Application Layer

COMP90007

Internet Technologies

The World Wide Web (WWW)

- World Wide Web key components are?
  - Client and Server software – **Firefox** is the client software for web access where **Apache** is on the server side
  - Web markup languages - **HTML** – how webpages are coded
  - Web scripting languages – More dynamicity to webpages - **Javascript**
  - **HTTP** – about how to transfer

Web Access

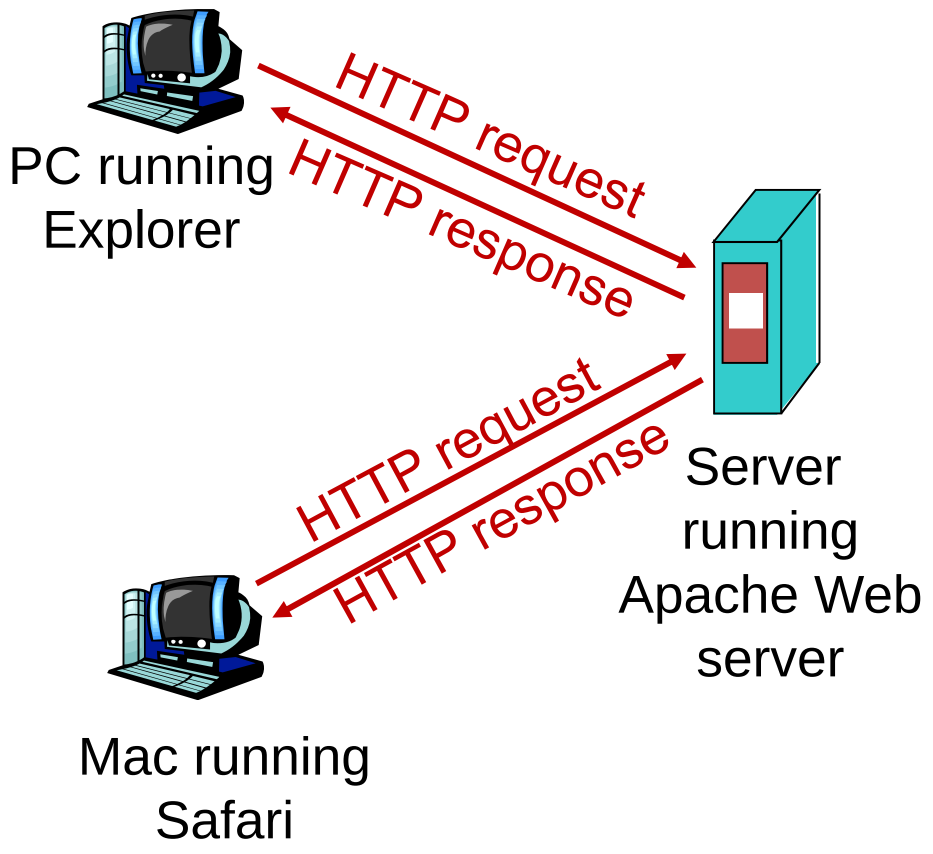
- A web page consists of objects
- An object can be HTML file but also JPEG image, Java applet, audio file, …
- A web page consists of a base HTML file which includes several referenced objects
- Each object is addressable by a URL (uniform resource locator)
- Example URL:

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

HTTP: hypertext transfer protocol

HyperText "text … cross-referencing between sections of text and associated graphic material"

- HTTP is at the application layer
- client/server model
  - ***client:*** browser that requests, receives and displays Web objects
  - ***server:*** Web server sends objects in response to requests



