Computación en Internet I

Andrés A. Aristizábal P.
aaaristizabal@icesi.edu.co

Departamento de Tecnologías de Información y Comunicaciones

UNIVERSIDAD ICESI

2023-1

Agenda

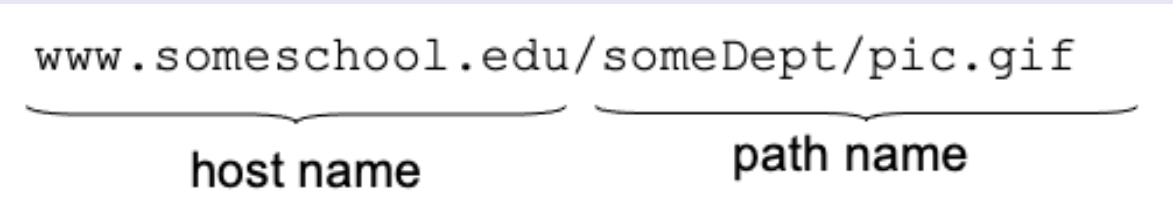
1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

Quick review

- Web page consists of objects, each of which can be stored on different Web servers.
- Object can be HTML file, JPEG image, Java applet, audio file.
- Web page consists of base HTML-file which includes several referenced objects, each addressable by a URL:



Overview

What is the hypertext transfer protocol (HTTP)?

What is the hypertext transfer protocol (HTTP)?

- 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

HTTP uses TCP

- Client initiates TCP connection (creates socket) to server, port 80.
- Server accepts TCP connection from client.
- Client/Server model:
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server).
  - The client sends request messages into its socket interface and receives response messages from its socket interface.
  - The server receives request messages into its socket interface and sends response messages from its socket interface.
  - Once the client/server sends a message into its socket interface, the message is the TCP's responsibility.
  - TCP provides a reliable data transfer service to HTTP.
  - Each message sent by client/server eventually arrives intact at the client/server.
- TCP connection closed.

HTTP is stateless

- Server maintains no information about past client requests.
- If a client asks for the same object twice in a period of a few seconds, the server does not respond by saying that it just served the object to the client, it resends it.

Overview

HTTP is stateless

- Server maintains no information about past client requests.
- If a client asks for the same object twice in a period of a few seconds, the server does not respond by saying that it just served the object to the client, it resends it.

Protocols that maintain state

- They are complex.
- Past history (state) must be maintained.
- If server/client crashes, their views of state may be inconsistent, must be reconciled.

Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

Connections

What type of connections does HTTP have?

Connections

What type of connections does HTTP have?

- Non-persistent HTTP.
- Persistent HTTP.

Non-persistent HTTP?

Non-persistent HTTP?

1. TCP connection opened.
2. At most one object sent over TCP connection.
3. TCP connection closed.

- Downloading multiple objects requires multiple connections.
- HTTP/1.0 employs non-persistent TCP connections.

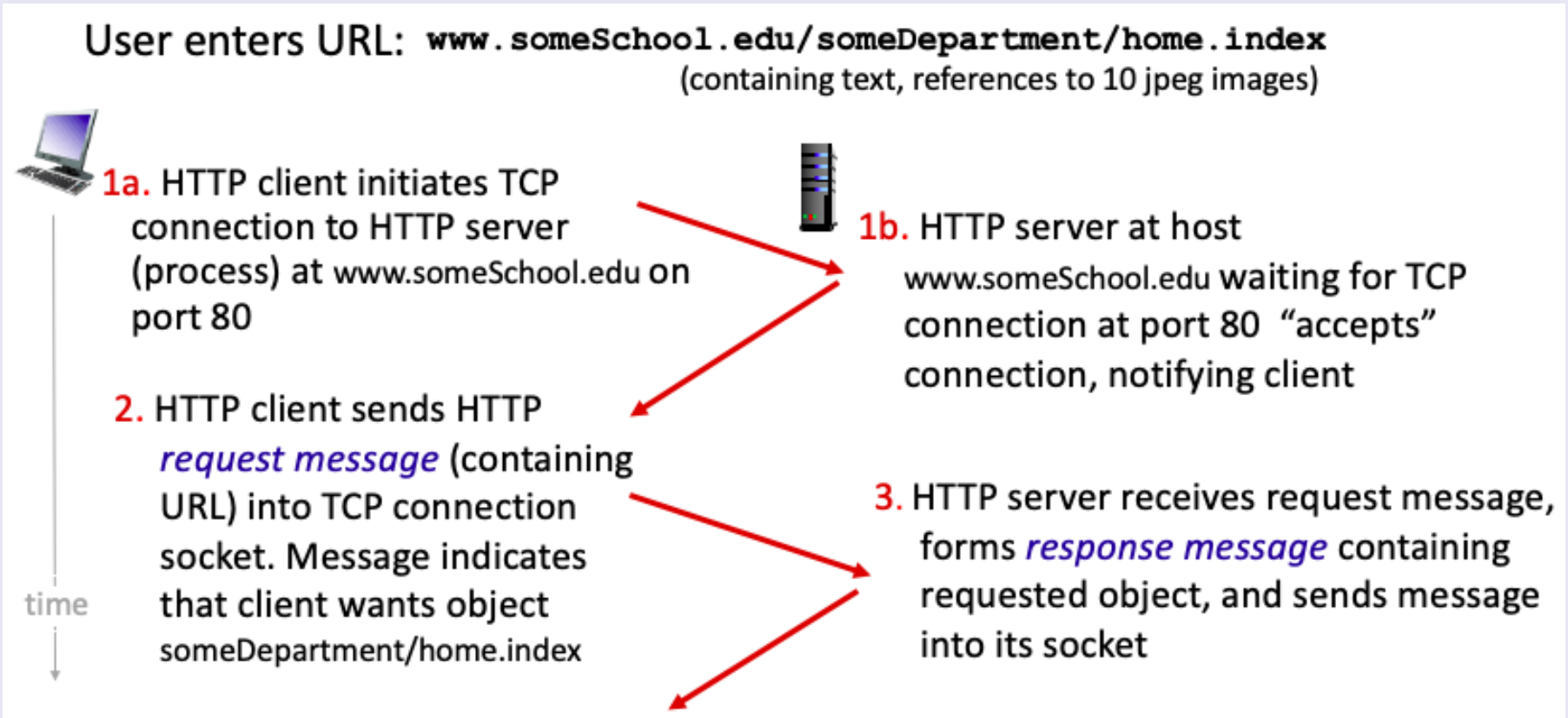
Persistent HTTP?

Persistent HTTP?

1. TCP connection opened to a server.
2. Multiple objects can be sent over single TCP connection between client, and that server.
3. TCP connection closed.

- HTTP/1.1 uses persistent TCP connections.

Non-persistent example



Non-persistent example

User enters URL: `www.someSchool.edu/someDepartment/home.index`
(containing text, references to 10 jpeg images)

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

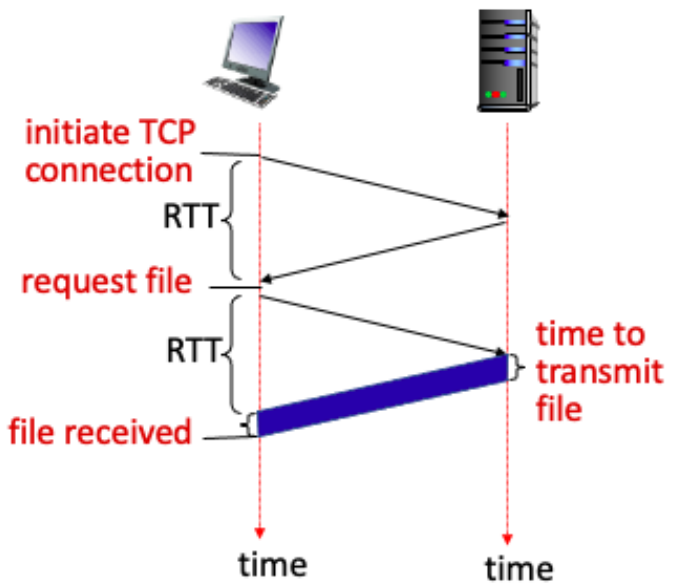
6. Steps 1-5 repeated for each of 10 jpeg objects

time

What is the Non-persistent response time?

What is the Non-persistent response time?

- RTT:
  - Time for a small packet to travel from client to server and back.
- HTTP response time (per object):
  - One RTT to initiate TCP connection.
  - One RTT for HTTP request and first few bytes of HTTP response to return.
  - Object/file transmission time.



