COSC264
Introduction to Computer Networks and the Internet

HTTP Basics (continued)

Dr. Barry Wu
Wireless Research Centre
University of Canterbury
barry.wu@canterbury.ac.nz

Outline

- HTTP
  - Non-persistent and persistent connections
  - Messages
  - Cookies
  - Web caching

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

Nonpersistent HTTP

Suppose user enters URL

`www.someSchool.edu/someDepartment/home.index`

(contains text, references to 10 jpeg images)

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

2. HTTP client sends 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

Nonpersistent HTTP (cont.)

4. HTTP server process tells TCP to close the TCP connection.
   (It doesn't actually close it until it knows for sure the client has received the response message.)

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

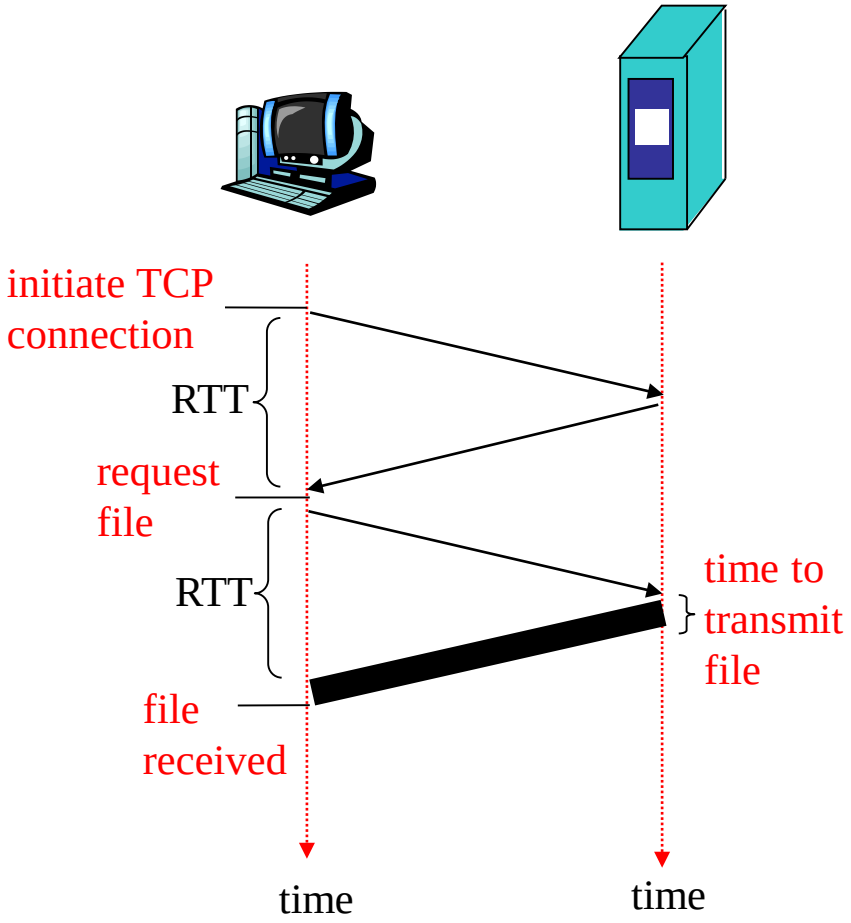
Response time modeling

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



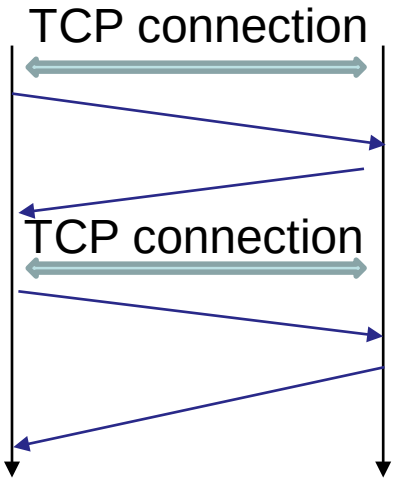
Persistent HTTP

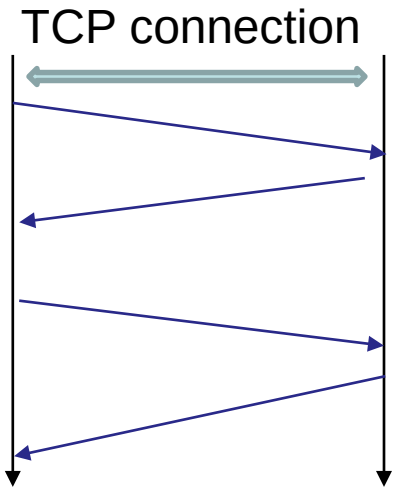
Nonpersistent HTTP issues:

- requires 2 RTTs per object
- OS must work and allocate host resources for each TCP connection
- *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 the same connection





