CS 372 Lecture #11

The Application Layer:

- More HTTP
- Cookies
- Caching

Note: Many of the lecture slides are based on presentations that accompany *Computer Networking: A Top Down Approach,* 6th edition, by Jim Kurose & Keith Ross, Addison-Wesley, 2013.

Client-server state: cookies

Many major Web sites use cookies

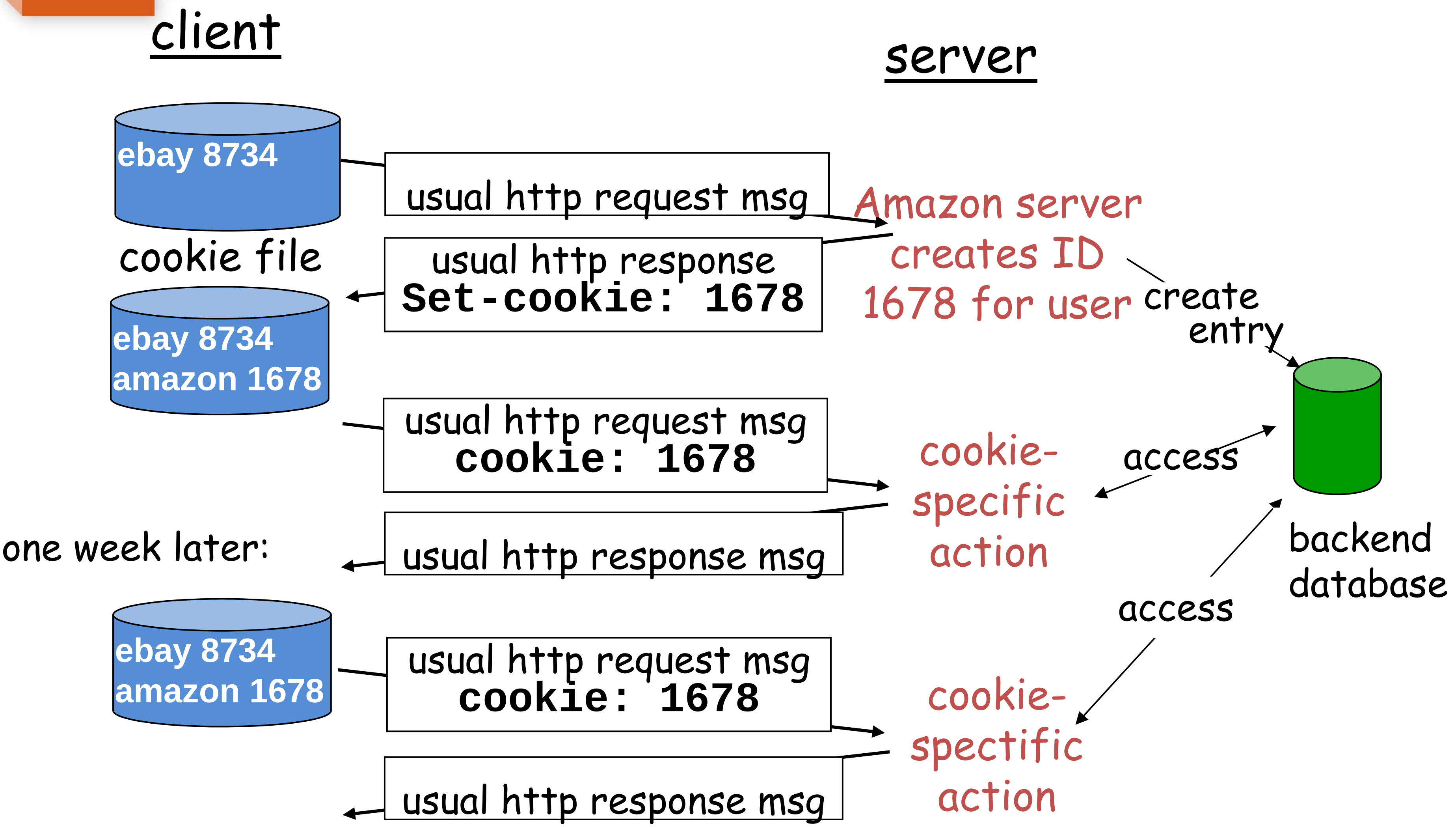
Four components:

1) cookie header line of HTTP *response* message
2) cookie header line in HTTP *request* message
3) cookie file kept on user's host, managed by user's browser
4) back-end database at Web site

Example:

- User visits specific e-commerce site for first time
- when initial HTTP requests arrives at site, site creates:
  - unique ID
  - entry in backend database for ID

Cookies: keeping "state"



Cookies (cont.)

