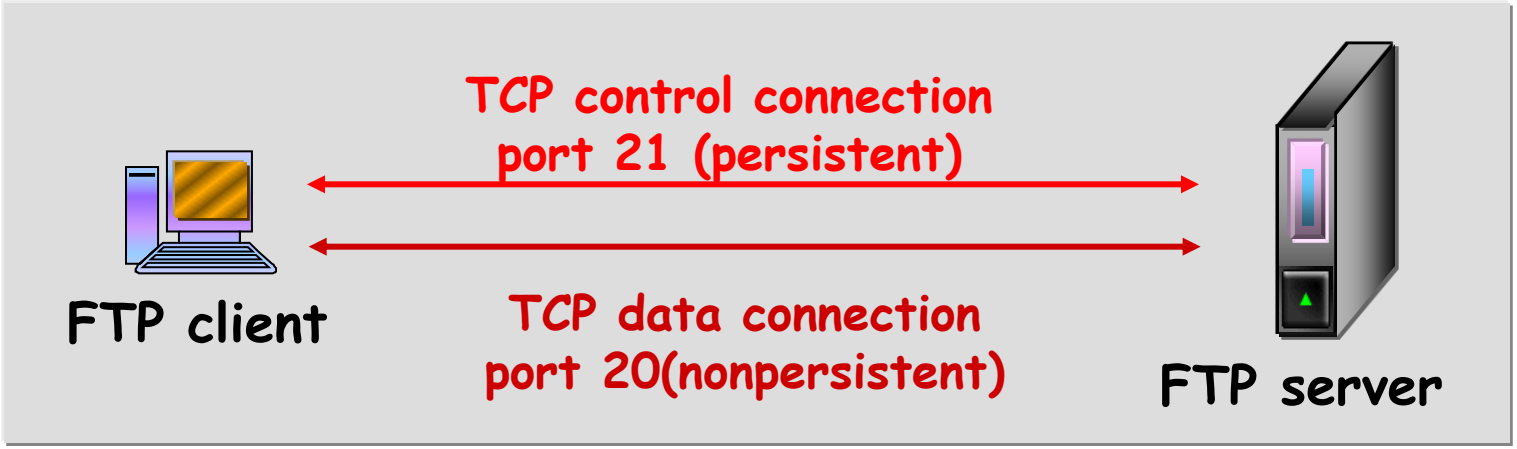
FTP

پروتکل FTP از دو اتصال TCP استفاده میکند



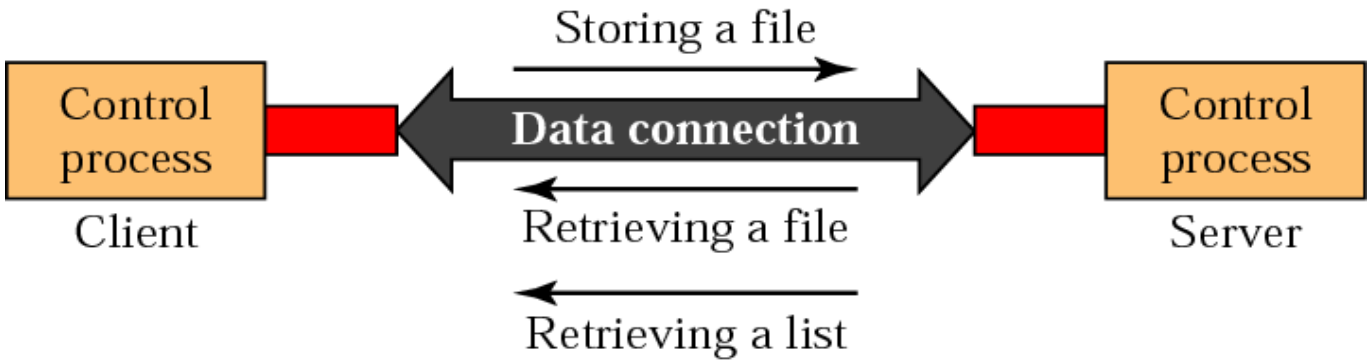
control connection: "**out of band**"