Non-persistent HTTP response time = 2RTT+ file transmission time

What issues do Non-persistent connections have?

What issues do Non-persistent connections have?

- Significant burden on the web server.
  - A brand-new connection must be established and maintained for each requested object.
  - TCP buffers must be allocated and TCP variables must be kept in both the client and server.
  - The server may be serving requests from hundreds of different clients simultaneously.
- Each object suffers a delivery delay of two RTTs.

How to deal with these issues?

How to deal with these issues?

- Using persistent connections.
  - Server leaves connection open after sending response.
  - Subsequent HTTP messages between same client/server sent over open connection.
    - An entire Web page can be sent over a single connection
    - Web pages on same server can be sent to same client over a single connection.
  - Client sends requests as soon as it encounters a referenced object.
    - Without waiting for replies to pending requests (pipelining).
  - As little as one RTT for all the referenced objects (cutting response time in half).

Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

HTTP Message Format

What types of message does HTTP have?

HTTP Message Format

What types of message does HTTP have?

- Request
- Response

HTTP Message Format

What types of message does HTTP have?

- Request
- Response

What about the request message?

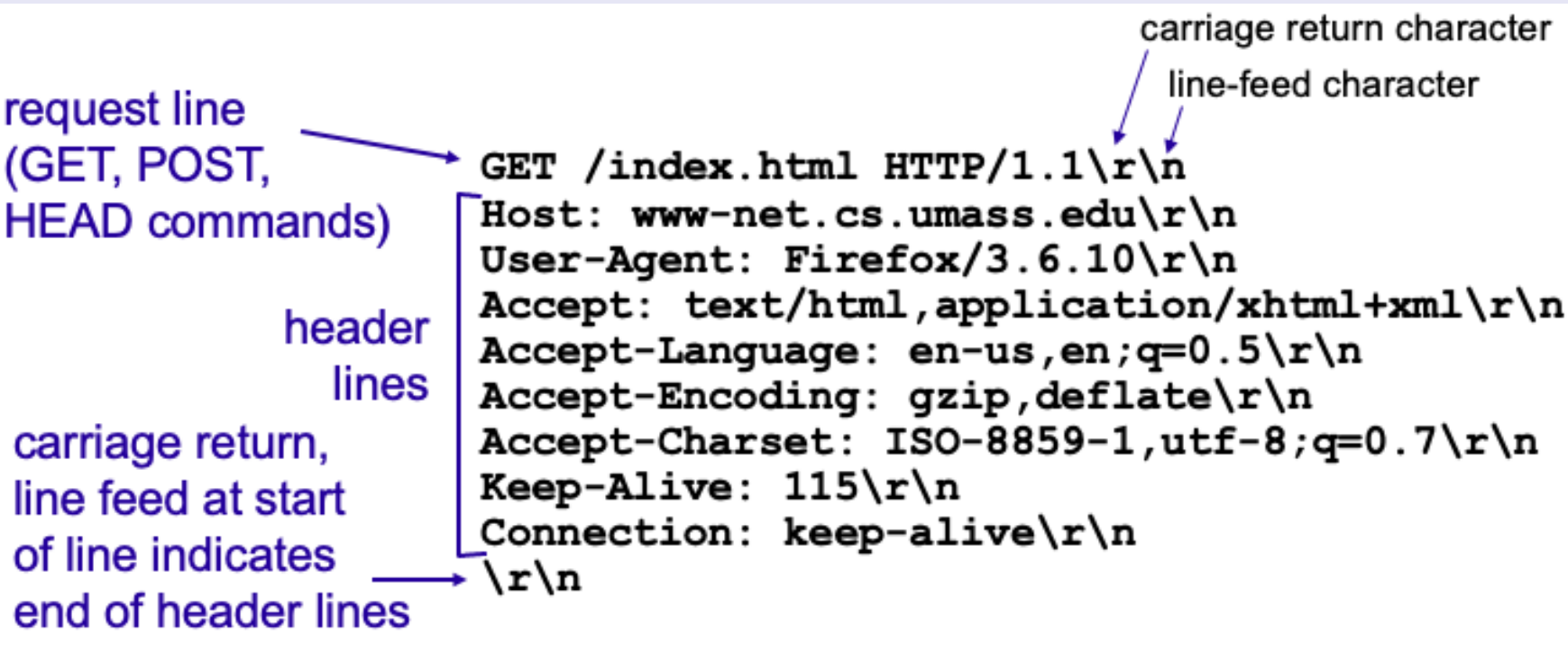
HTTP Message Format

What types of message does HTTP have?

- Request
- Response

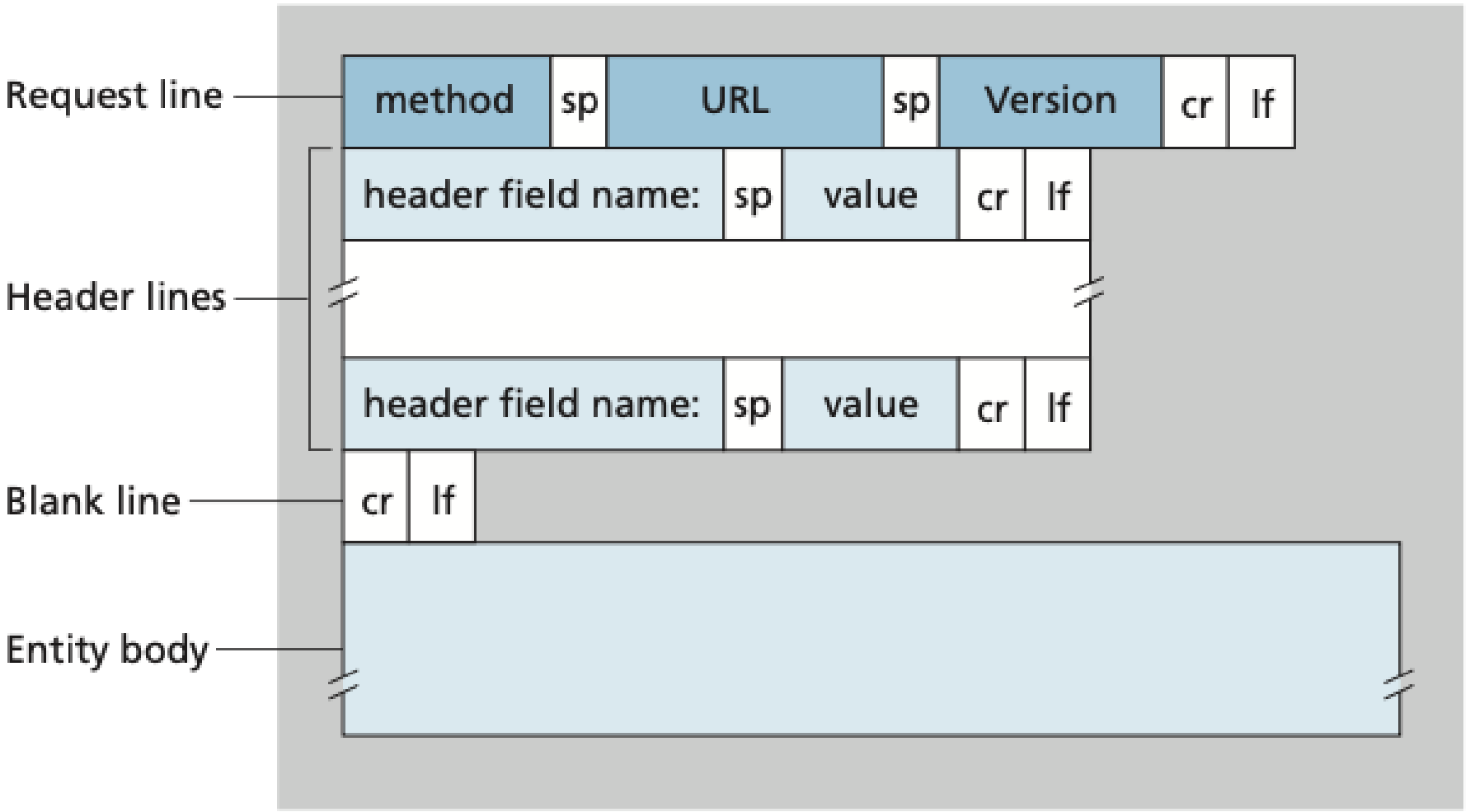
What about the request message?