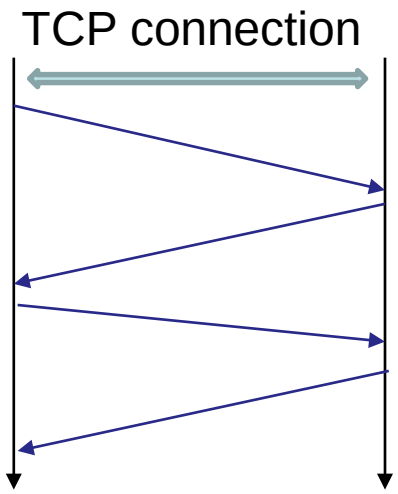
Persistent HTTP + pipelining

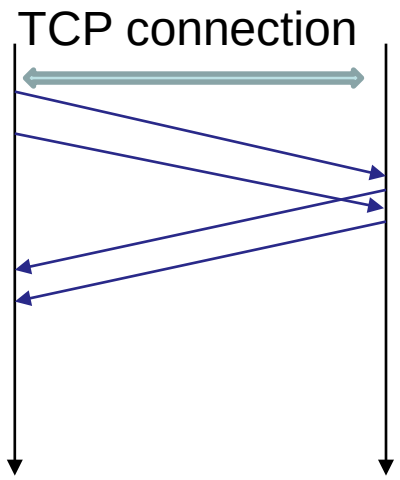
Persistent without pipelining:

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

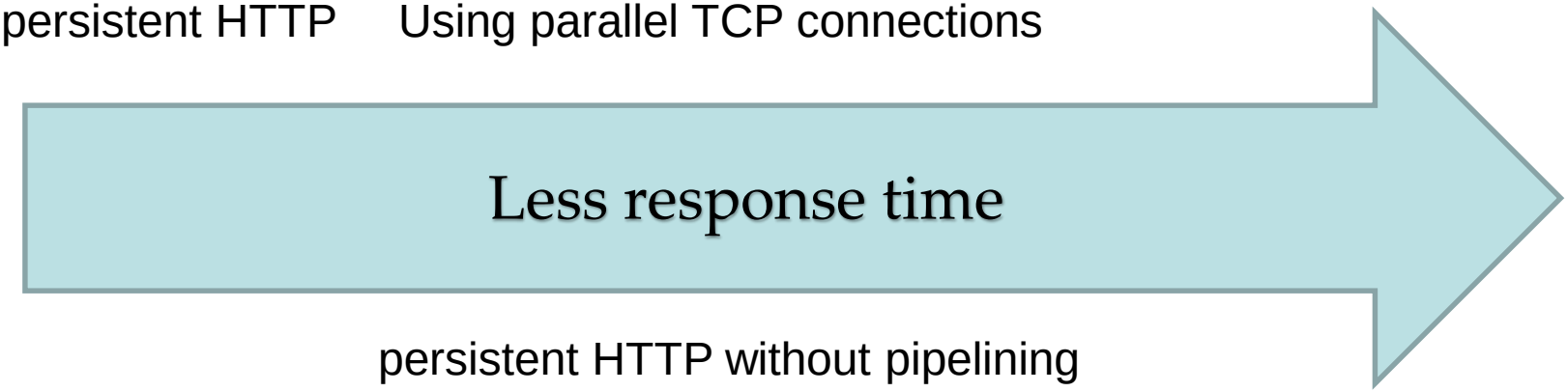
Persistent with pipelining:

- 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





Nonpersistent HTTP Using parallel TCP connections



persistent HTTP without pipelining

persistent HTTP with pipelining

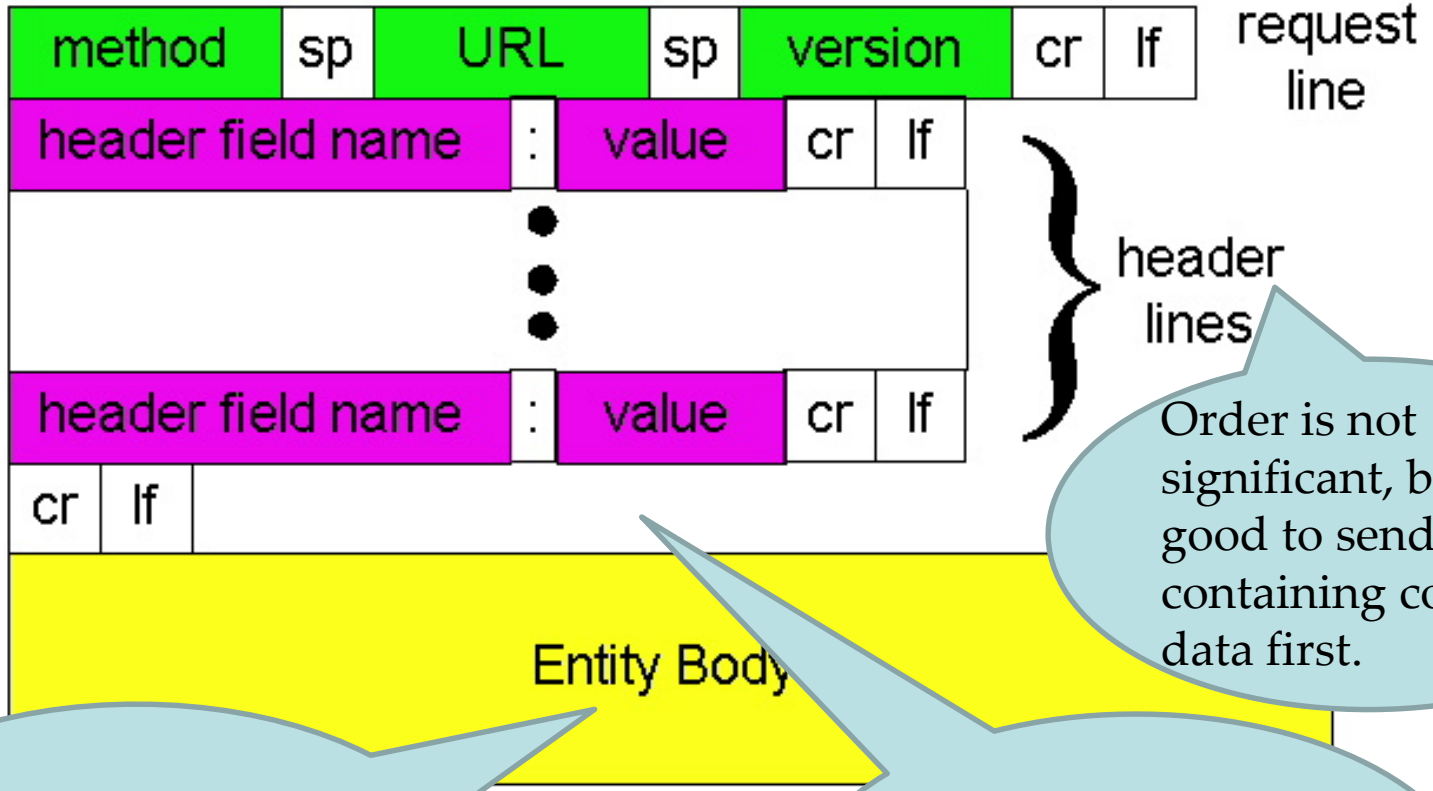
Outline

- HTTP
  - Non-persistent and persistent connections
  - Messages
  - Cookies
  - Web caching

HTTP Message Format

two types of HTTP messages: *request*, *response*

HTTP request message: general format



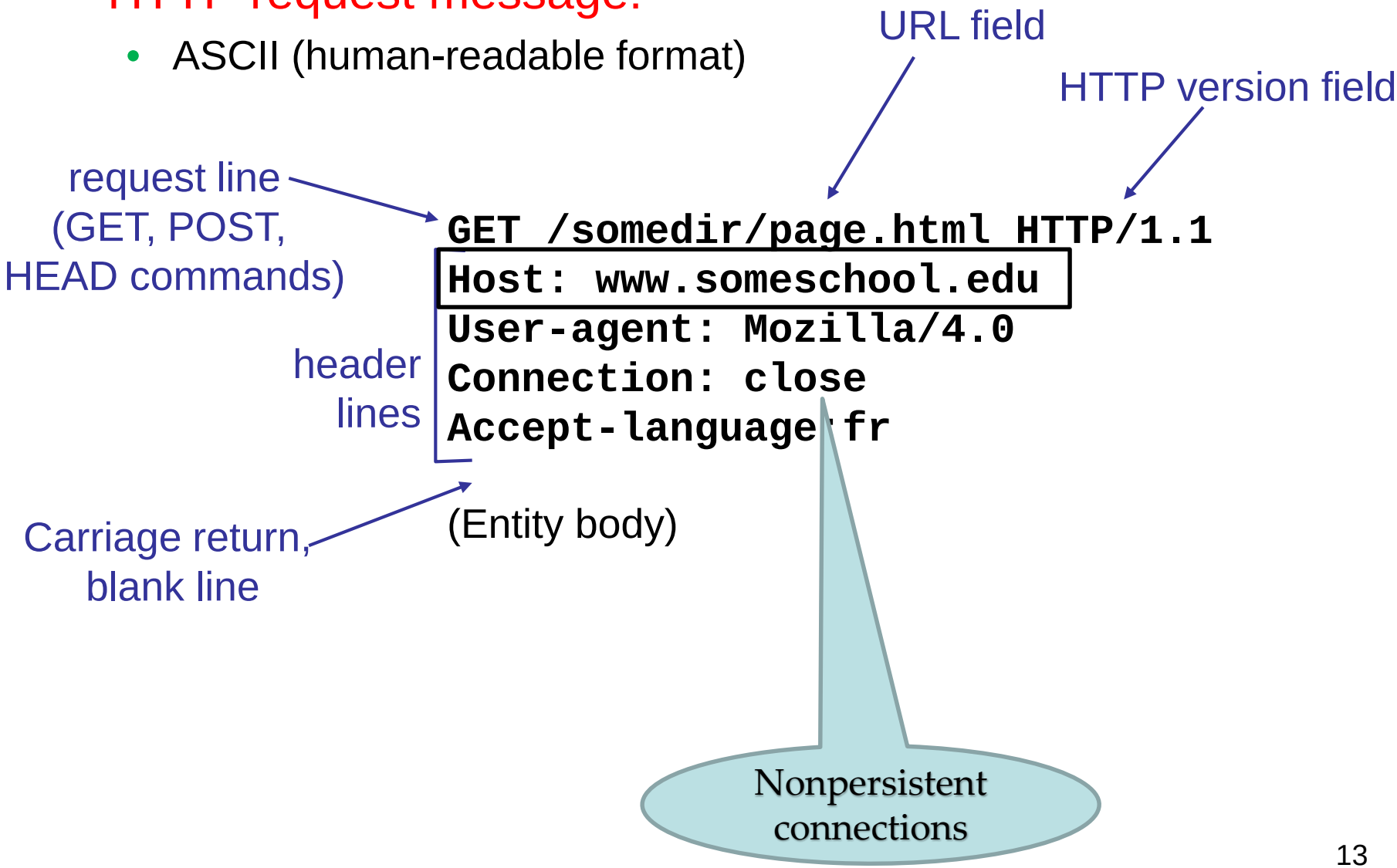
Order is not significant, but it is good to send fields containing control data first.

