WEB TECHNOLOGIES

CHAPTER 1

WEB ESSENTIALS: CLIENTS, SERVERS, AND COMMUNICATION

Modified by Ahmed Sallam

Based on original slides by Jeffrey C. Jackson

THE INTERNET

Technical origin: ARPANET (late 1960's)

- One of earliest attempts to network heterogeneous, geographically dispersed computers
- Email first available on ARPANET in 1972 (and quickly very popular!)

ARPANET access was limited to select DoD-funded organizations

THE INTERNET

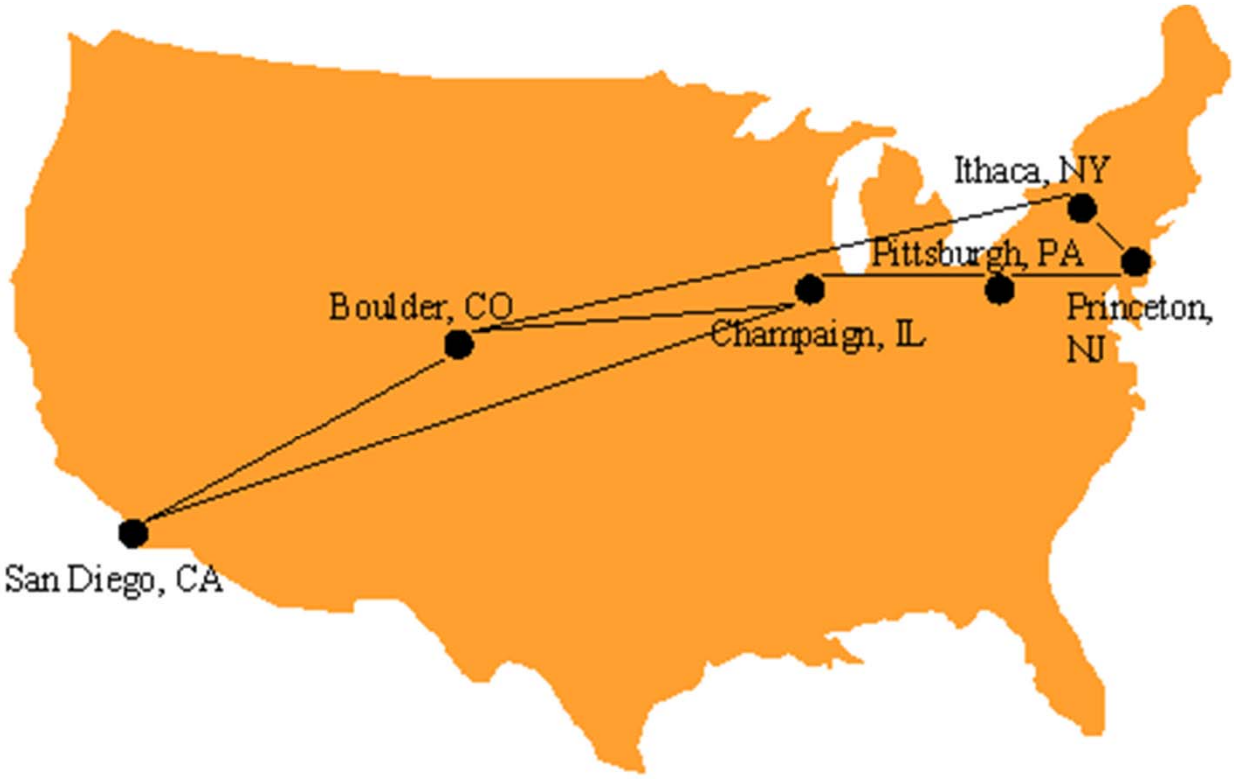
Open-access networks

- Regional university networks (e.g., SURAnet)
- CSNET for CS departments not on ARPANET

NSFNET (1985-1995)

- Primary purpose: connect supercomputer centers
- Secondary purpose: provide backbone to connect regional networks

THE INTERNET



THE INTERNET

Internet: the network of networks connected via the public backbone and communicating using TCP/IP communication protocol

- Backbone initially supplied by NSFNET, privately funded (ISP fees) beginning in 1995

INTERNET PROTOCOLS

Communication protocol: how computers talk

- Cf. telephone “protocol”: how you answer and end call, what language you speak, etc.

Internet protocols developed as part of ARPANET research

- ARPANET began using TCP/IP in 1982

Designed for use both within local area networks (LAN’s) and between networks

INTERNET PROTOCOL (IP)

IP is the fundamental protocol defining the Internet (as the name implies!)

IP address:

- 32-bit number (in IPv4)
- Associated with at most one device at a time (although device may have more than one)
- Written as four dot-separated bytes, e.g. 192.0.34.166

IP

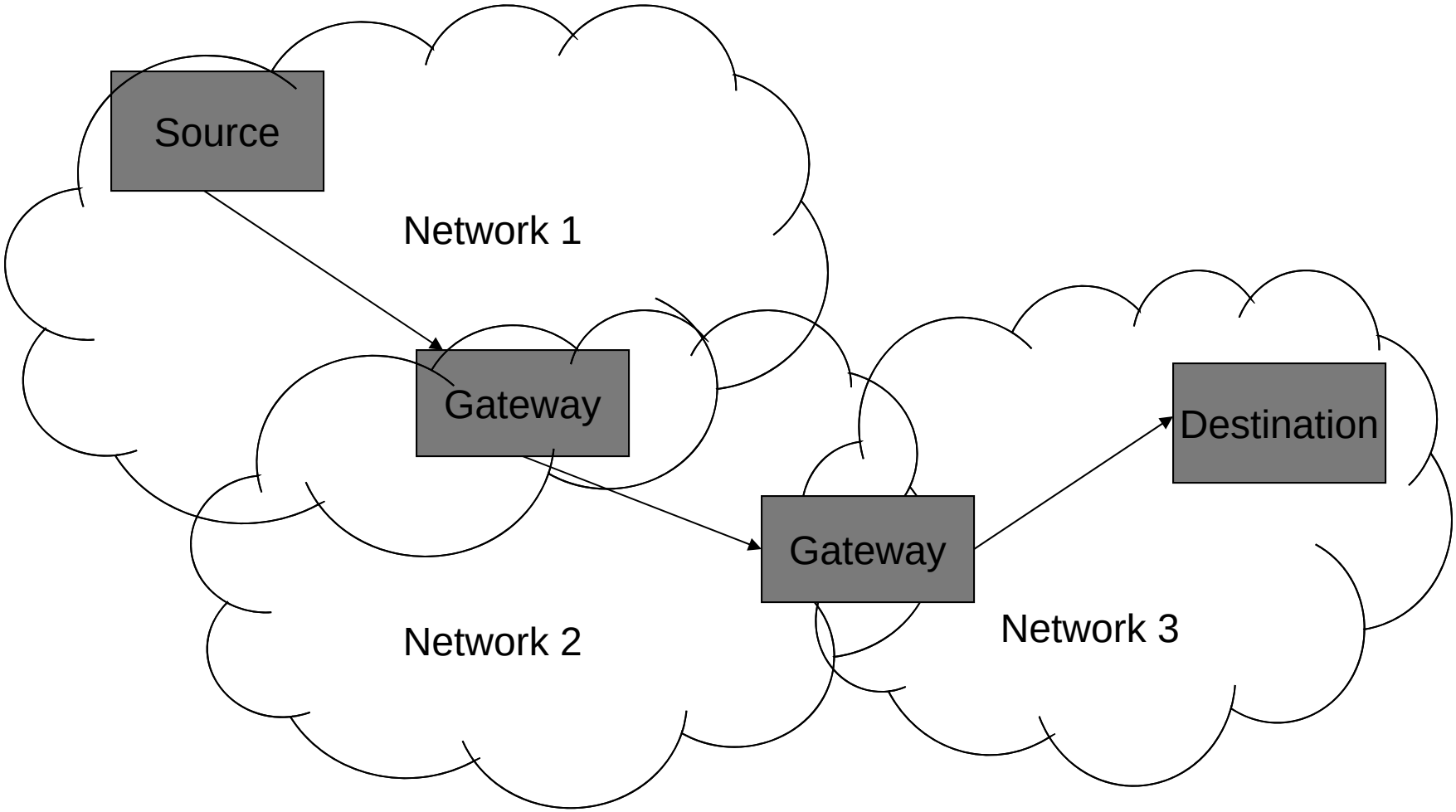
IP function: transfer data from source device to destination device