- ASCII (human-readable format)



What is the general format of an HTTP request message?

What is the general format of an HTTP request message?



Which methods do request messages have?

Which methods do request messages have?

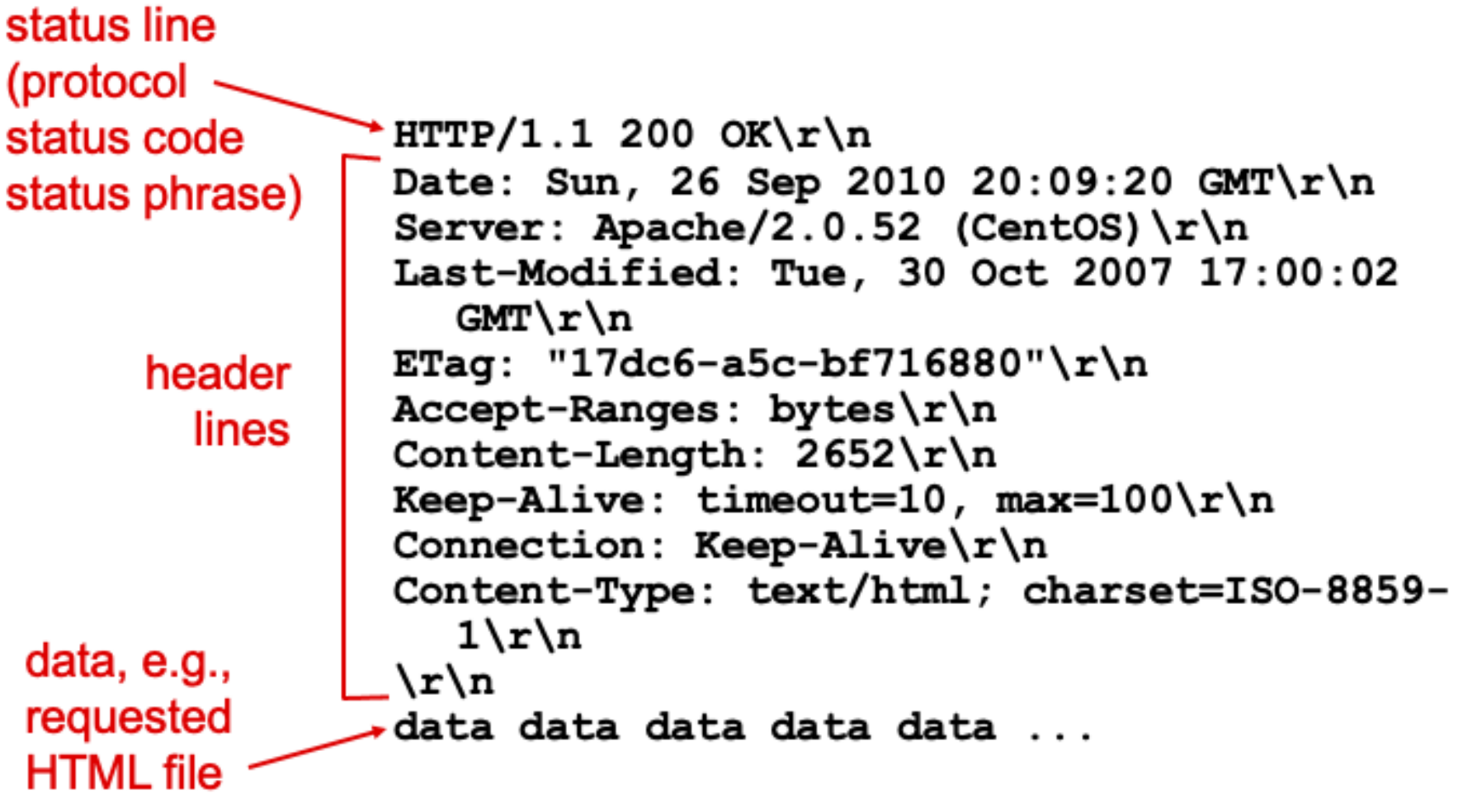
- POST method:
  - Web page often includes form input.
  - User input sent from client to server in entity body of HTTP POST request message.
- GET method (for sending data to server):
  - Include user data in URL field of HTTP GET request message (after a ?):
    - www.somesite.com/animalsearch?monkeys&banana

Which methods do request messages have?

- HEAD method:
  - Requests headers (only) that would be returned if specified URL were requested with an HTTP GET method.
- PUT method:
  - Uploads new file (object) to server.
  - Completely replaces file that exists at specified URL with content in entity body of POST HTTP request message.
- DELETE method:
  - Allows a user, or an application, to delete an object on a Web server.

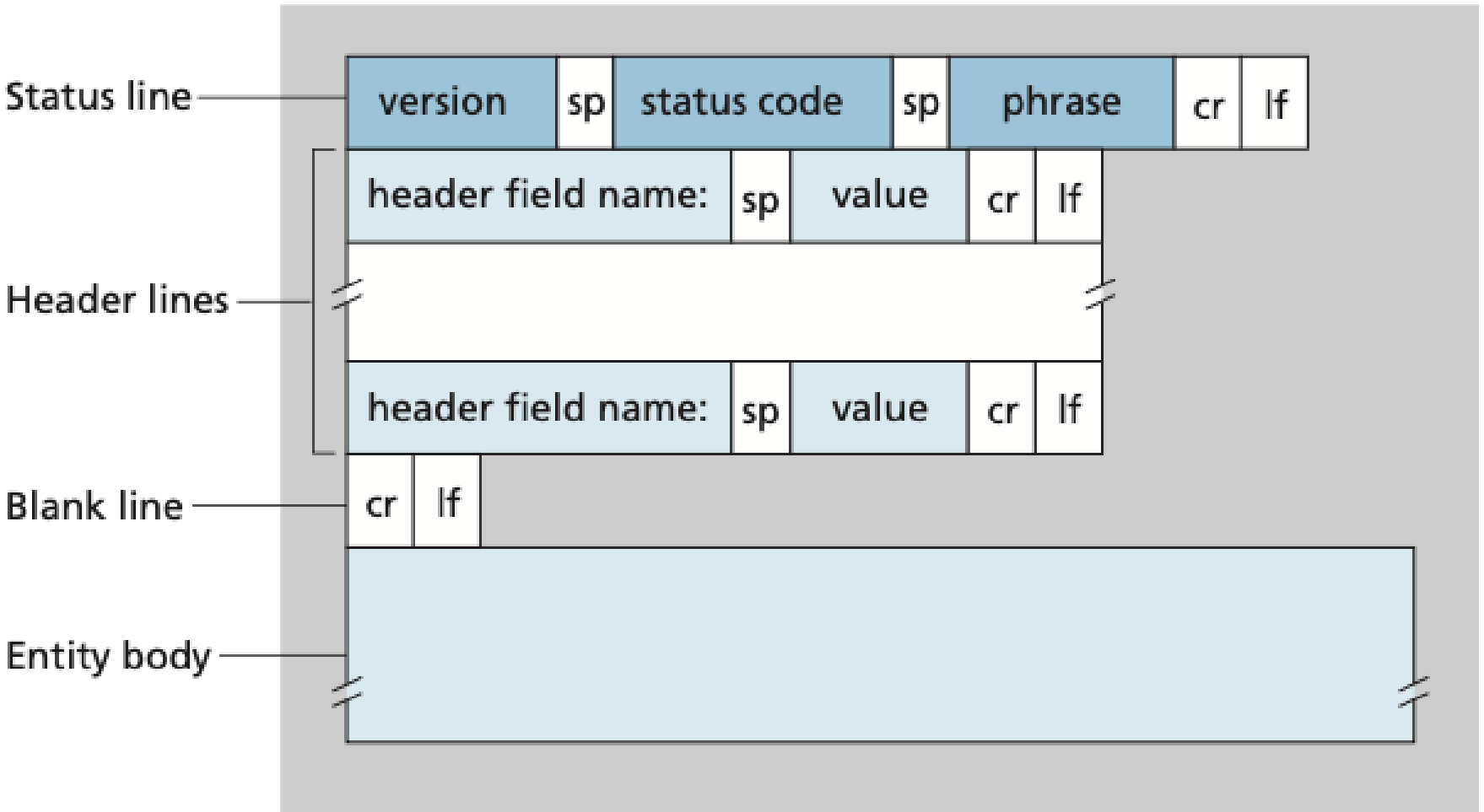
What would be an example of a response message?

What would be an example of a response message?



What is the general format of an HTTP response message?