Empty for GET but it is used with POST (filling a form);

An empty line to indicate the end of the header section

HTTP request message

- HTTP request message:
  - ASCII (human-readable format)



Each language-range MAY be given an associated quality value which represents an estimate of the user's preference for the languages specified by that range. The quality value defaults to "q=1".

For example,
Accept-Language: da, en-gb;q=0.8, en;q=0.7

would mean: "*I prefer Danish, but will accept British English and other types of English*."

Method types

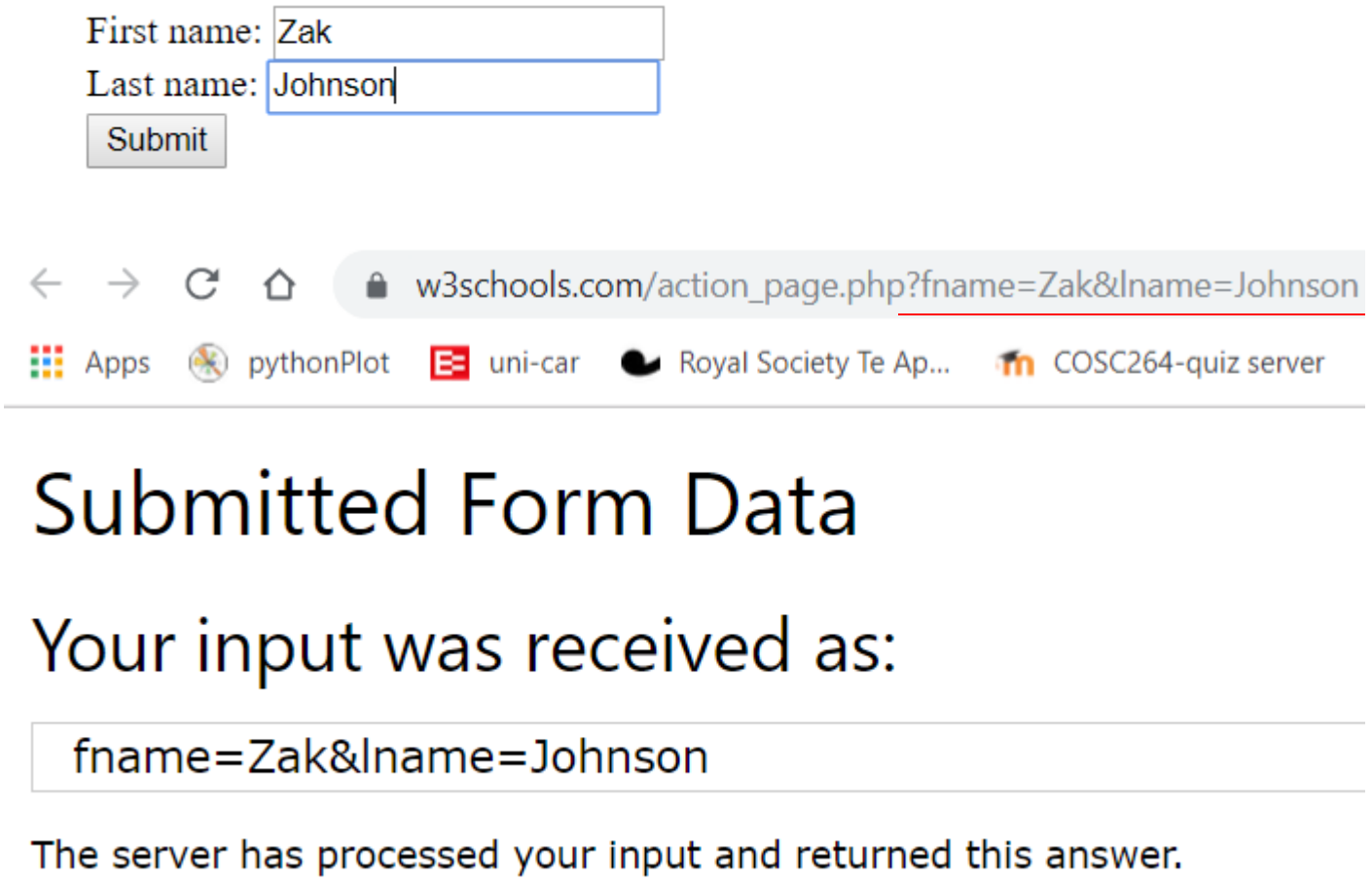
HTTP/1.0

- GET
- POST – filling a form
- HEAD
  - asks server to leave requested object out of response
  - Can be used for debugging