IP source software creates a packet representing the data

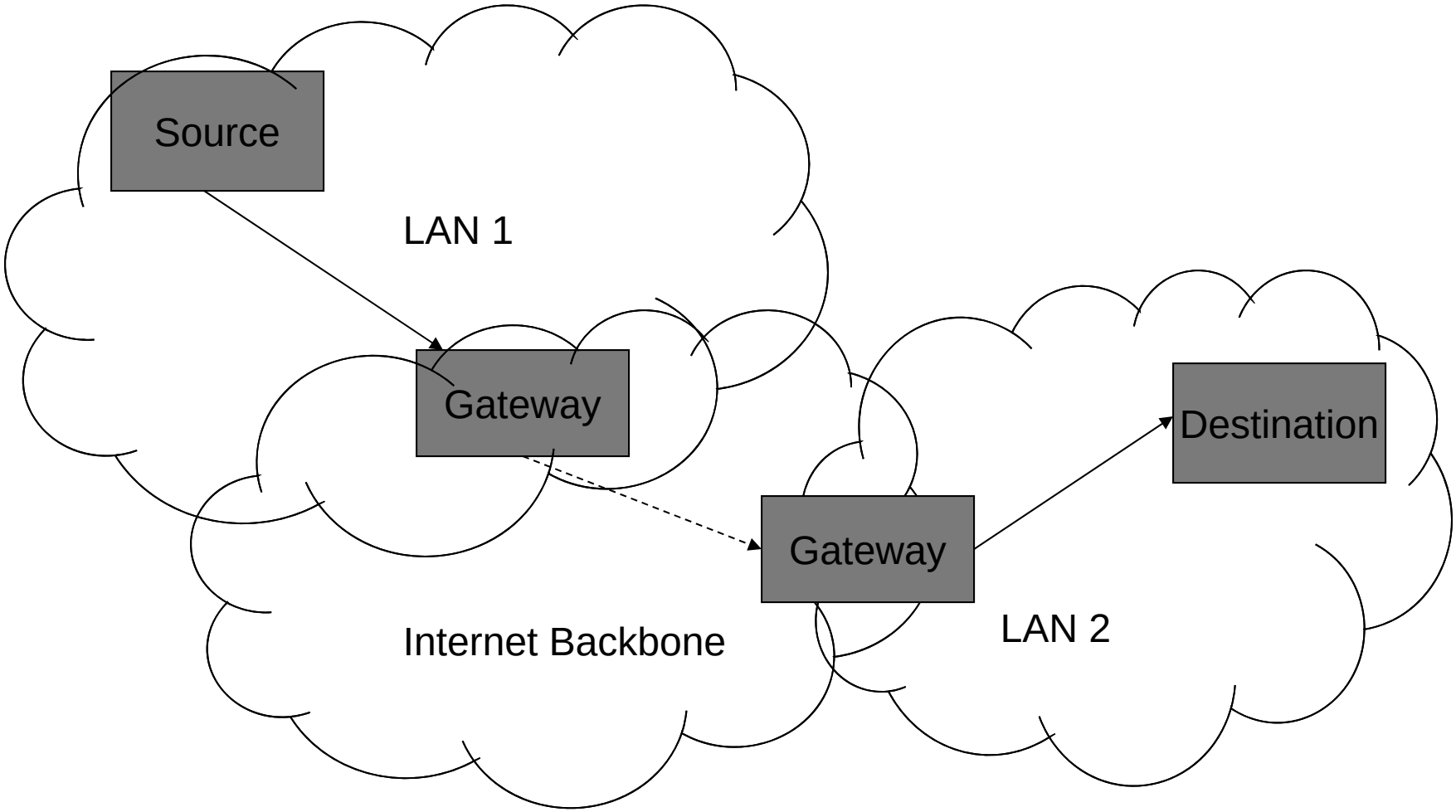
- Header: source and destination IP addresses, length of data, etc.
- Data itself

If destination is on another LAN, packet is sent to a gateway that connects to more than one network

IP


Source
Network 1
Gateway
Gateway
Destination
Network 2
Network 3

IP


Source
LAN 1
Gateway
Destination
Gateway
Internet Backbone
LAN 2

TRANSMISSION CONTROL PROTOCOL (TCP)