FTP: Separate Control, Data Connections

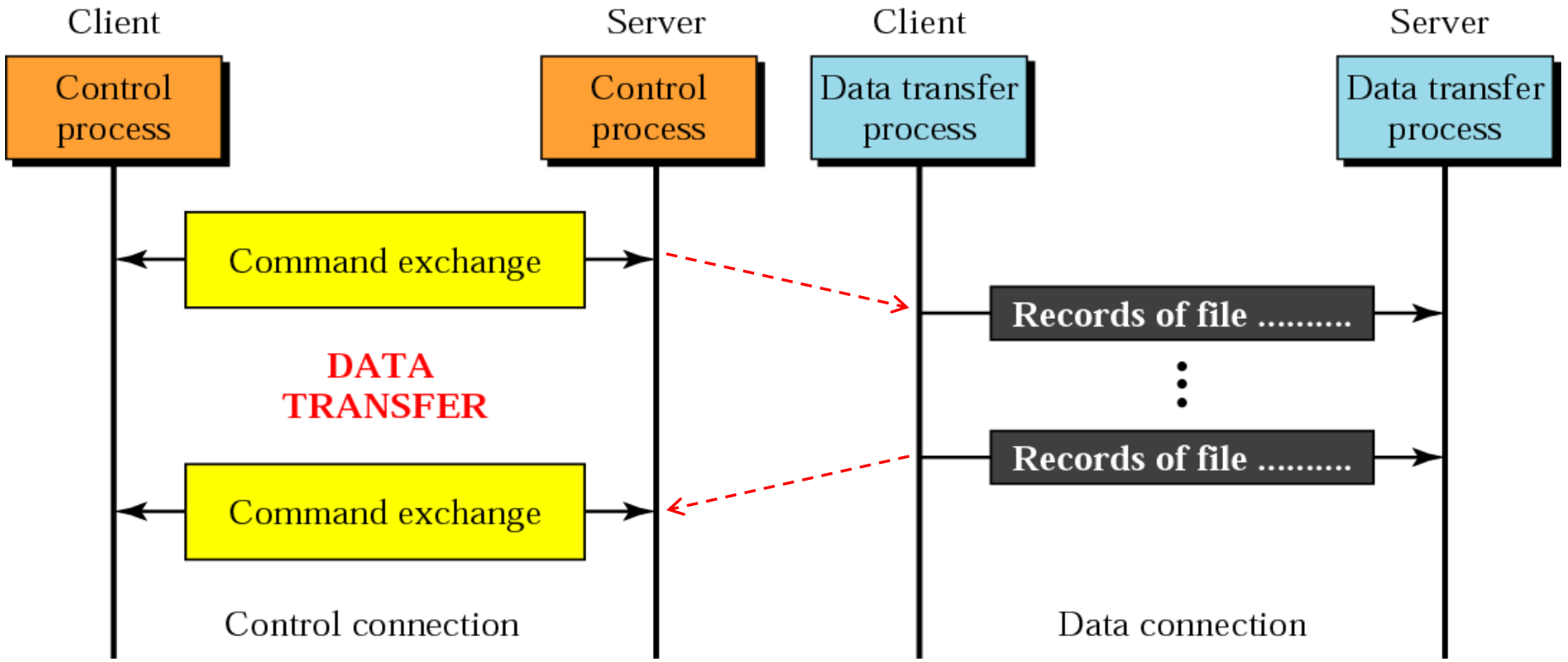


- FTP client contacts FTP server at port 21, specifying TCP as transport protocol.
- Client obtains authorization over control connection.
- Client browses remote directory by sending commands over control connection.
- When server receives a command for a file transfer, server opens a TCP data conn. to client at port 20.

File transfer



Control and data connections



List of FTP commands in UNIX

Commands
!, $, account, append, ascii, bell, binary, bye, case, cd, cdup, close, cr, delete, debug, dir, discount, form, get, glob, hash, help, lcd, ls, macdef, mdelete, mdir, mget, mkdir, mls, mode, mput, nmap, ntrans, open, prompt, proxy, sendport, put, pwd, quit, quote, recv, remotehelp, rename, reset, rmdir, runique, send, status, struct, sunique, tenex, trace, type, user, verbose,?