What cookies can provide:

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

aside

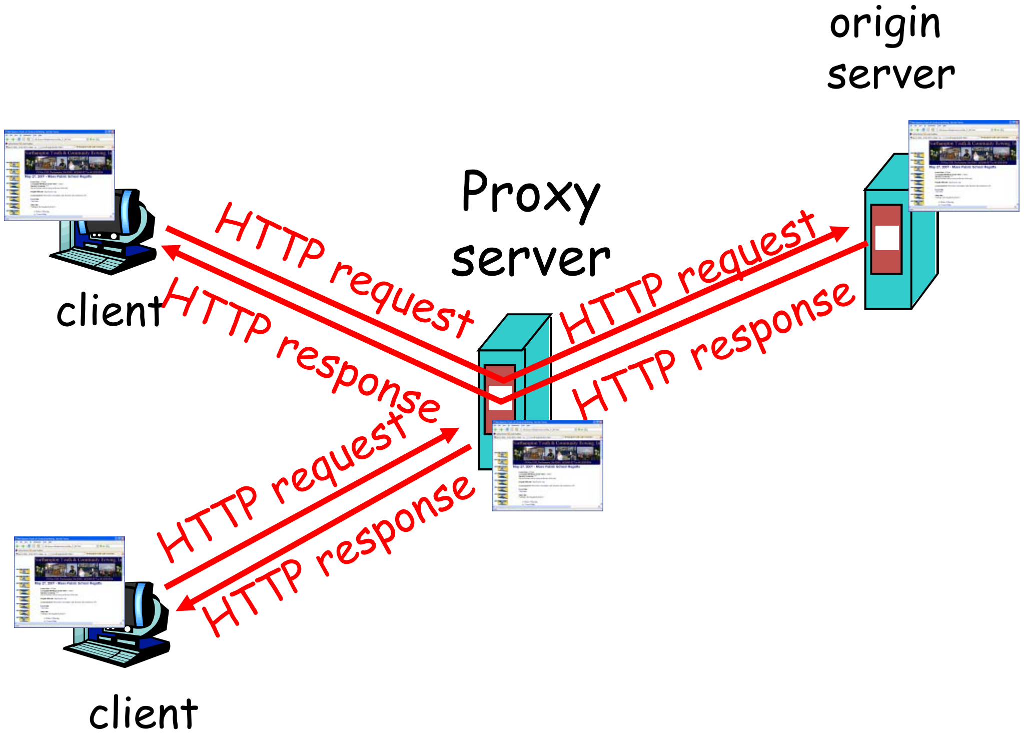
Cookies and privacy:

- cookies permit sites to learn a lot about you
- you may be giving your name and e-mail to sites

Web caches (proxy server)

Goal: satisfy client request without involving origin server

- User's 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)
- Cached objects have “expiration” date/time

Why Web caching?

- reduce response time for client request
- reduce traffic on an institution’s access link.
- enables “poor” providers to effectively deliver content