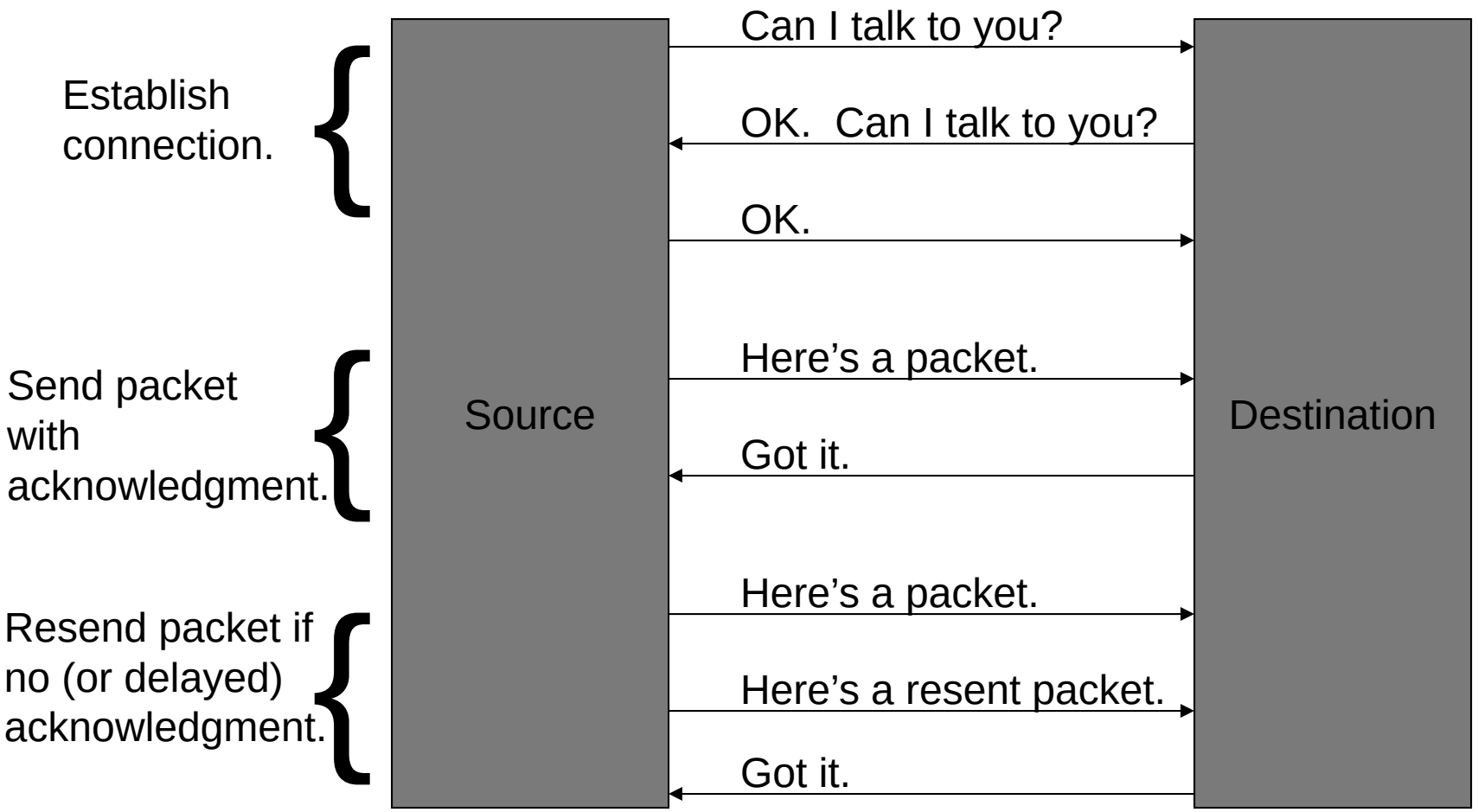
Limitations of IP:

- No guarantee of packet delivery (packets can be dropped)
- Communication is one-way (source to destination)

TCP adds concept of a connection on top of IP

- Provides guarantee that packets delivered
- Provide two-way (full duplex) communication

TCP

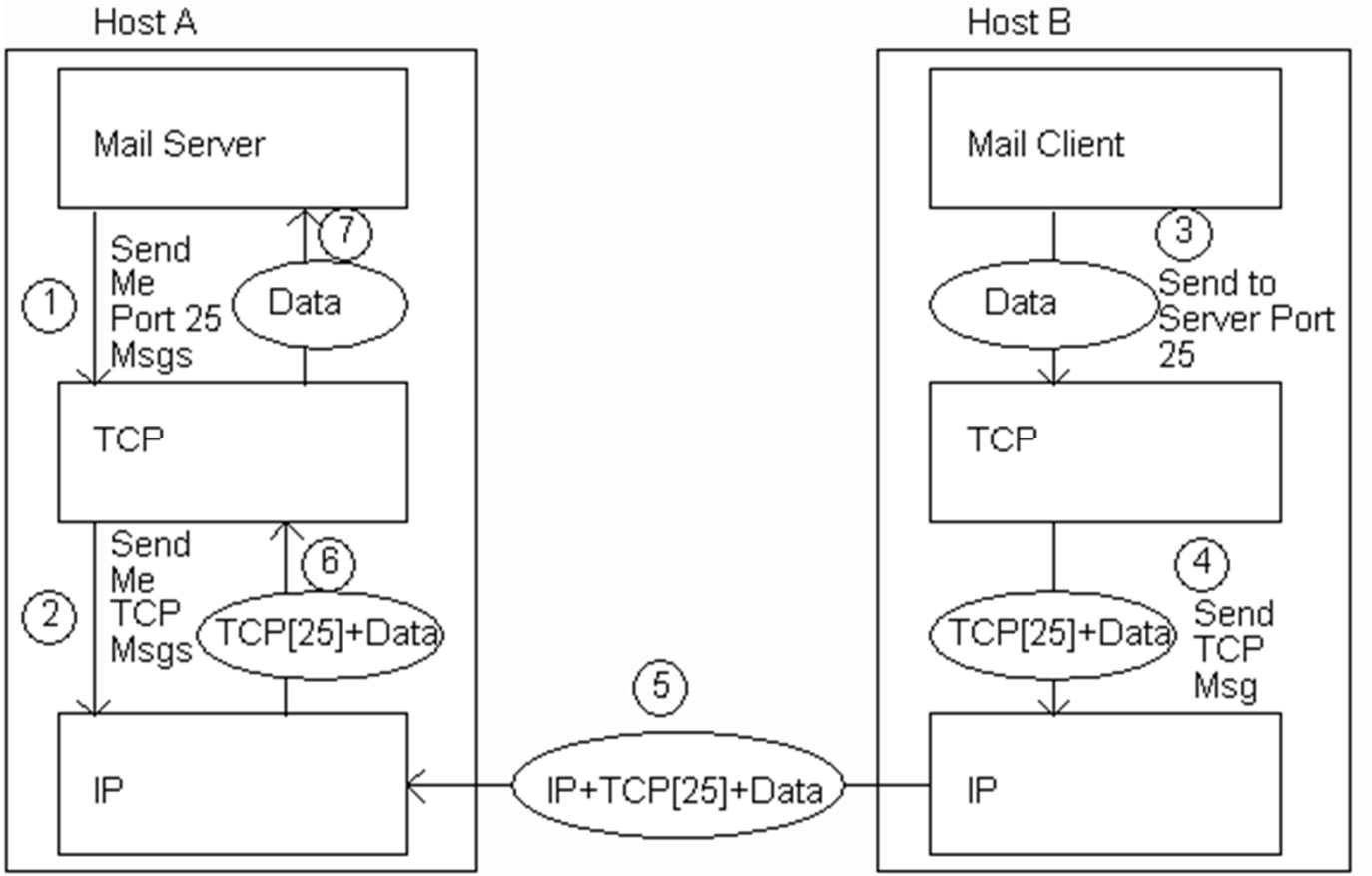

Establish connection.
Send packet with acknowledgment.
Resend packet if no (or delayed) acknowledgment.
Source
Destination
Can I talk to you?
OK. Can I talk to you?
OK.
Here's a packet.
Got it.
Here's a packet.
Here's a resent packet.
Got it.