What is the general format of an HTTP response message?



What about HTTP response status codes?

What about HTTP response status codes?

- Status code appears in 1st line in server-to-client response message.
- Some sample codes:
  - 200 OK: request succeeded, requested object later in this message.
  - 301 Moved Permanently: requested object moved, new location specified later in this message (in Location: field)
  - 400 Bad Request: request msg not understood by server
  - 404 Not Found: requested document not found on this server
  - 505 HTTP Version Not Supported

Trying out HTTP (client side) for yourself

Trying out HTTP (client side) for yourself

1. Netcat to your favorite Web server:
   - nc -c -v gaia.cs.umass.edu 80
   - Opens TCP connection to port 80 (default HTTP server port) at gaia.cs.umass. edu
   - Anything typed in will be sent to port 80 at gaia.cs.umass.edu
2. Type in a GET HTTP request:
   - GET /kurose_ross/interactive/index.php HTTP/1.1
     Host: gaia.cs.umass.edu
   - By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to HTTP server
3. Look at response message sent by HTTP server
   - Or use Wireshark to look at captured HTTP request/response

Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

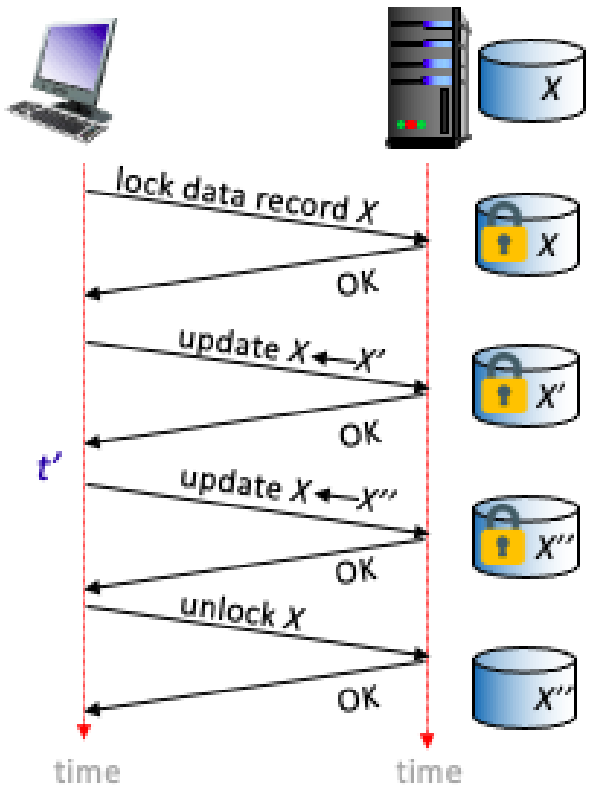
User-Server Interaction: Cookies

Maintaining user/server state

- HTTP GET/response interaction is stateless.
- No notion of multi-step exchanges of HTTP messages to complete a Web transaction.
  - No need for client/server to track state of multi-step exchange.
  - All HTTP requests are independent of each other.
  - No need for client/server to recover from a partially-completed-but-never-completely-completed transaction.

Maintaining user/server state

a stateful protocol: client makes two changes to X, or none at all



Q: what happens if network connection or client crashes at t' ?

User-Server Interaction: Cookies

How are web sites able to maintain some sort of state between transactions?

User-Server Interaction: Cookies

How are web sites able to maintain some sort of state between transactions?

- By means of cookies.

User-Server Interaction: Cookies

How are web sites able to maintain some sort of state between transactions?

- By means of cookies.

Which are their four components?

User-Server Interaction: Cookies

How are web sites able to maintain some sort of state between transactions?

- By means of cookies.

Which are their four components?

- Cookie header line of HTTP response message.
- Cookie header line in next HTTP request message
- Cookie file kept on user's host, managed by user's browser.
- Back-end database at Web site.

User-Server Interaction: Cookies

Example?

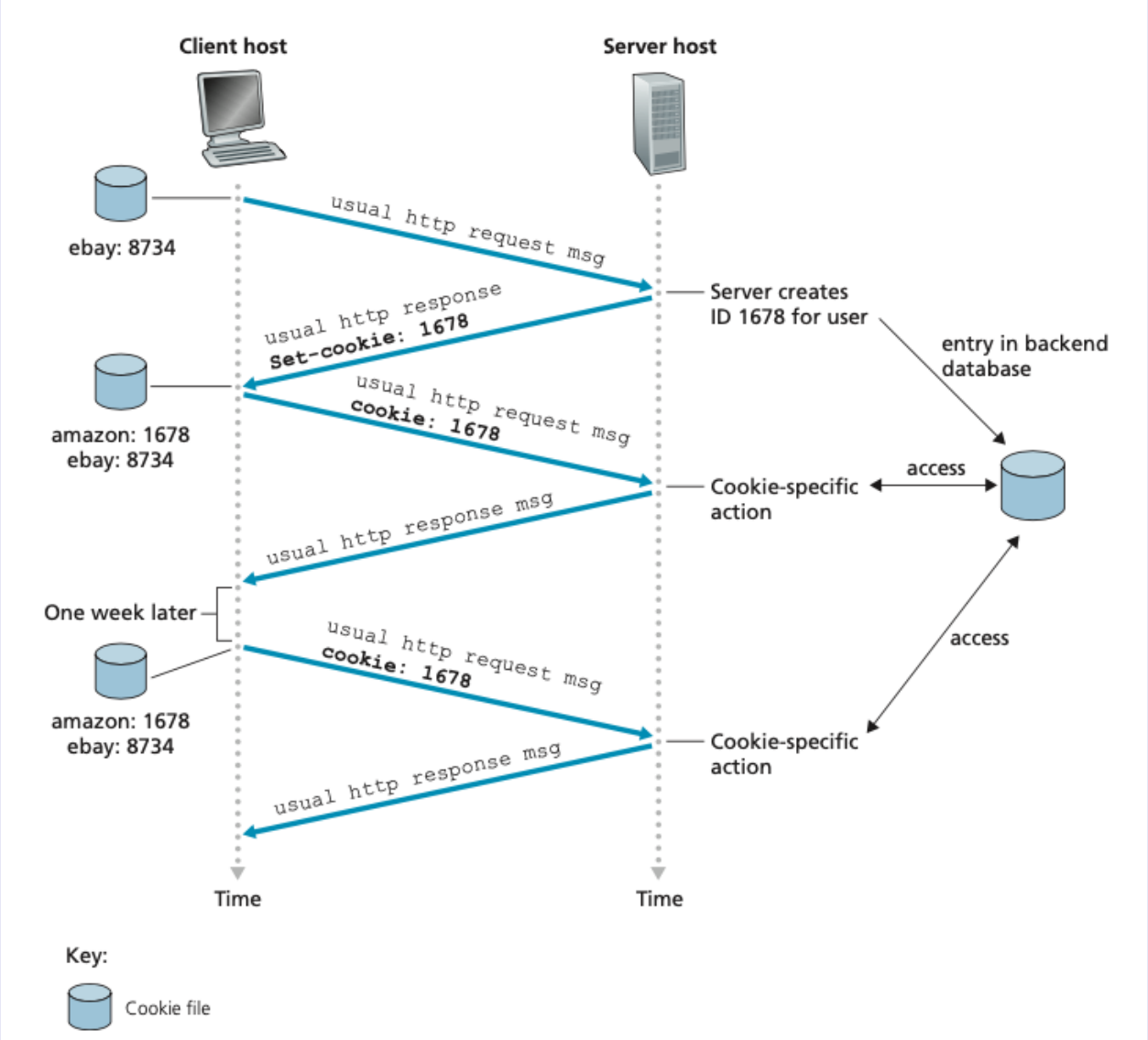
User-Server Interaction: Cookies

Example?

- Susan uses browser on laptop, visits specific e-commerce site for first time.
- When initial HTTP requests arrives at site, site creates:
  - Unique ID (aka cookie).
  - Entry in backend database for ID.
- Subsequent HTTP requests from Susan to this site will contain cookie ID value, allowing site to identify Susan.

User-Server Interaction: Cookies

A more detailed example?



User-Server Interaction: Cookies

For what can cookies used for?

User-Server Interaction: Cookies

For what can cookies used for?

- Authorization.
- Shopping carts
- Recommendations
- User session state (Web e-mail)

User-Server Interaction: Cookies

For what can cookies used for?

- Authorization.
- Shopping carts
- Recommendations
- User session state (Web e-mail)

Which are their privacy issues?