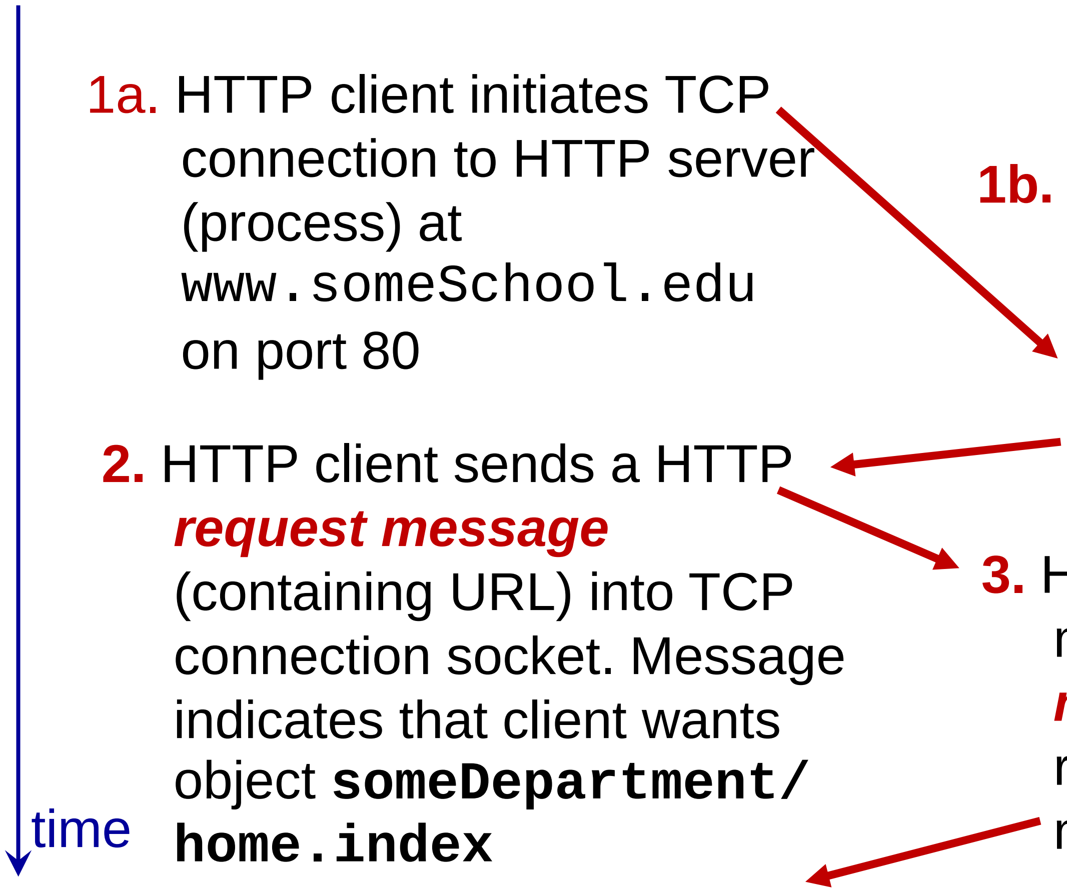
HTTP Connections

- Non-persistent HTTP
  - at most one object sent over a TCP connection
- Persistent HTTP
  - multiple objects can be sent over a single TCP connection between client and server

Non-persistent HTTP (I)

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

contains text and references to 10 images



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

1b. HTTP server at host **`www.someSchool.edu`** waiting for TCP connection at port 80. Accepts connection, notifying client

2. HTTP client sends a 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

Non-persistent HTTP (II)