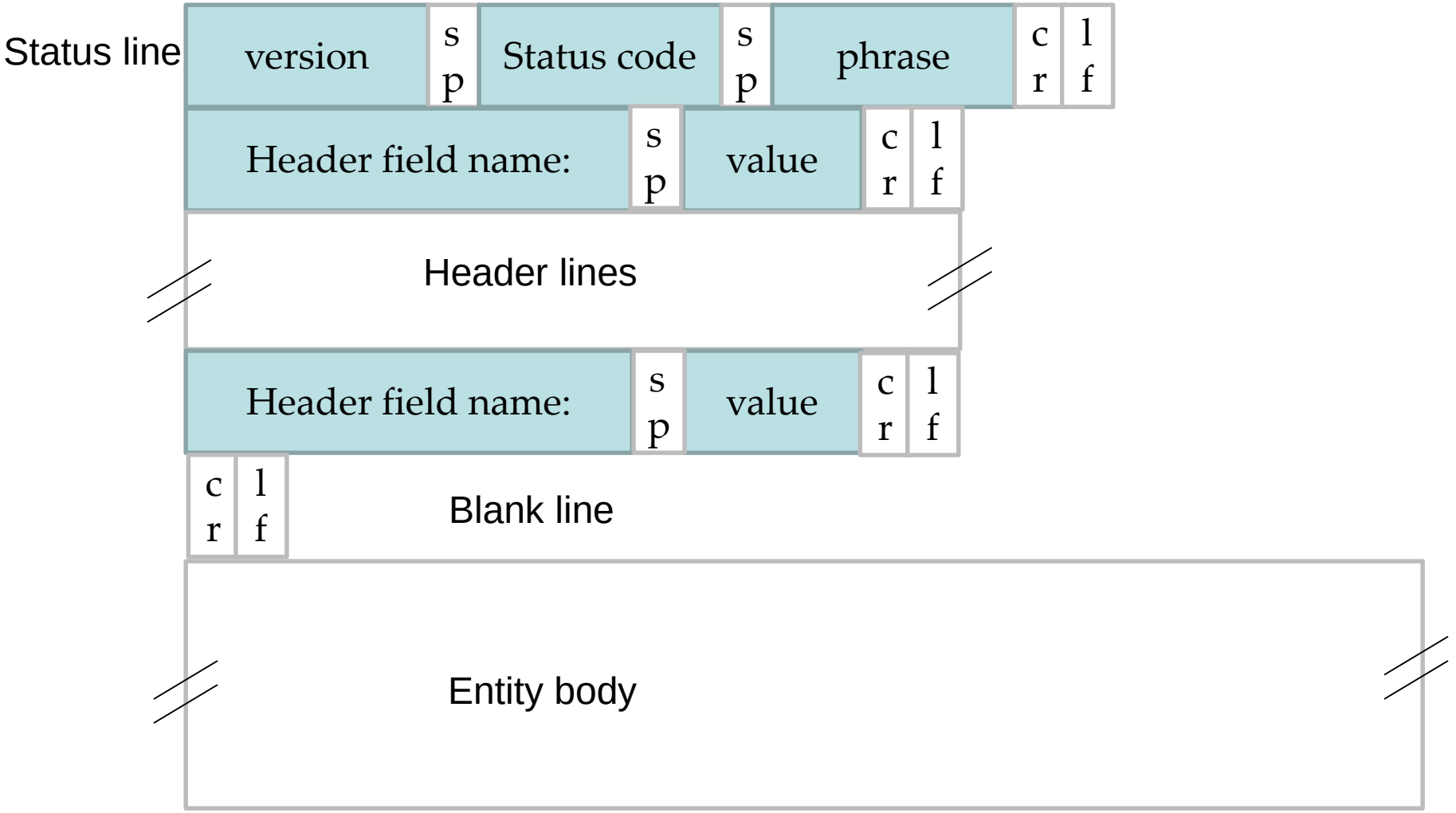
HTTP/1.1

- GET, POST, HEAD
- PUT
  - uploads file in entity body to path specified in URL field
- DELETE
  - deletes file specified in the URL field

Form submission can be done with GET as well;



General format of a response message



HTTP response status codes

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:)