User-Server Interaction: Cookies

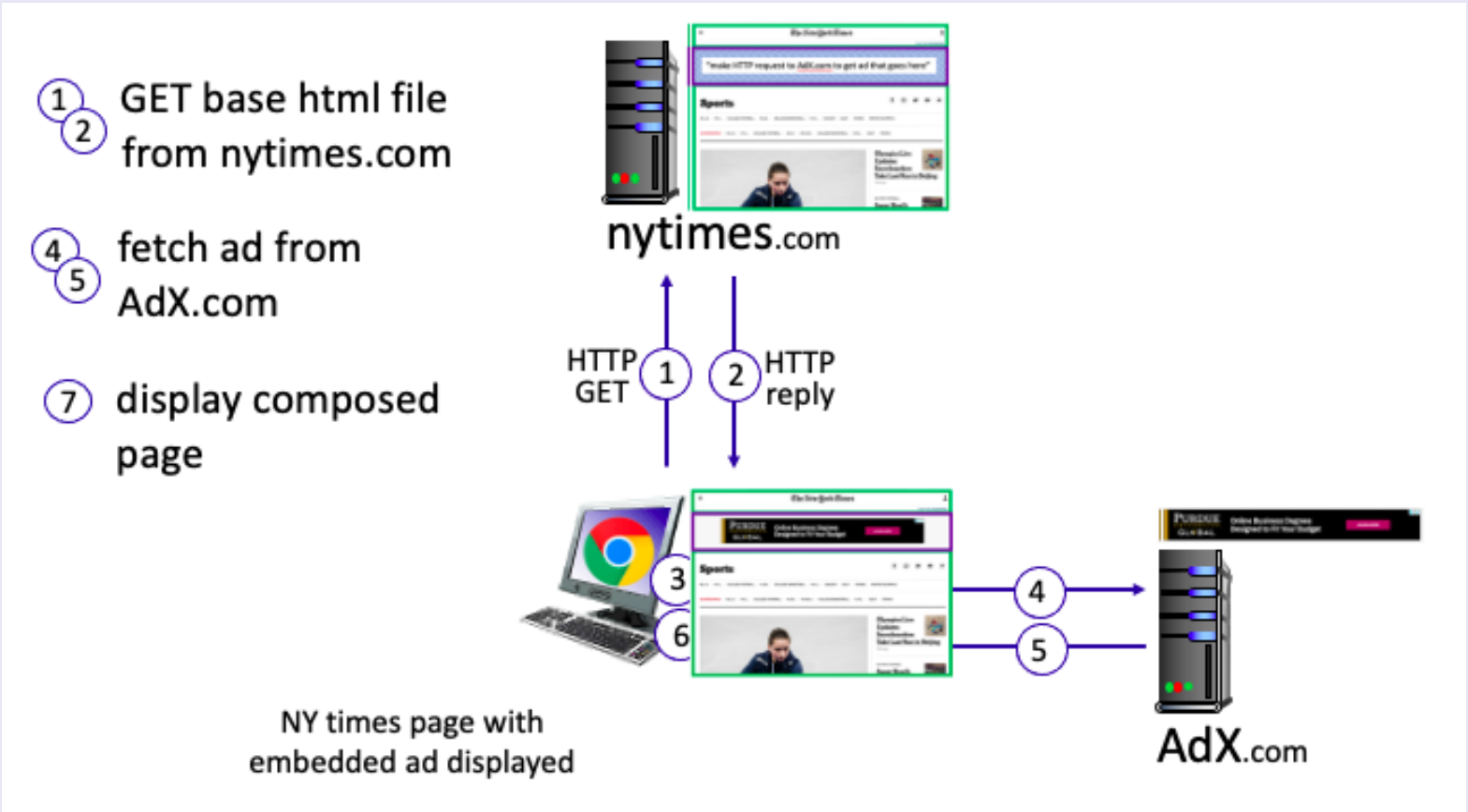
For what can cookies used for?

- Authorization.
- Shopping carts
- Recommendations
- User session state (Web e-mail)

Which are their privacy issues?

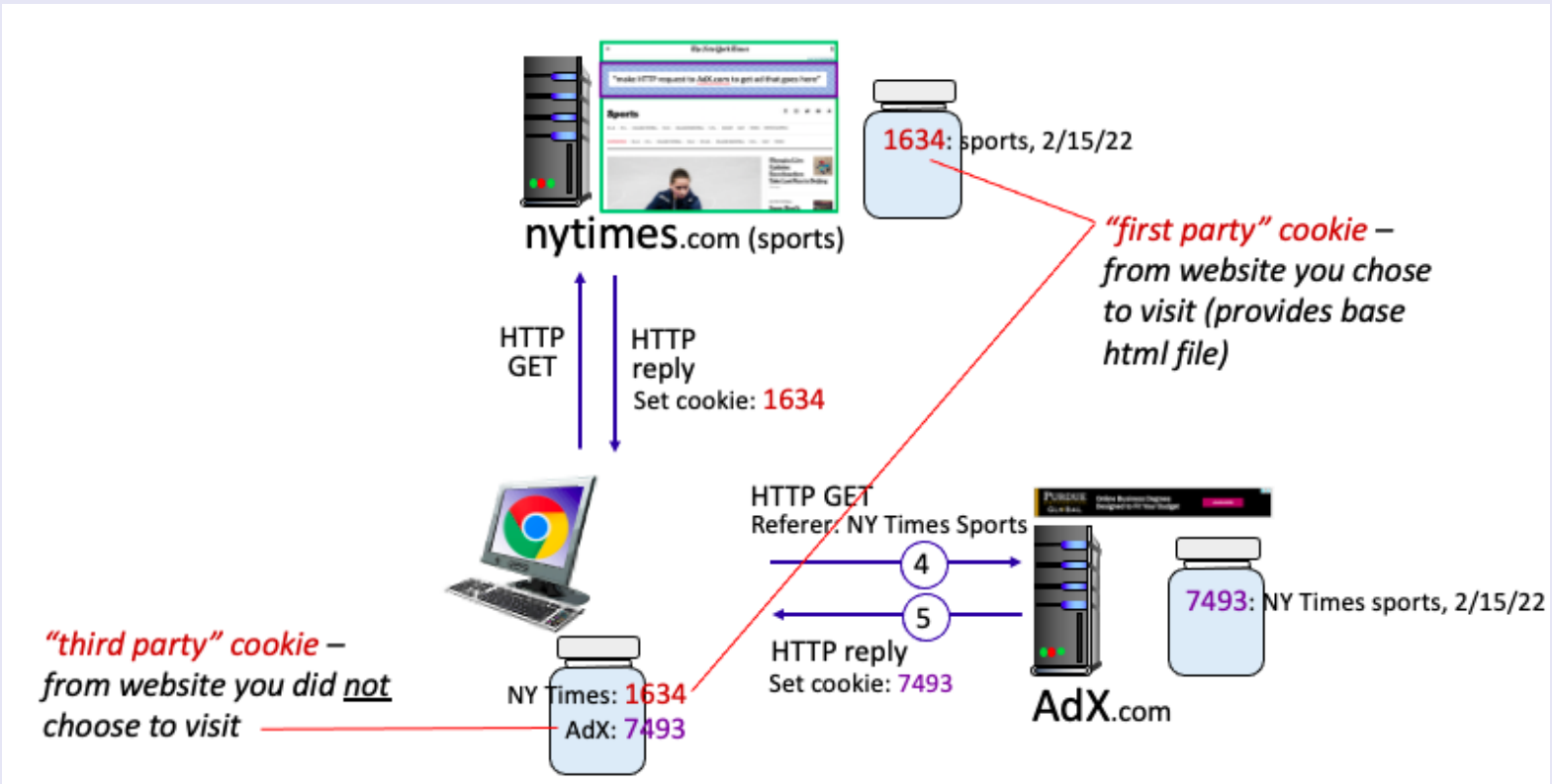
- Cookies permit sites to learn a lot about you on their site.
- Third party persistent cookies (tracking cookies) allow common identity (cookie value) to be tracked across multiple web sites.