4. HTTP client receives response message containing HTML file

5. HTTP server closes TCP connection.

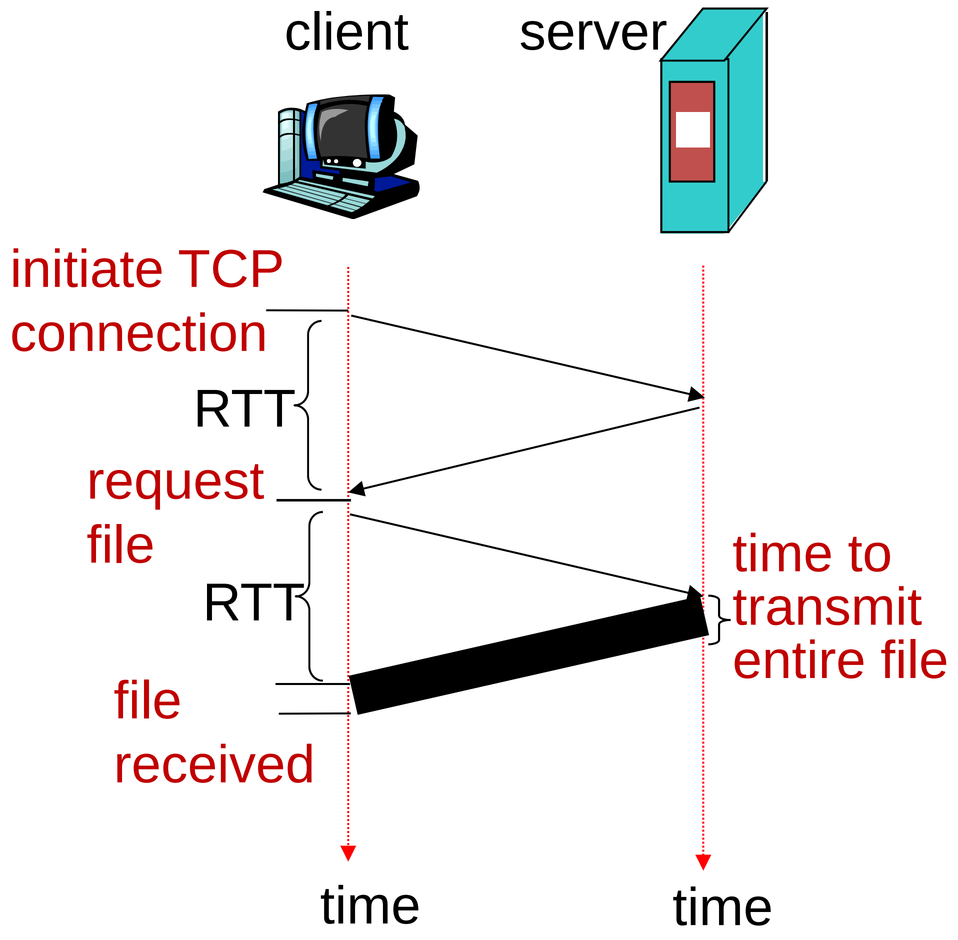
6. Parses HTML file, and finds 10 referenced jpeg objects

time

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

Non-Persistent HTTP: Response Time

- Round Trip Time (RTT) – time for 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 response time = 2 RTT+ file transmission time



Non-Persistent HTTP – Issues

- Requires new connection per requested object
- OS overhead for *each* TCP connection
- Delivery delay of 2 RTTs per requested object

Persistent HTTP

- Server leaves connection open after sending response
- Subsequent HTTP messages between same client/server sent over open connection
- *Pipelining* – client sends request as soon as it encounters a referenced object
  - ➔as little as one RTT for all the referenced objects
- Server closes a connection if it hasn't been used for some time

HTTP Request Message: Example

request line (GET, POST, HEAD commands)