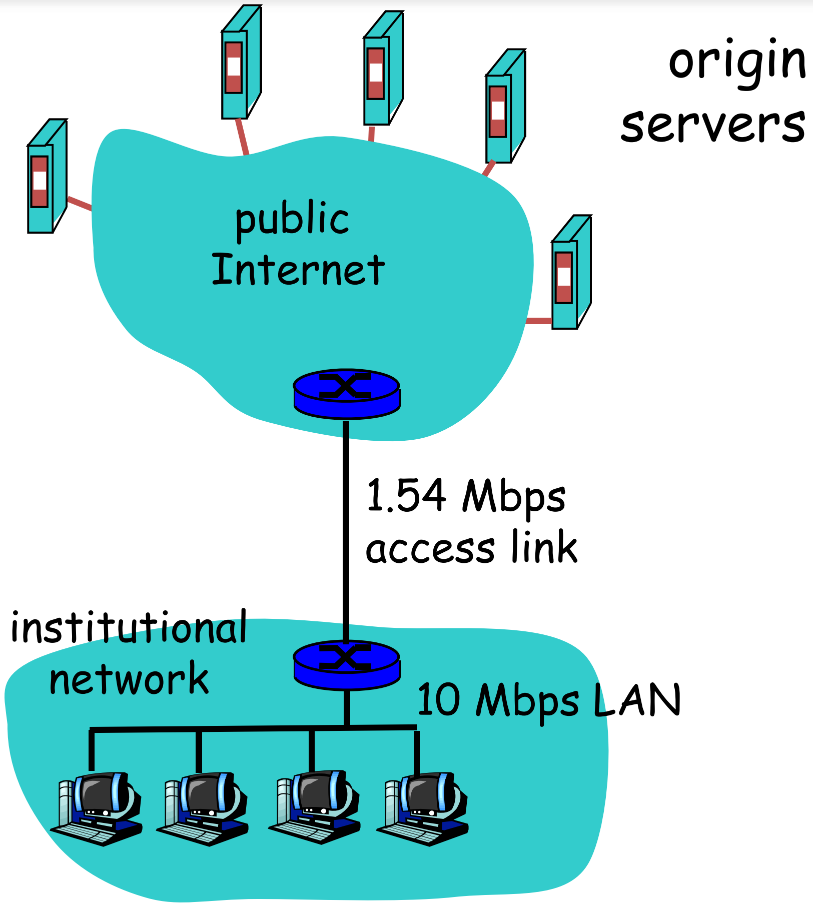
Example (no caching)

Assumptions

- average object size = 100K bits
- average request rate from institution's browsers to origin servers = 15 requests per second
- delay from institutional router to any origin server and back to router = 2 seconds

Consequences

- utilization on LAN = 15%
- utilization on access link = ~100%



total average delay = Internet delay + access delay + LAN delay
= 2 seconds + minutes + milliseconds

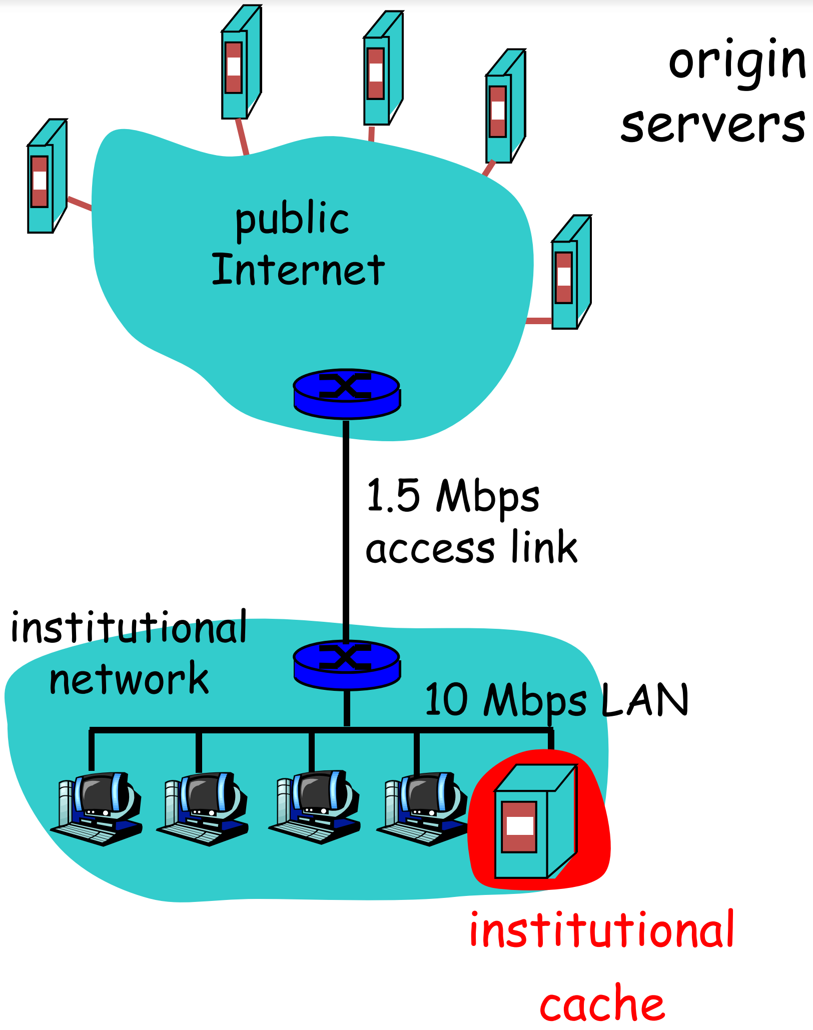
Example (caching)