TCP

TCP also adds concept of a port

- TCP header contains port number representing an application program on the destination computer
- Some port numbers have standard meanings
    - Example: port 25 is normally used for email transmitted using the Simple Mail Transfer Protocol (SMTP)
- Other port numbers are available first-come-first served to any application

TCP


Host A
Host B
Mail Server
Mail Client
Send Me Port 25 Msgs
Data
Send to Server Port 25
TCP
TCP
Send Me TCP Msgs
TCP[25]+Data
TCP[25]+Data
Send TCP Msg
IP
IP+TCP[25]+Data
IP
1
2
3
4
5
6
7

USER DATAGRAM PROTOCOL (UDP)

Like TCP in that:

- Builds on IP
- Provides port concept

Unlike TCP in that:

- No connection concept
- No transmission guarantee

Advantage of UDP vs. TCP:

- Lightweight, so faster for one-time messages

DOMAIN NAME SERVICE (DNS)

DNS is the "phone book" for the Internet

- Map between host names and IP addresses
- DNS often uses UDP for communication

Host names

- Labels separated by dots, e.g., `www.example.org`
- Final label is *top-level domain*
  - Generic: .com, .org, etc.
  - Country-code: .us, .il, etc.

DNS

Domains are divided into second-level domains, which can be further divided into subdomains, etc.

- E.g., in www.example.com, example is a second-level domain

A host name plus domain name information is called the fully qualified domain name of the computer

- Above, www is the host name, www.example.com is the FQDN

DNS

nslookup program provides command-line access to DNS (on most systems)

looking up a host name given an IP address is known as a reverse lookup

- Recall that single host may have mutliple IP addresses.
- Address returned is the canonical IP address specified in the DNS system.

ANALOGY TO TELEPHONE NETWORK

IP ~ the telephone network

TCP ~ calling someone who answers, having a conversation, and hanging up

UDP ~ calling someone and leaving a message

DNS ~ directory assistance

HIGHER-LEVEL PROTOCOLS

Many protocols build on TCP

- Telephone analogy: TCP specifies how we initiate and terminate the phone call, but some other protocol specifies how we carry on the actual conversation

Some examples:

- SMTP (email)
- FTP (file transfer)
- HTTP (transfer of Web documents)

WORLD WIDE WEB

Originally, one of several systems for organizing Internet-based information

- Competitors: WAIS, Gopher, ARCHIE

Distinctive feature of Web: support for hypertext (text containing links)

- Communication via Hypertext Transport Protocol (HTTP)
- Document representation using Hypertext Markup Language (HTML)

WORLD WIDE WEB

The Web is the collection of machines (Web servers) on the Internet that provide information, particularly HTML documents, via HTTP.

Machines that access information on the Web are known as Web clients. A Web browser is software used by an end user to access the Web.

HYPERTEXT TRANSPORT PROTOCOL (HTTP)

HTTP is based on the request-response communication model:

- Client sends a request
- Server sends a response

HTTP is a stateless protocol:

- The protocol does not require the server to remember anything about the client between requests.

HTTP

Normally implemented over a TCP connection (80 is standard port number for HTTP)

Typical browser-server interaction:

- User enters Web address in browser
- Browser uses DNS to locate IP address
- Browser opens TCP connection to server
- Browser sends HTTP request over connection
- Server sends HTTP response to browser over connection
- Browser displays body of response in the client area of the browser window

HTTP

The information transmitted using HTTP is often entirely text

Can use the Internet's Telnet protocol to simulate browser request and view server response

HTTP

Connect

```
$ telnet www.example.org 80
Trying 192.0.34.166...
Connected to www.example.com
(192.0.34.166).
Escape character is '^]'.
```

Send Request

```
GET / HTTP/1.1
Host: www.example.org
```

Receive Response

```
HTTP/1.1 200 OK
Date: Thu, 09 Oct 2003 20:30:49 GMT
…
```

HTTP REQUEST

Structure of the request:

- start line
- header field(s)
- blank line
- optional body

HTTP REQUEST

Structure of the request:

- **start line**
- header field(s)
- blank line
- optional body

HTTP REQUEST

Start line

- Example: `GET / HTTP/1.1`

Three space-separated parts:

- HTTP request method
- Request-URI
- HTTP version

HTTP REQUEST

Start line

- Example: `GET / HTTP/1.1`

Three space-separated parts:

- HTTP request method
- Request-URI
- **HTTP version**
  - We will cover 1.1, in which version part of start line must be exactly as shown

HTTP REQUEST

Start line

- Example: `GET / HTTP/1.1`

Three space-separated parts:

- HTTP request method
- **Request-URI**
- HTTP version

HTTP REQUEST

Uniform Resource Identifier (URI)

- Syntax: *`scheme : scheme-depend-part`*
  - Ex: In `http://www.example.com/` the scheme is `http`
- Request-URI is the portion of the requested URI that follows the host name (which is supplied by the required Host header field)
  - Ex: `/` is Request-URI portion of `http://www.example.com/`

URI

URI's are of two types:

- Uniform Resource Name (URN)
  - Can be used to identify resources with unique names, such as books (which have unique ISBN's)
  - Scheme is `urn`
- Uniform Resource Locator (URL)
  - Specifies location at which a resource can be found
  - In addition to `http`, some other URL schemes are `https`, `ftp`, `mailto`, and `file`

HTTP REQUEST

Start line

- Example: `GET / HTTP/1.1`

Three space-separated parts:

- **HTTP request method**
- Request-URI
- HTTP version

HTTP REQUEST

Common request methods:

- GET
  - Used if link is clicked or address typed in browser
  - No body in request with GET method
- POST
  - Used when submit button is clicked on a form
  - Form information contained in body of request
- HEAD
  - Requests that only header fields (no body) be returned in the response

HTTP REQUEST

Structure of the request:

- start line
- **header field(s)**
- blank line
- optional body

HTTP REQUEST

Header field structure:

- *field name* : *field value*

Syntax

- Field name is not case sensitive
- Field value may continue on multiple lines by starting continuation lines with white space
- Field values may contain MIME types, quality values, and wildcard characters (*'s)

HTTP REQUEST

Common header fields:

- Host: host name from URL (required)
- User-Agent: type of browser sending request
- Accept: MIME types of acceptable documents
- Connection: value `close` tells server to close connection after single request/response
- Content-Type: MIME type of (POST) body, normally application/x-www-form-urlencoded
- Content-Length: bytes in body
- Referer: URL of document containing link that supplied URI for this HTTP request

MULTIPURPOSE INTERNET MAIL EXTENSIONS (MIME)

Convention for specifying content type of a message

- In HTTP, typically used to specify content type of the body of the response

MIME content type syntax:

- *top-level type / subtype*

Examples: text/html, image/jpeg

HTTP QUALITY VALUES AND WILDCARDS

Example header field with quality values:

```
accept:
   text/xml,text/html;q=0.9,
   text/plain;q=0.8, image/jpeg,
   image/gif;q=0.2,*/*;q=0.1
```

Quality value applies to all preceding items

Higher the value, higher the preference

Note use of wildcards to specify quality 0.1 for any MIME type not specified earlier

HTTP RESPONSE

Structure of the response:

- status line
- header field(s)
- blank line
- optional body

HTTP RESPONSE

Structure of the response:

- **status line**
- header field(s)
- blank line
- optional body

HTTP RESPONSE

Status line

- Example: HTTP/1.1 200 OK

Three space-separated parts:

- HTTP version
- status code
- reason phrase (intended for human use)

HTTP RESPONSE

Status code

- Three-digit number
- First digit is class of the status code:
  - 1=Informational
  - 2=Success
  - 3=Redirection (alternate URL is supplied)
  - 4=Client Error
  - 5=Server Error
- Other two digits provide additional information

HTTP RESPONSE

Structure of the response:

- status line
- **header field(s)**
- blank line
- optional body

HTTP RESPONSE

Common header fields:

- Connection, Content-Type, Content-Length
- Date: date and time at which response was generated (required)
- Location: alternate URI if status is redirection
- Last-Modified: date and time the requested resource was last modified on the server
- Expires: date and time after which the client's copy of the resource will be out-of-date
- ETag: a unique identifier for this version of the requested resource (changes if resource changes)