400 Bad Request

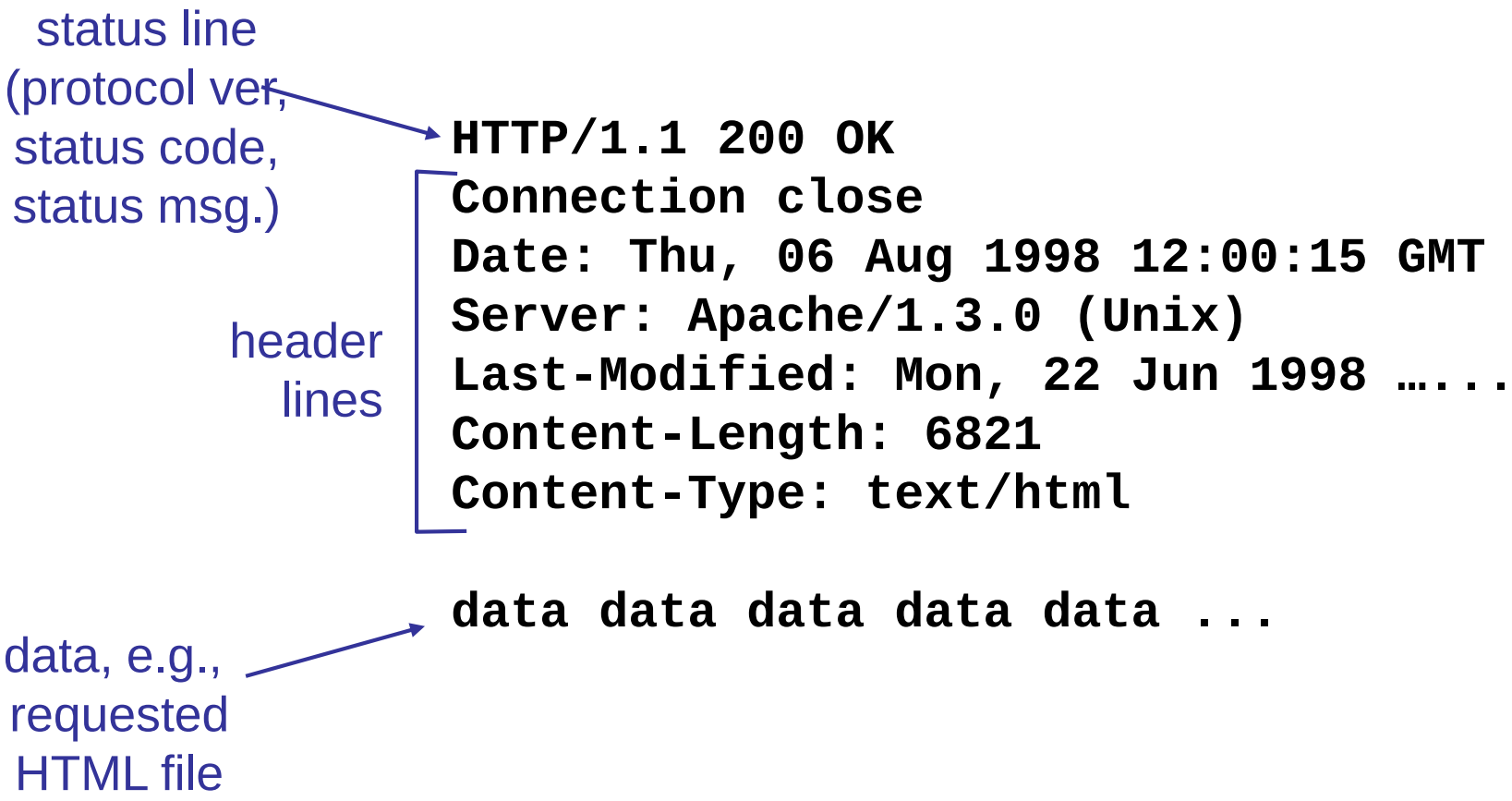
- request message not understood by server

404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported

HTTP response message



```
duser@192.168.88.155:~/libbgpdump-1.4.99.11$ telnet cis.poly.edu 80
Trying 128.238.26.21...
Connected to cis.poly.edu.
Escape character is '^]'.
GET /~ross/ HTTP/1.1
Host:cis.poly.edu

HTTP/1.1 200 OK
Date: Fri, 09 Aug 2019 02:46:08 GMT
Server: Apache/2.4.6
Last-Modified: Mon, 12 Nov 2018 16:25:17 GMT
ETag: "cf-57a7a257df256"
Accept-Ranges: bytes
Content-Length: 207
Content-Type: text/html; charset=UTF-8

<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta http-equiv="refresh"content="0;url=http://nyu.edu/projects/keithwross/">
<title> Automatic Forwarding </title>
</head>
Connection closed by foreign host.
```

An HTTP request message

An HTTP response message, sent by the server.

ross.html

```
1  <head>
2  <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
3  <meta http-equiv="refresh"content="0;url=http://nyu.edu/projects/keithwross/">
4  <title> Automatic Forwarding </title>
5  </head>
6
```

Not secure | nyu.edu/projects/keithwross/

Apps pythonPlot uni-car Royal Society Te Ap... COSC264-quiz server

Keith W. Ross

Dean of Engineering and Computer Science, NYU Shanghai
& Leonard J. Shustek Professor of Computer Science, CSE Dept,

Keith Ross is the Dean of Engineering and Computer Science at NYU Shar
Shustek Chair Professor of Computer Science at NYU. He also holds an aff
the Department of Computer Science at the Courant Institute of Mathema
Center for Data Science at NYU.

Previously he was a professor at University of Pennsylvania (13 years) and
Institute (5 years). He was the Department Head of the CSE Department a
and he joined NYU Shanghai in 2013. He received a Ph.D. in Computer an
from The University of Michigan.