header lines

```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

Persistent HTTP

indicates end of header lines

HTTP Response Message: Example

200 OK – request succeeded, requested object later in this msg
….
404 Not Found – requested document not found on this server

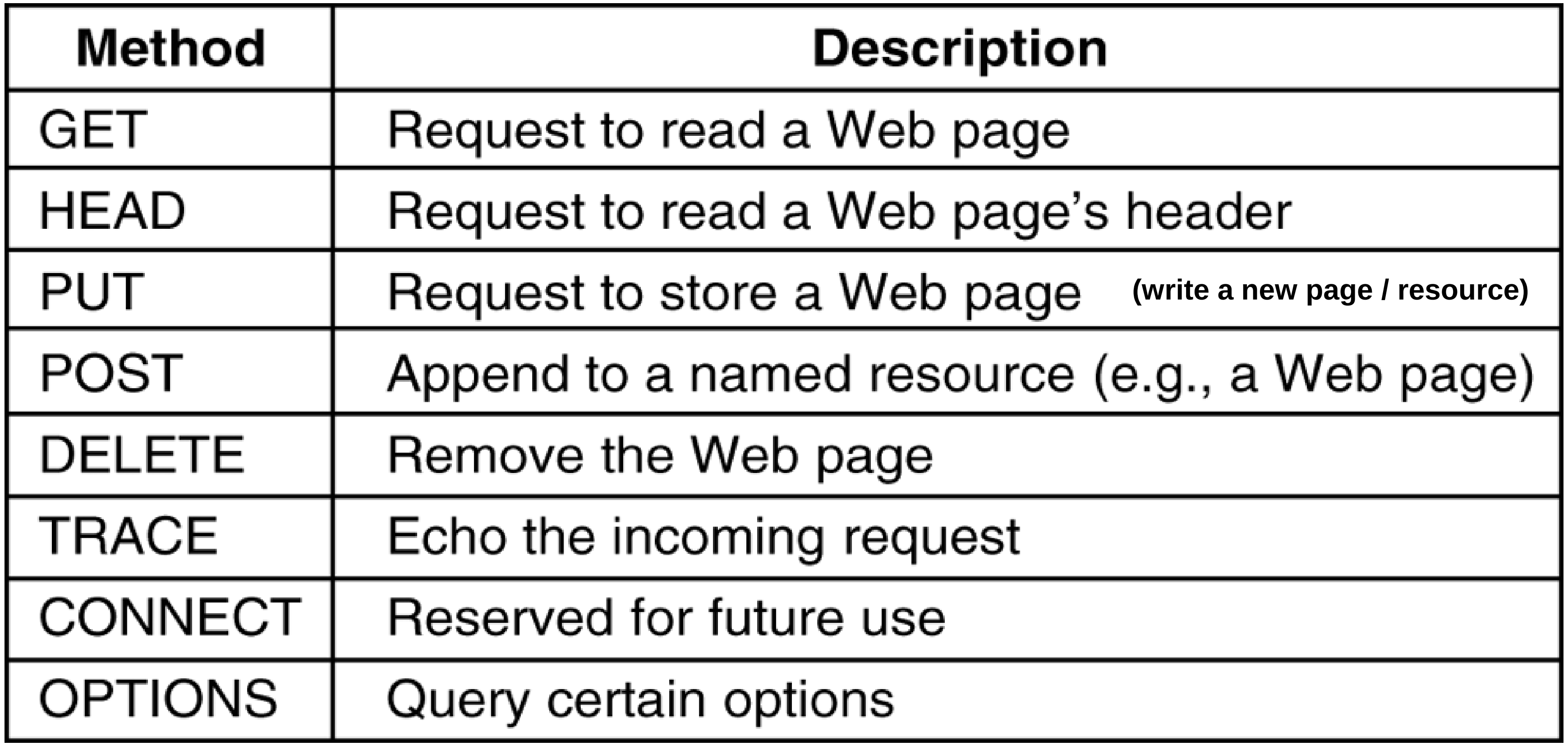
status line:

header lines

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-1\r\n
\r\n
data data data data data ...
```

data, e.g., requested HTML file

HTTP Request Methods



Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page **(write a new page / resource)**
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

HTTP Error Codes

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

Cookies

- **The http servers are stateless**
- Cookies to place small amount (<4Kb) of info on users computer and re-use deterministically (RFC 2109)
- Questionable mechanism for tracking users (invisibly perhaps)