Tracking a user's browsing behavior



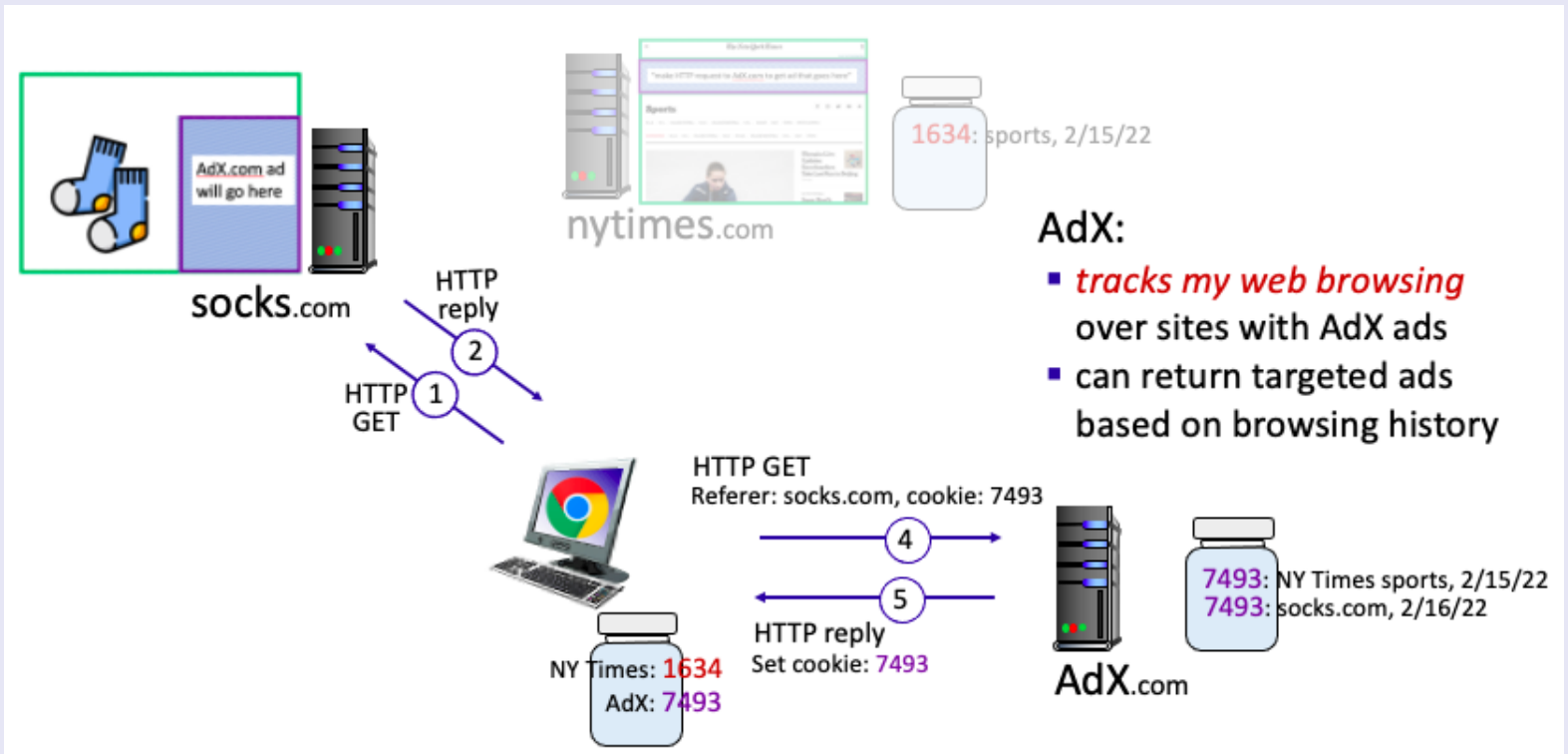
User-Server Interaction: Cookies

Tracking a user's browsing behavior



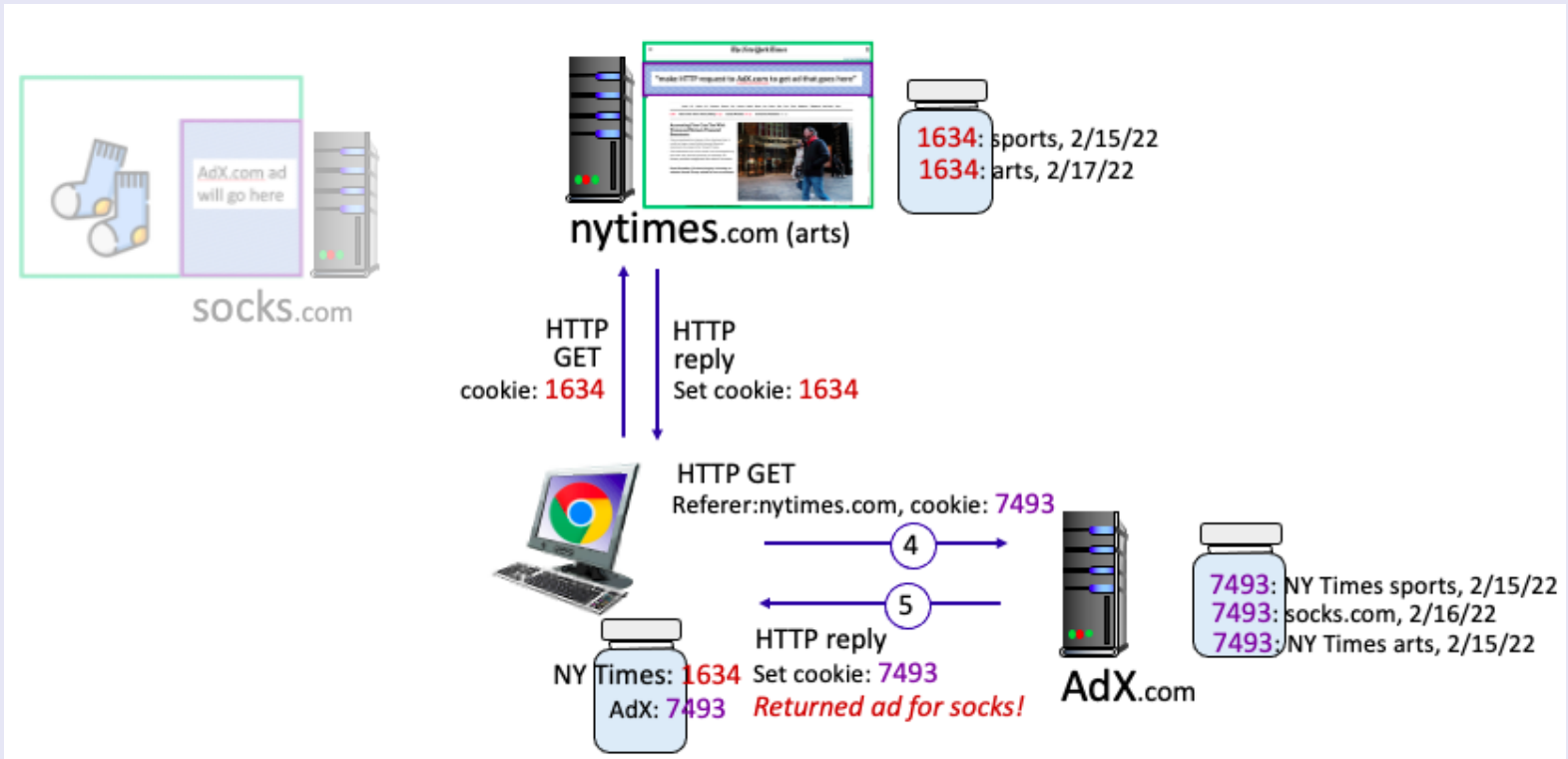
User-Server Interaction: Cookies

Tracking a user's browsing behavior



User-Server Interaction: Cookies

Tracking a user's browsing behavior (one day later)



User-Server Interaction: Cookies

Cookies can be used to:

- Track user behavior on a given website (first party cookies).
- Track user behavior across multiple websites (third party cookies) without user ever choosing to visit tracker site.
- Tracking may be invisible to user:
  - Rather than displayed ad triggering HTTP GET to tracker, could be an invisible link.
- Third party tracking via cookies:
- Disabled by default in Firefox, Safari browsers.
- To be disabled in Chrome browser in 2023.