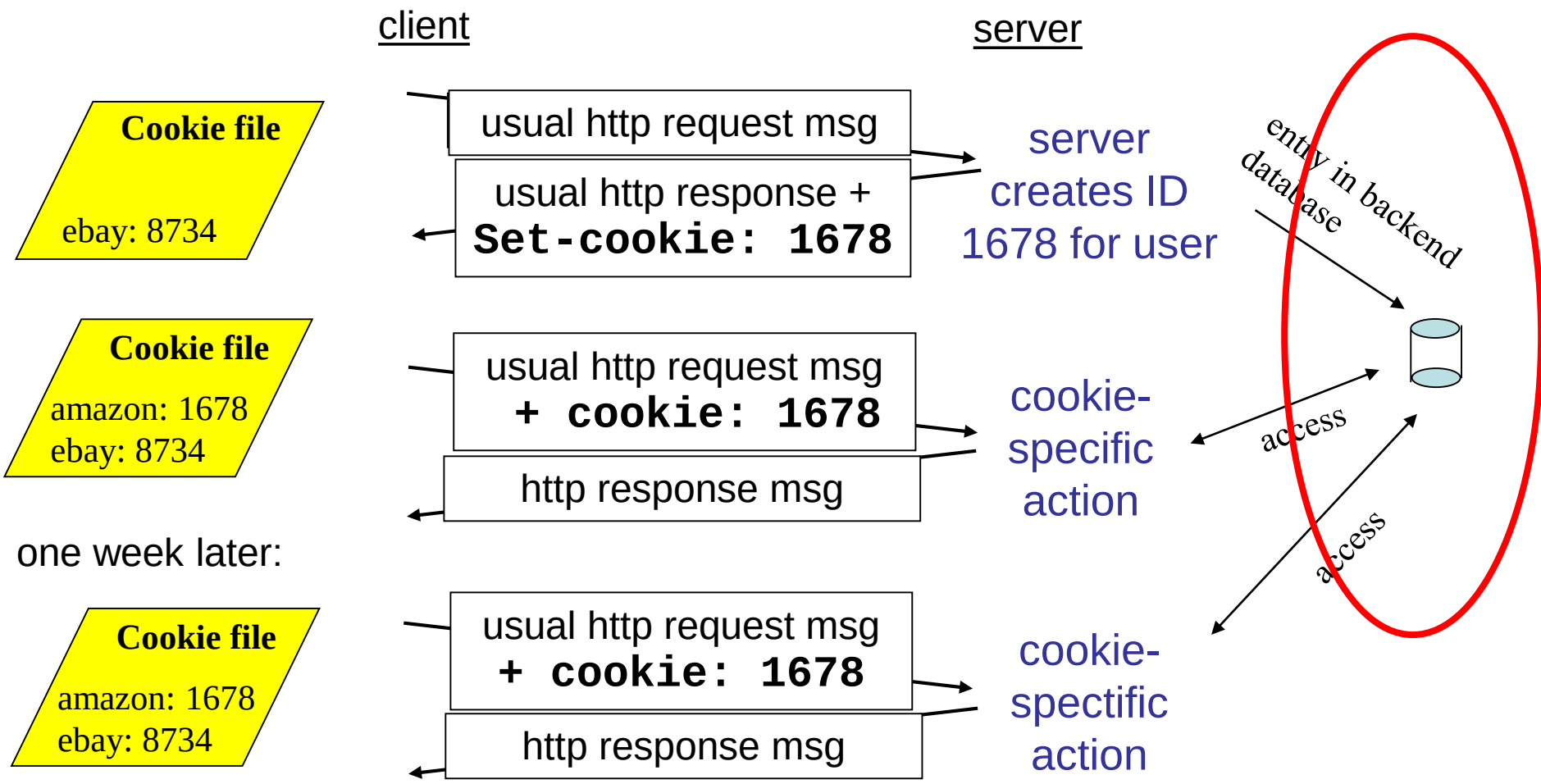
User-server state: cookies

Many major Web sites use cookies to *identify users*.

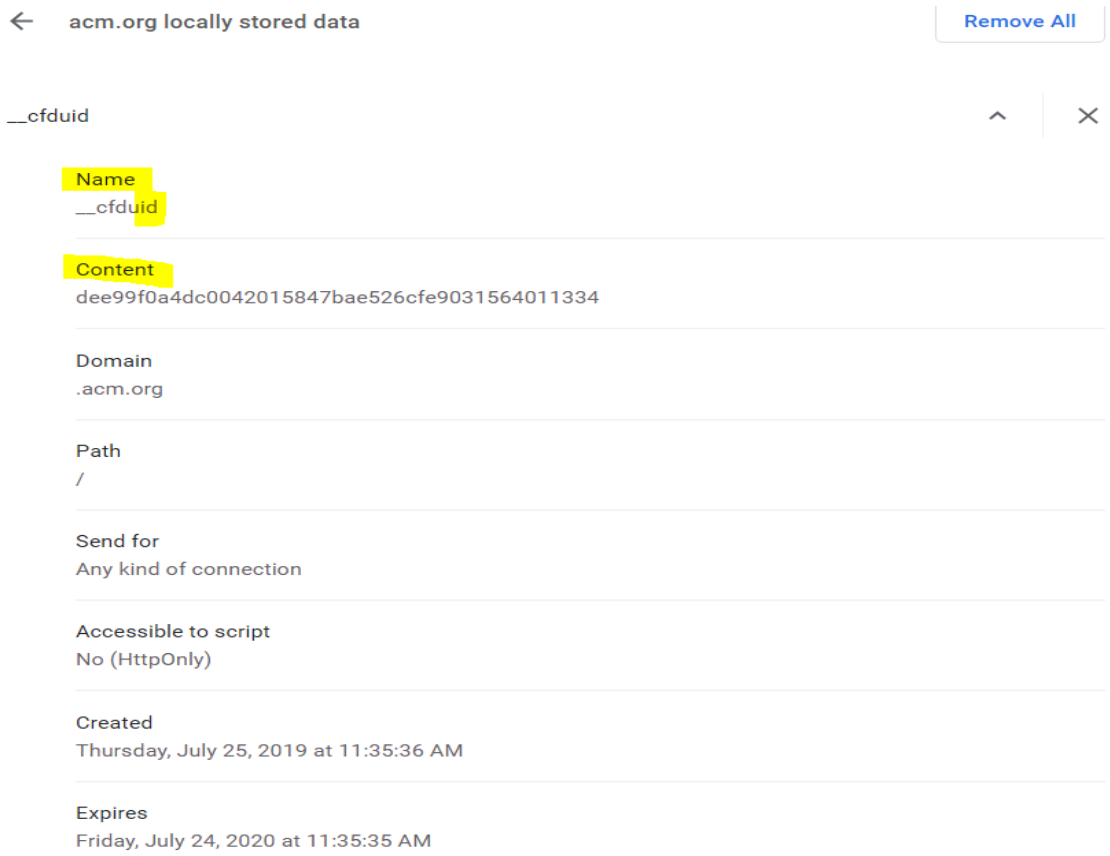
Four components:

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

Cookies: keeping “state” (cont.)