User-server Interaction: Cookies Example 1

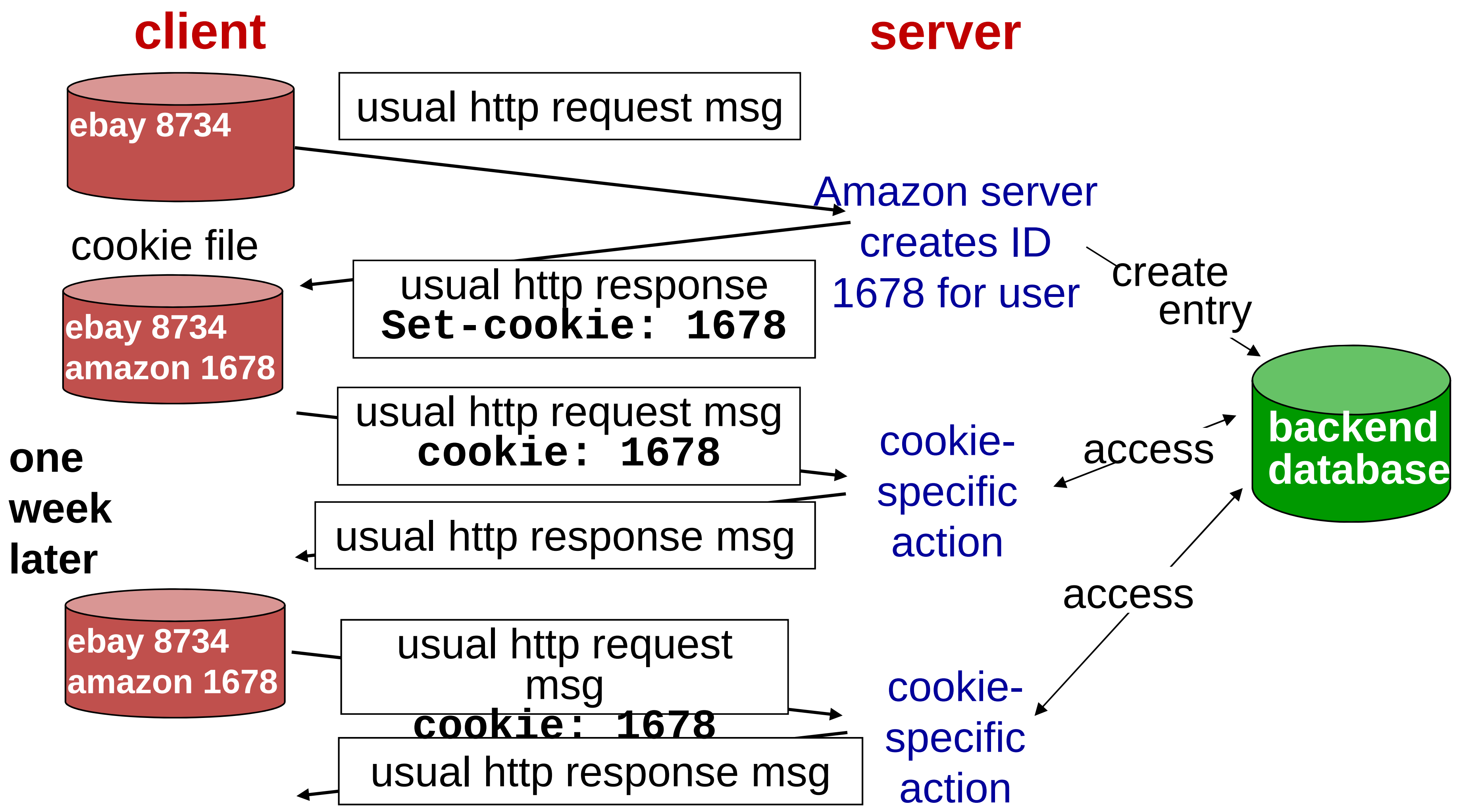
Susan always accesses the Internet from her (*cookie-enabled*) home PC. She visits a specific (*cookie-enabled*) e-commerce site for the first time

- When the initial HTTP requests arrives at the site, the site creates:
  - unique ID
  - entry in backend database for ID
- The e-commerce site then responds to Susan's browser, including in the HTTP response
  - Set-cookie: 1234 — ID