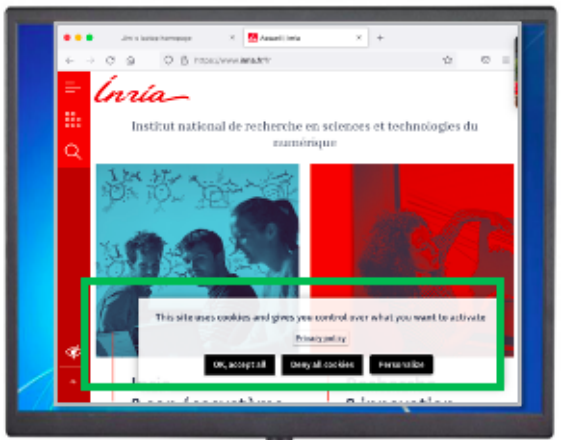
EU General Data Protection Regulation (GDPR)

"Natural persons may be associated with online identifiers [...] such as internet protocol addresses, cookie identifiers or other identifiers [...].

This may leave traces which, in particular when combined with unique identifiers and other information received by the servers, may be used to create profiles of the natural persons and identify them."

GDPR, recital 30 (May 2018)

when cookies can identify an individual, cookies are considered personal data, subject to GDPR personal data regulations



User has explicit control over whether or not cookies are allowed

Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

Web cache

What is the goal of web cache?

Web cache

What is the goal of web cache?

- Proxy servers are used to satisfy client requests without involving origin server.

Web cache

What is the goal of web cache?

- Proxy servers are used to satisfy client requests without involving origin server.

How does it work?

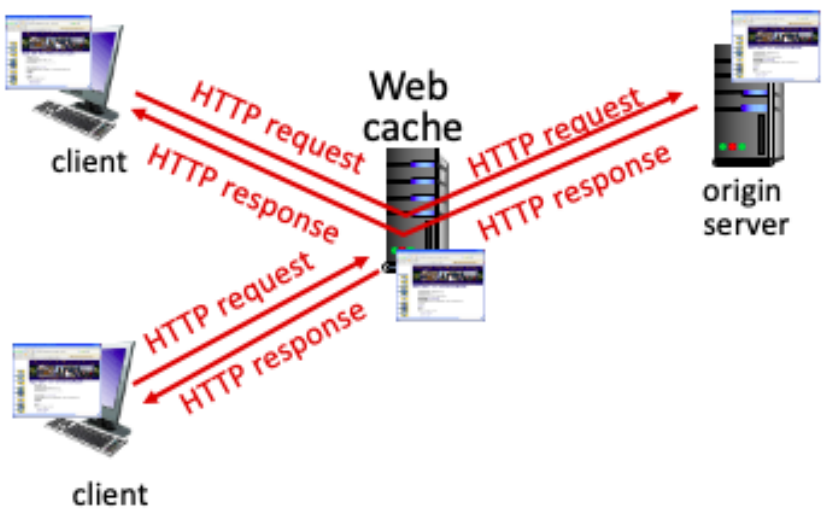
Web cache

What is the goal of web cache?

- Proxy servers are used to satisfy client requests without involving origin server.

How does it work?

- Browser sends all HTTP requests to cache.
- If object in cache: cache returns object to client.
- Else cache requests object from origin server, caches received object, then returns object to client.



How does it work?

- Web cache acts as both client and server.
- Server for original requesting client.
- Client to origin server.
  - Server tells cache about object's allowable caching in response header:

```
Cache-Control: max-age=<seconds>
```

```
Cache-Control: no-cache
```

Why web caching?

Why web caching?

- Reduce response time for client request.
  - Cache is closer to client.
- Reduces traffic on an institution's access link.
- Internet is dense with caches.
  - Enables poor content providers to more effectively deliver content.

Web cache

What can be a problem of web caching?

What can be a problem of web caching?

- The copy of an object residing in the cache may be stale.
- The object housed in the Web server may have been modified since the copy was cached at the client.

Web cache

What can be a problem of web caching?

- The copy of an object residing in the cache may be stale.
- The object housed in the Web server may have been modified since the copy was cached at the client.

How to deal with this?

What can be a problem of web caching?

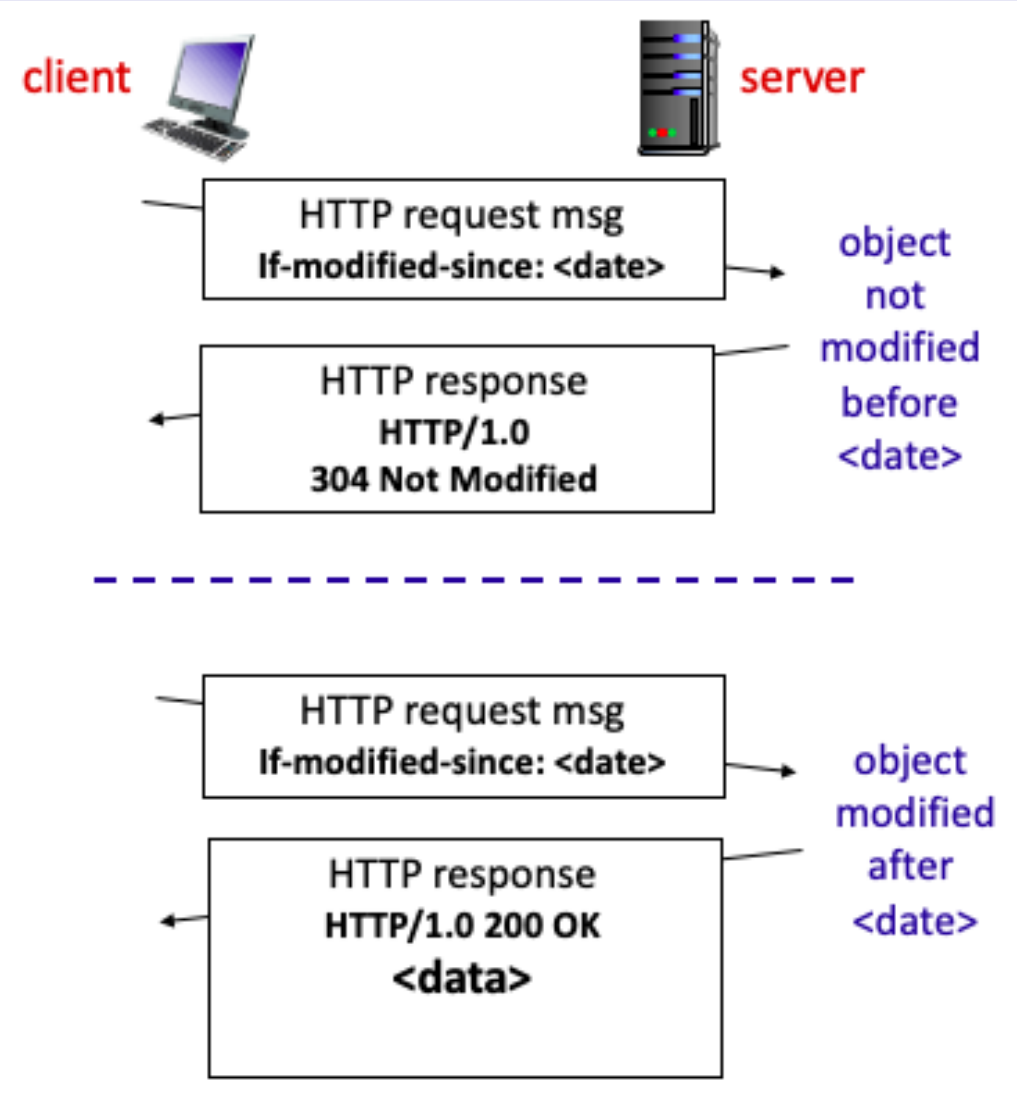
- The copy of an object residing in the cache may be stale.
- The object housed in the Web server may have been modified since the copy was cached at the client.

How to deal with this?

- HTTP has a mechanism that allows a cache to verify that its objects are up to date.
- This mechanism is called the conditional GET.

How does it work?

How does it work?



Agenda del día

1. **HTTP**
   - Overview
   - Connections
   - HTTP Message Format
   - User-Server Interaction: Cookies
   - Web cache
   - HTTP/2
2. **Workshop**

What is HTTP/2?

What is HTTP/2?

- The first new version of HTTP since HTTP/1.1.
- Standardized in 1997.
- Over 40 % of the top 10 million websites supported HTTP/2 by 2020.
- Most browsers (Google Chrome, Internet Explorer, Safari, Opera, and Firefox).

What is HTTP/2?

- The first new version of HTTP since HTTP/1.1.
- Standardized in 1997.
- Over 40 % of the top 10 million websites supported HTTP/2 by 2020.
- Most browsers (Google Chrome, Internet Explorer, Safari, Opera, and Firefox).

What are its goals?

What is HTTP/2?