Same assumptions

- … but with caching
- Suppose cache hit rate is 0.4
  - i.e. only 60% of requests go to origin servers

Consequences

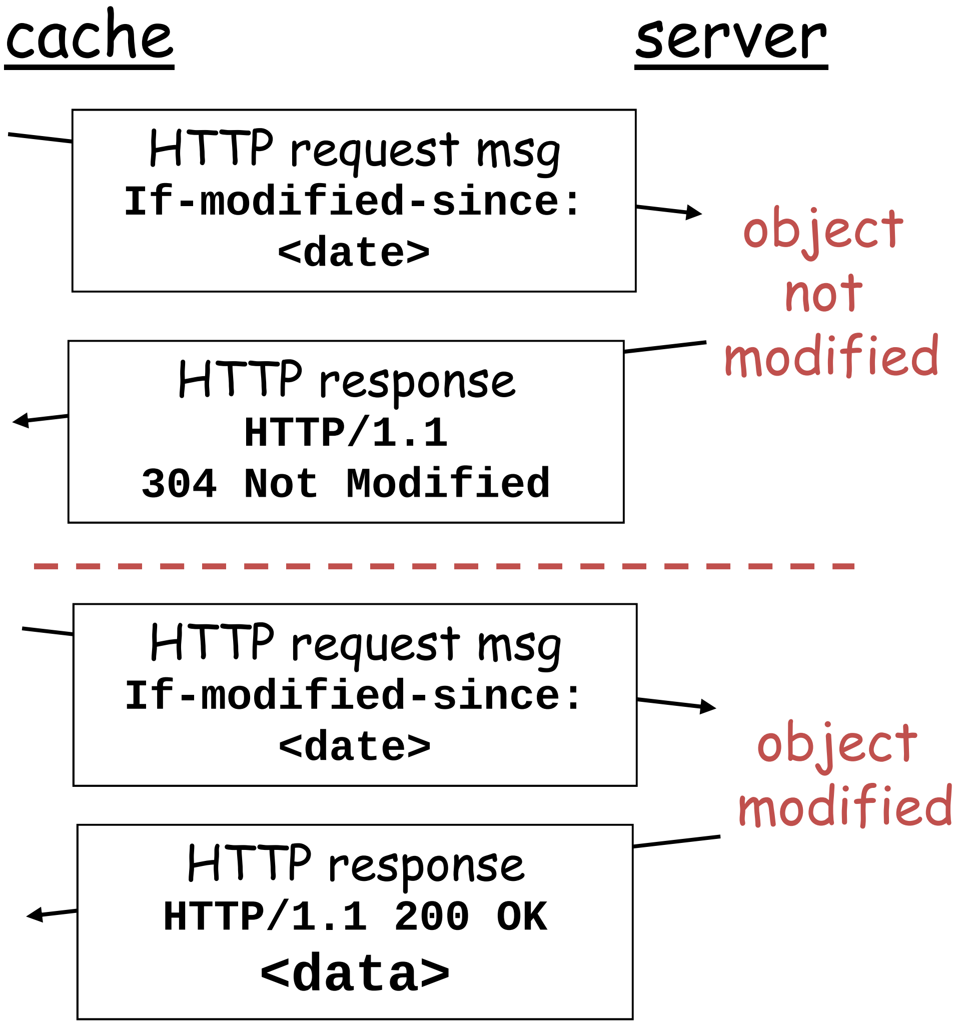
- 40% of requests will be satisfied almost immediately
- utilization of access link is reduced to 60%, resulting in negligible delays (say 10 ms) because of no congestion



total average delay = Internet delay + access delay + LAN delay
= 0.6*(2.0) seconds + 0.4*10 <u>milliseconds</u> < 1.4 seconds

Conditional GET

- Goal: don't send object if cache has up-to-date cached version
- cache: specify date of cached copy in HTTP request
  If-modified-since: <date>
- server: response contains no object if cached copy is up-to-date:
  HTTP/1.1 304 Not Modified



cache

server

HTTP request msg
If-modified-since: <date>

HTTP response
HTTP/1.1
304 Not Modified

object not modified

HTTP request msg
If-modified-since: <date>

HTTP response
HTTP/1.1 200 OK
<data>

object modified

Summary Lecture #11

- Definitions
  - Cookie
  - Caching
- HTTP
  - Conditional GET