User-server Interaction: Cookies Example (Contd)

- Susan’s browser appends a line to a cookie file that it manages
  - www.e-commerce-site.com 1234
- Next time Susan request a page from that site, a cookie header line will be added to her request
  - Cookie: 1234
- The server will then perform a cookie-specific action

Keeping "state" with Cookies – Example 2



Beyond User Tracking: Advantages of Cookies

- Authorization
- Shopping carts
- Recommendations
- User session state

Cookies vs Sessions

- **Both introduce "memory" or state into HTTP and are about multiple TCP connections**

Sessions

- Sessions information regarding visitor's interaction stored at the server side: upto some hours
- When user closes the website, the session ends
- Sessions information size can be large

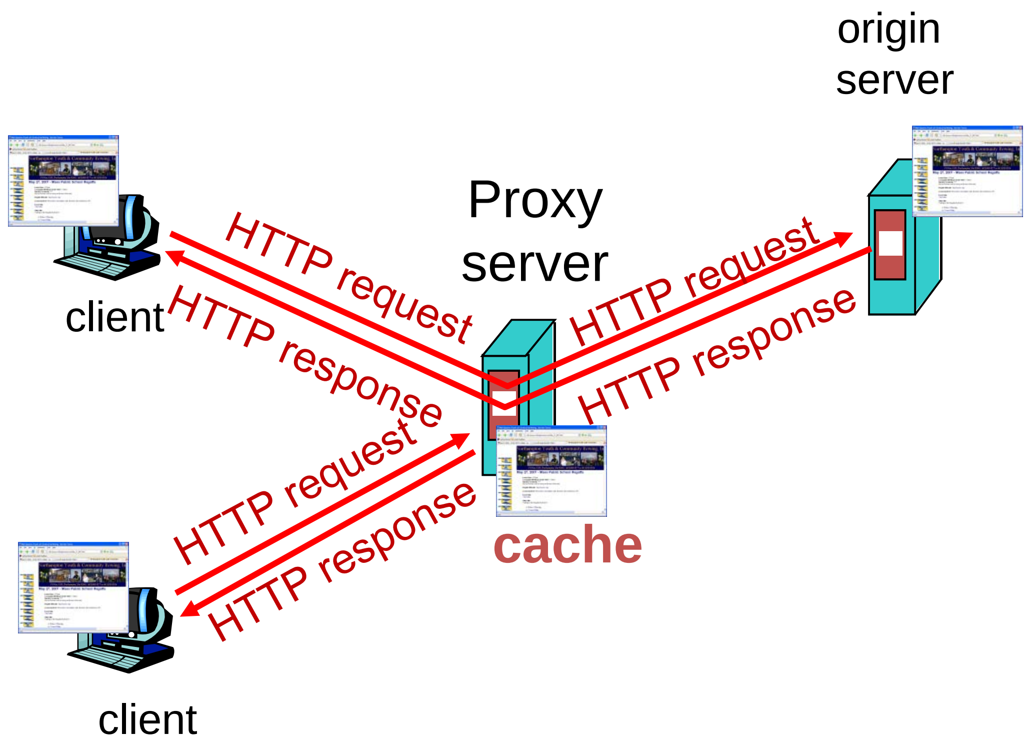
Cookies

- Cookies are transferred between server and client
- Cookie information stored at both client and server
- Maintain client information until deleted
- Cookies information size limited

Web Caches (Proxy Server)

Goal: satisfy client request without involving origin server

- User sets browser to access Web via cache ➔browser sends all HTTP requests to cache
  - **if object in cache,** cache returns object
  - **else** cache requests object from origin server, then returns object to client



More about Web Caching

- Cache acts as both client and server
- Typically cache is installed by ISP (university, company, residential ISP)
- Causes problems for frequently changing data though

<u>Why Web caching?</u>

- Reduce response time for client request
- Reduce traffic on an institution’s access link