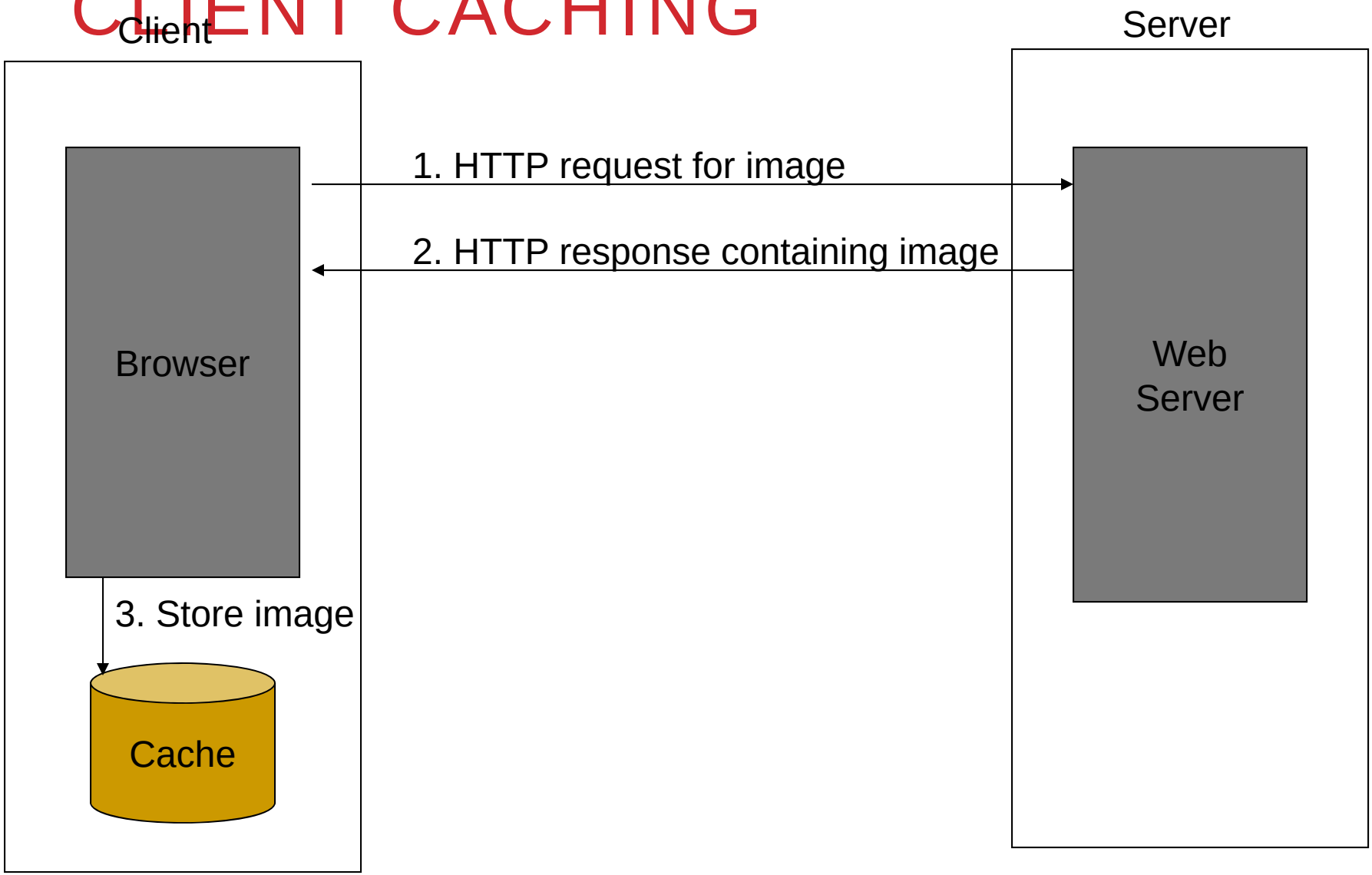
CLIENT CACHING

A cache is a local copy of information obtained from some other source

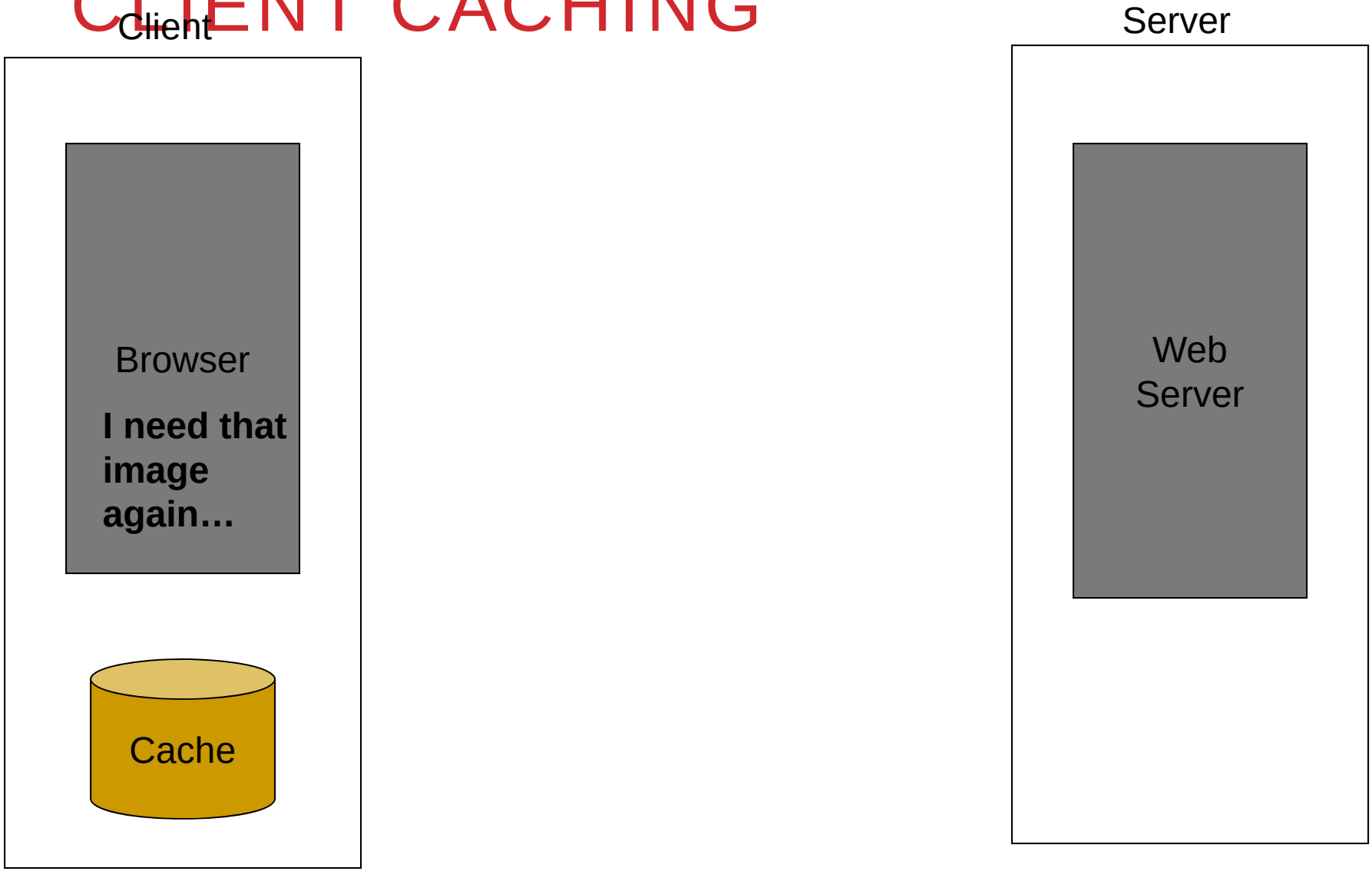
Most web browsers use cache to store requested resources so that subsequent requests to the same resource will not necessarily require an HTTP request/response