- The first new version of HTTP since HTTP/1.1.
- Standardized in 1997.
- Over 40 % of the top 10 million websites supported HTTP/2 by 2020.
- Most browsers (Google Chrome, Internet Explorer, Safari, Opera, and Firefox).

What are its goals?

- Decreased delay in multi-object HTTP requests.
- Increased flexibility at server in sending objects to client:
  - Methods, status codes, most header fields unchanged from HTTP 1.1.
  - Transmission order of requested objects based on client-specified object priority (not necessarily FCFS).
  - Push unrequested objects to client.
  - Divide objects into frames, schedule frames to mitigate HOL blocking.

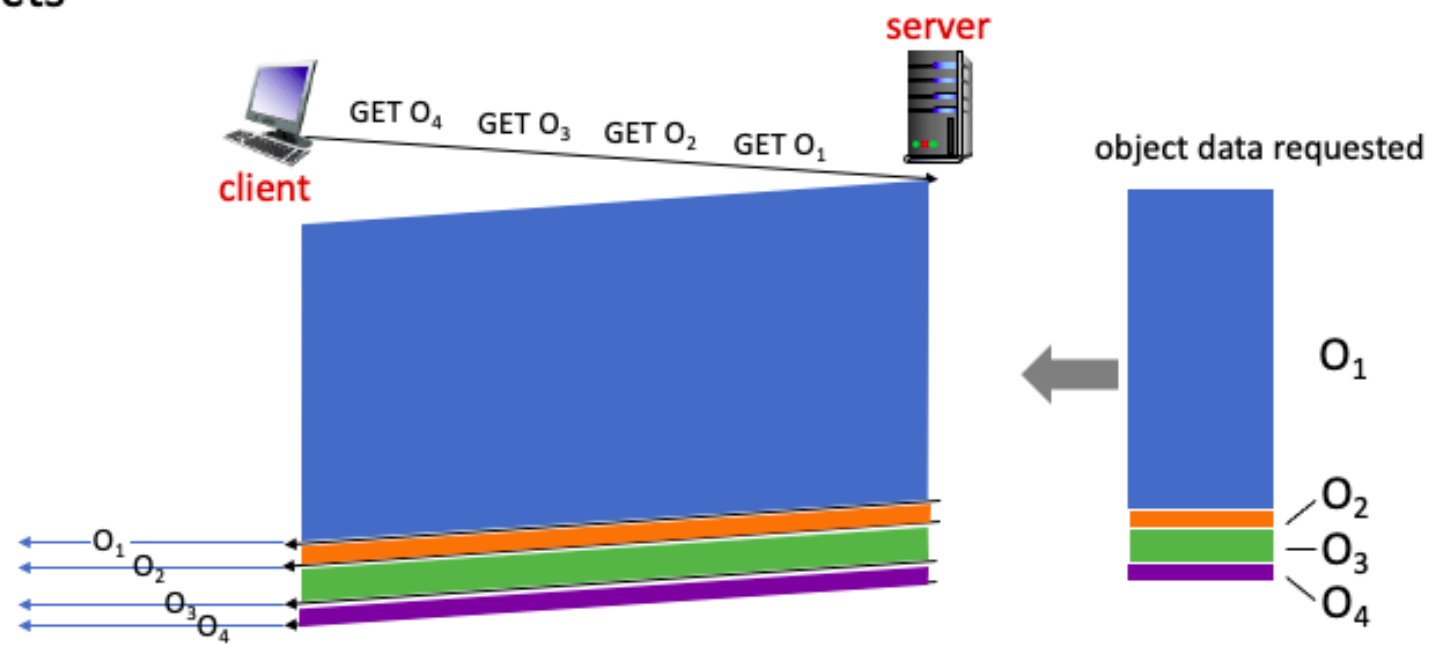
What is the Head of Line (HOL) blocking problem?

What is the Head of Line (HOL) blocking problem?

- Recall that HTTP/1.1 uses persistent TCP connections, allowing a Web page to be sent from server to client over a single TCP connection.
- By having only one TCP connection per Web page, the number of sockets at the server is reduced.
- But sending all the objects in a Web page over a single TCP connection has a Head of Line (HOL) blocking problem.
- A performance-limiting phenomenon that occurs when a line of packets is held up in a queue by a first packet.

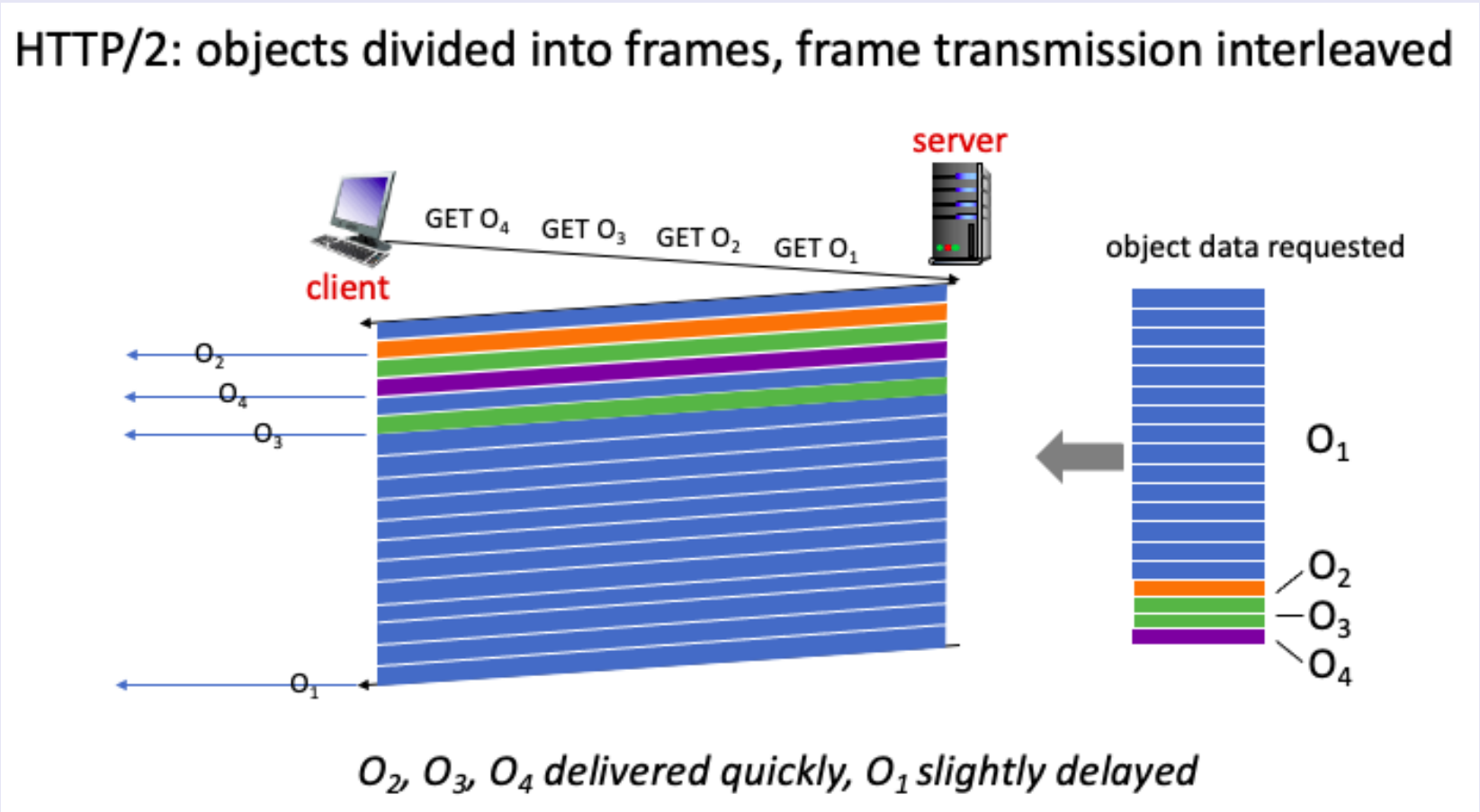
HOL

HTTP 1.1: client requests 1 large object (e.g., video file) and 3 smaller objects



objects delivered in order requested: O_2, O_3, O_4 wait behind O_1

HOL


HTTP/2: objects divided into frames, frame transmission interleaved
server
client
GET O4
GET O3
GET O2
GET O1
object data requested
O2
O4
O3
O1
O1
O2
O3
O4
O2, O3, O4 delivered quickly, O1 slightly delayed

Workshop

Complete workshop for today's class. To be handed in at the end of the class.