Example 1

```
$ ftp ftp.uok.ac.ir
Connected to ftp.uok.ac.ir
220 Server ready
Name: Abdollahpouri
Password: xxxxxxx
ftp > ls /usr/user/report
200 OK
150 Opening ASCII mode
.....
.....
226 transfer complete
ftp > close
221 Goodbye
ftp > quit
```

Example 2

```
$ ftp internic.net
Connected to internic.net
220 Server ready
Name: anonymous
331 Guest login OK, send "guest" as password
Password: guest
ftp > pwd
257 '/' is current directory
ftp > ls
200 OK
150 Opening ASCII mode
bin
...
ftp > close
221 Goodbye
ftp > quit
```

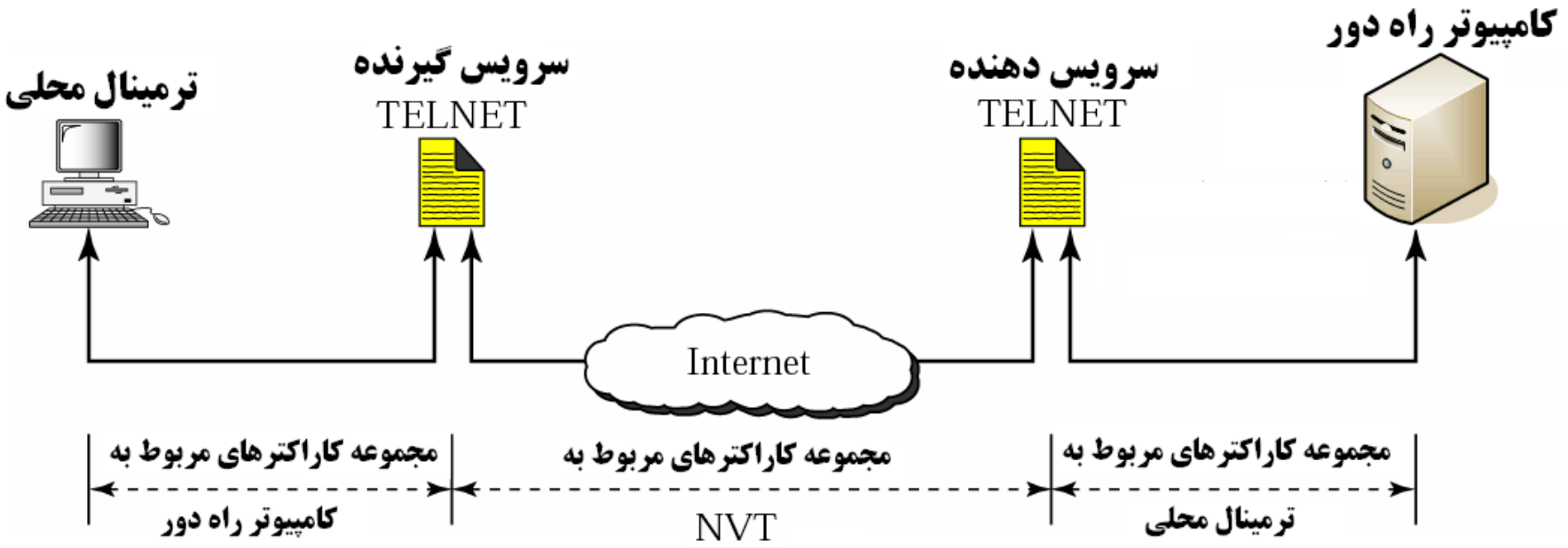
Telnet

(remote login)

دسترسی از راه دور

این پروتکل کاربر را قادر میسازد که با استفاده از یک ترمینال، از هر مکانی و با استفاده از یک خط ارتباطی همانند خط تلفن با یک سیستم راه دور ارتباط برقرار کرده و وارد سیستم شده، از آن
سرویس بگیرد.

Telnet از پروتکل TCP و پورت ۲۳ استفاده میکند


ترمینال محلی
سرویس گیرنده
TELNET
سرویس دهنده
TELNET
کامپیوتر راه دور
Internet
مجموعه کاراکترهای مربوط به
کامپیوتر راه دور
مجموعه کاراکترهای مربوط به
NVT
مجموعه کاراکترهای مربوط به
ترمینال محلی


Questions