- Ex: icon appearing multiple times in a Web page

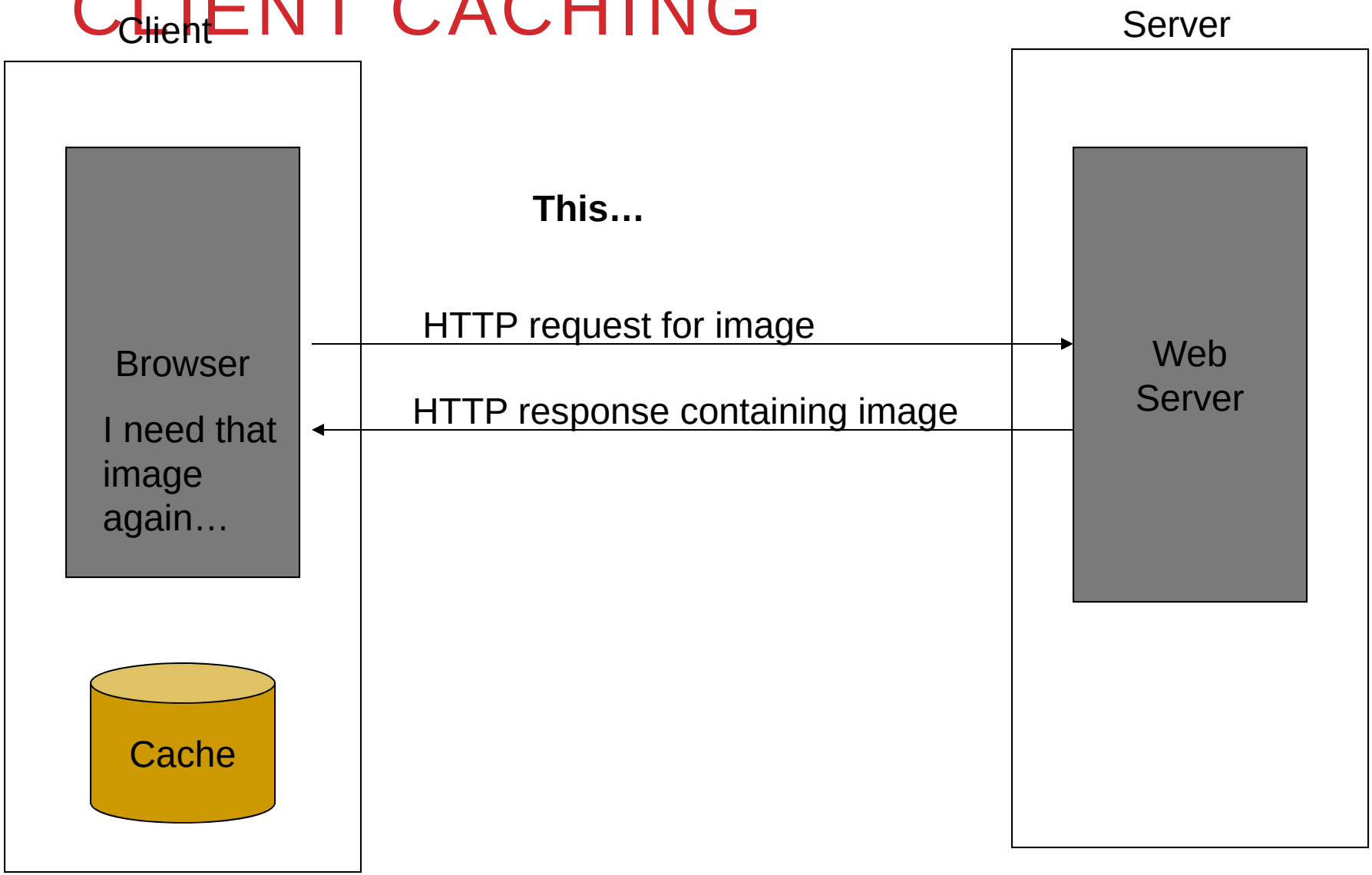
CLIENT CACHING



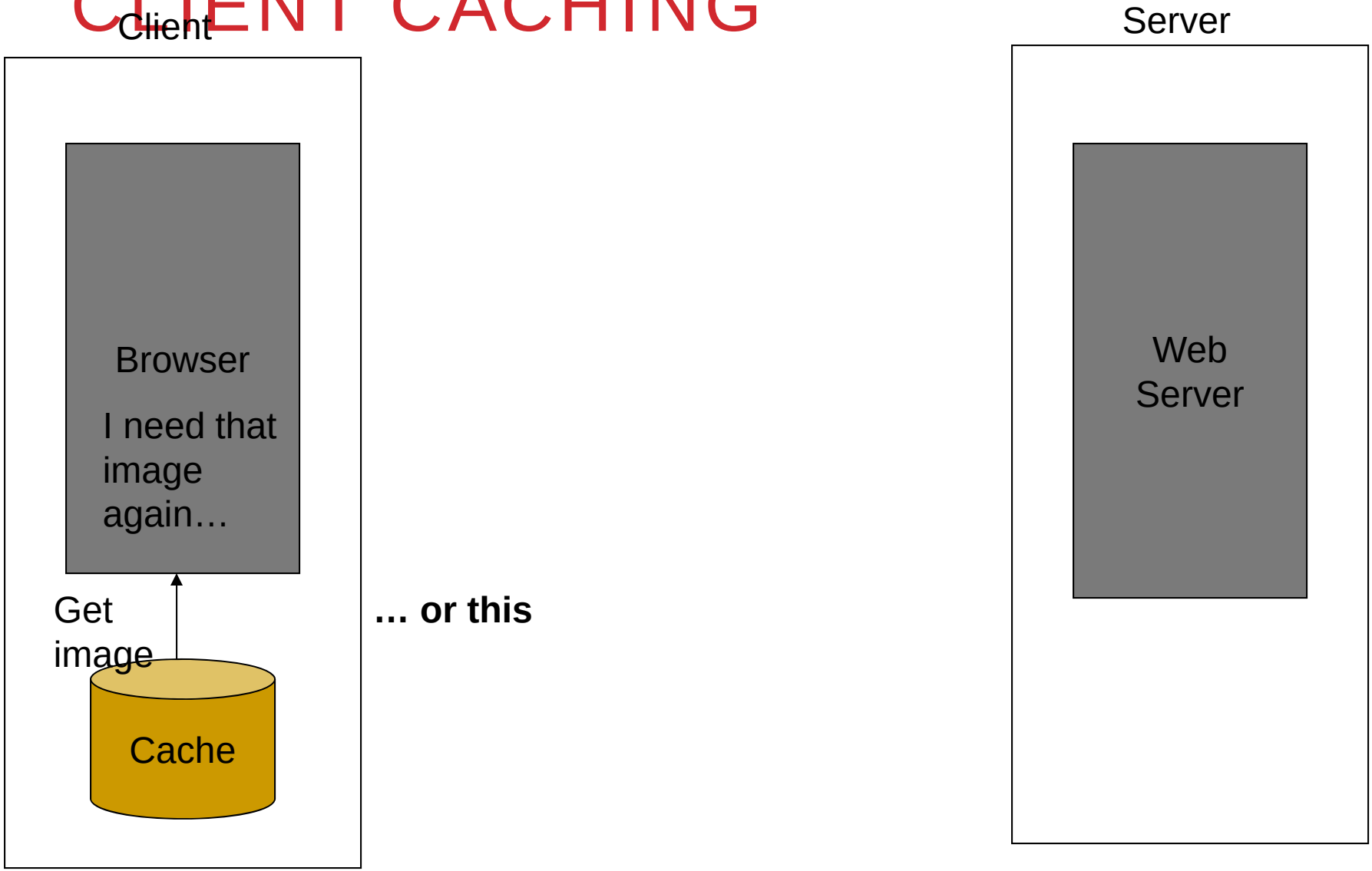
CLIENT CACHING



CLIENT CACHING



CLIENT CACHING



CLIENT CACHING

Cache advantages

- (Much) faster than HTTP request/response
- Less network traffic
- Less load on server

Cache disadvantage

- Cached copy of resource may be invalid (inconsistent with remote version)

CLIENT CACHING

Validating cached resource:

- Send HTTP HEAD request and check Last-Modified or ETag header in response
- Compare current date/time with Expires header sent in response containing resource
- If no Expires header was sent, use heuristic algorithm to estimate value for Expires
    - Ex: Expires = 0.01 * (Date – Last-Modified) + Date

CHARACTER SETS

Every document is represented by a string of integer values (code points)

The mapping from code points to characters is defined by a character set

Some header fields have character set values:

- Accept-Charset: request header listing character sets that the client can recognize
  - Ex: accept-charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
- Content-Type: can include character set used to represent the body of the HTTP message
  - Ex: Content-Type: text/html; charset=UTF-8

CHARACTER SETS

Technically, many "character sets" are actually character encodings

- An encoding represents code points using variable-length byte strings
- Most common examples are Unicode-based encodings UTF-8 and UTF-16

IANA maintains complete list of Internet-recognized character sets/encodings

CHARACTER SETS

Typical US PC produces ASCII documents

US-ASCII character set can be used for such documents, but is not recommended

UTF-8 and ISO-8859-1 are supersets of US-ASCII and provide international compatibility

- UTF-8 can represent all ASCII characters using a single byte each and arbitrary Unicode characters using up to 4 bytes each
- ISO-8859-1 is 1-byte code that has many characters common in Western European languages, such as é

WEB CLIENTS

Many possible web clients:

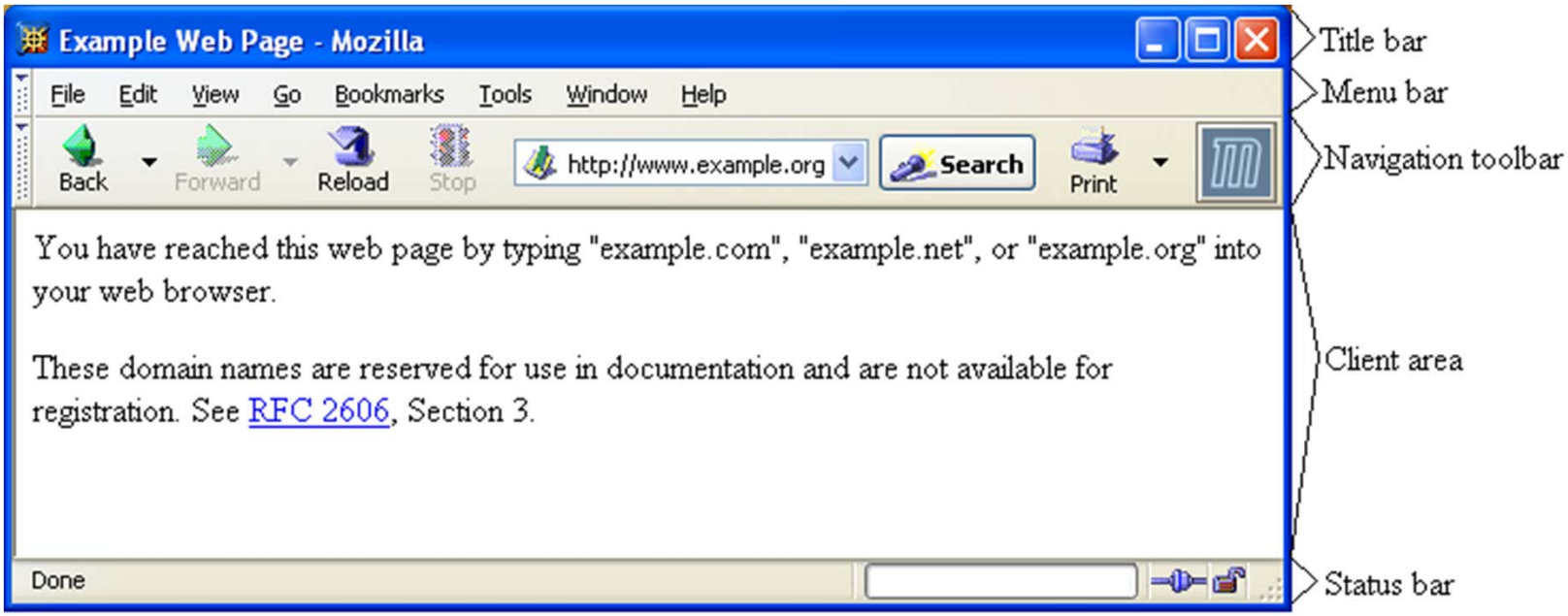
- Text-only “browser” (lynx)
- Mobile phones
- Robots (software-only clients, e.g., search engine “crawlers”)
- etc.

We will focus on traditional web browsers

WEB BROWSERS

First graphical browser running on general-purpose platforms: Mosaic (1993)

WEB BROWSERS

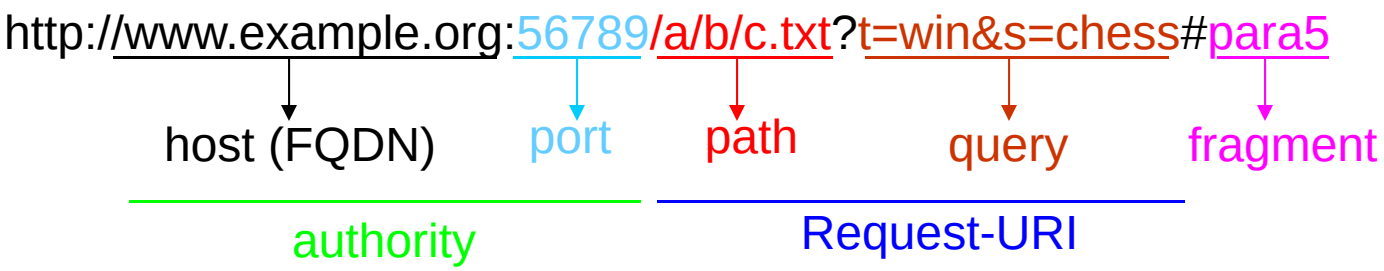


WEB BROWSERS

Primary tasks:

- Convert web addresses (URL's) to HTTP requests
- Communicate with web servers via HTTP
- Render (appropriately display) documents returned by a server

HTTP URL'S



Browser uses authority to connect via TCP

Request-URI included in start line (/ used for path if none supplied)

Fragment identifier not sent to server (used to scroll browser client area)

WEB BROWSERS

Standard features

- Save web page to disk
- Find string in page
- Fill forms automatically (passwords, CC numbers, …)
- Set preferences (language, character set, cache and HTTP parameters)
- Modify display style (e.g., increase font sizes)
- Display raw HTML and HTTP header info (e.g., Last-Modified)
- Choose browser themes (skins)
- View history of web addresses visited
- Bookmark favorite pages for easy return

WEB BROWSERS

Additional functionality:

- Execution of scripts (e.g., drop-down menus)
- Event handling (e.g., mouse clicks)
- GUI for controls (e.g., buttons)
- Secure communication with servers
- Display of non-HTML documents (e.g., PDF) via plug-ins

WEB SERVERS

Basic functionality:

- Receive HTTP request via TCP
- Map Host header to specific virtual host (one of many host names sharing an IP address)
- Map Request-URI to specific resource associated with the virtual host
  - File: Return file in HTTP response
  - Program: Run program and return output in HTTP response
- Map type of resource to appropriate MIME type and use to set Content-Type header in HTTP response
- Log information about the request and response

WEB SERVERS

httpd: UIUC, primary Web server c. 1995

Apache: “A patchy” version of httpd, now the most popular server (esp. on Linux platforms)

IIS: Microsoft Internet Information Server

Tomcat:

- Java-based
- Provides container (Catalina) for running Java servlets (HTML-generating programs) as back-end to Apache or IIS
- Can run stand-alone using Coyote HTTP front-end

WEB SERVERS

Some Coyote communication parameters:

- Allowed/blocked IP addresses
- Max. simultaneous active TCP connections
- Max. queued TCP connection requests
- “Keep-alive” time for inactive TCP connections

Modify parameters to tune server performance

WEB SERVERS

Some Catalina container parameters:

- Virtual host names and associated ports
- Logging preferences
- Mapping from Request-URI’s to server resources
- Password protection of resources
- Use of server-side caching

TOMCAT WEB SERVER

HTML-based server administration

Browse to
http://localhost:8080
and click on Server Administration link

- `localhost` is a special host name that means “this machine”