A cookie sample



Cookies (continued)

What cookies can bring:

- shopping carts
- recommendations
- user session state

aside

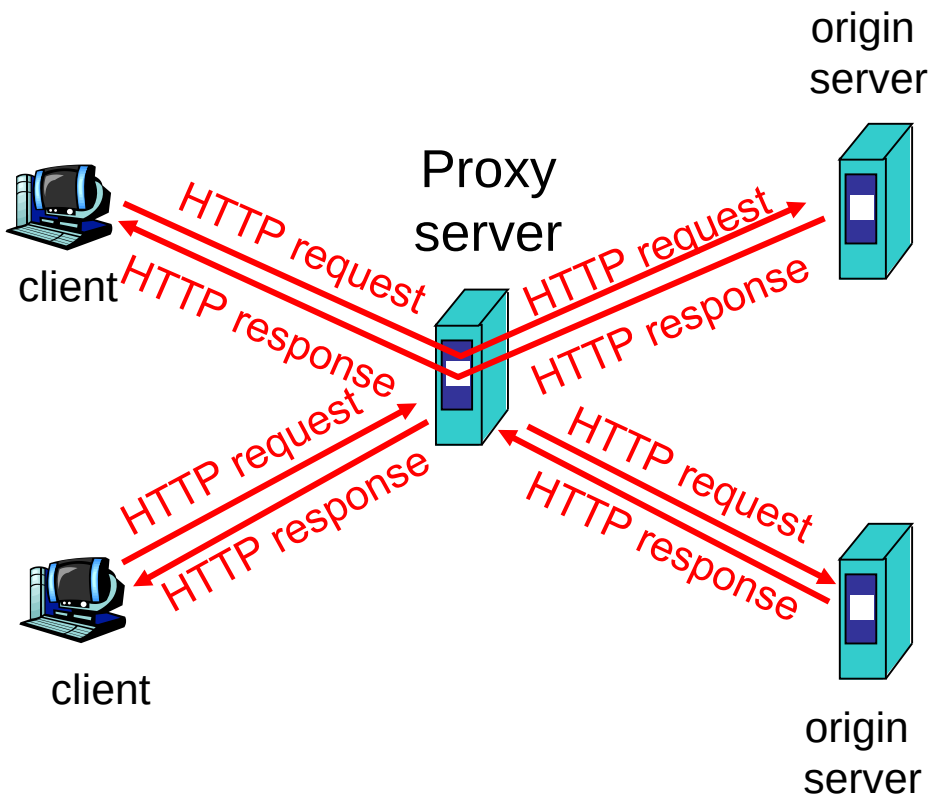
Cookies and privacy:

1. cookies permit sites to learn a lot about you
2. you may supply name and e-mail to sites
3. search engines use redirection & cookies to learn yet more
4. advertising companies obtain info across sites

Web caches (proxy server)

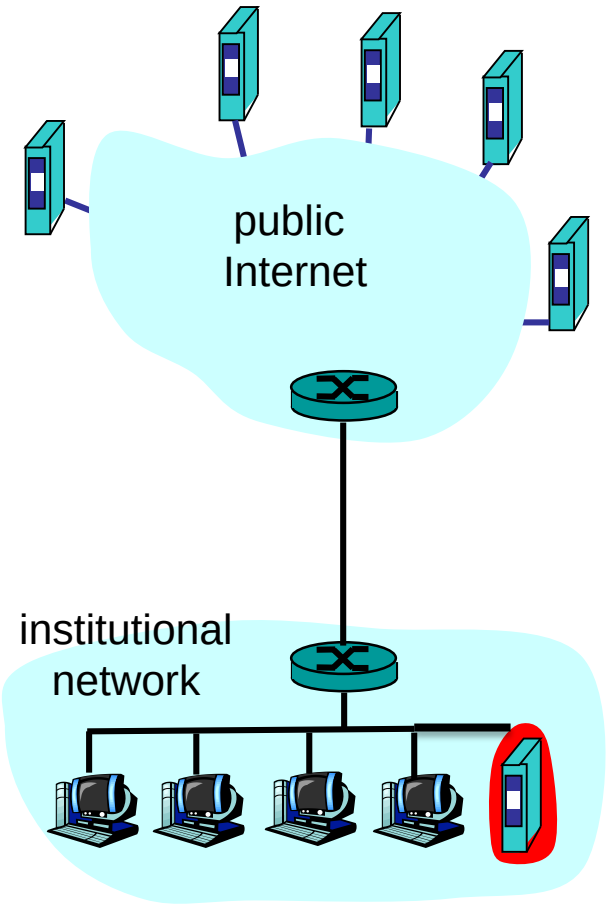
Goal: satisfy client request without involving origin server

- user sets browser: Web accesses via cache
- browser sends all HTTP requests to cache
  - 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)



Why Web caching?

- Reduce response time for client request.
- Reduce traffic on an institution's access link.
- Lowers bandwidth costs.

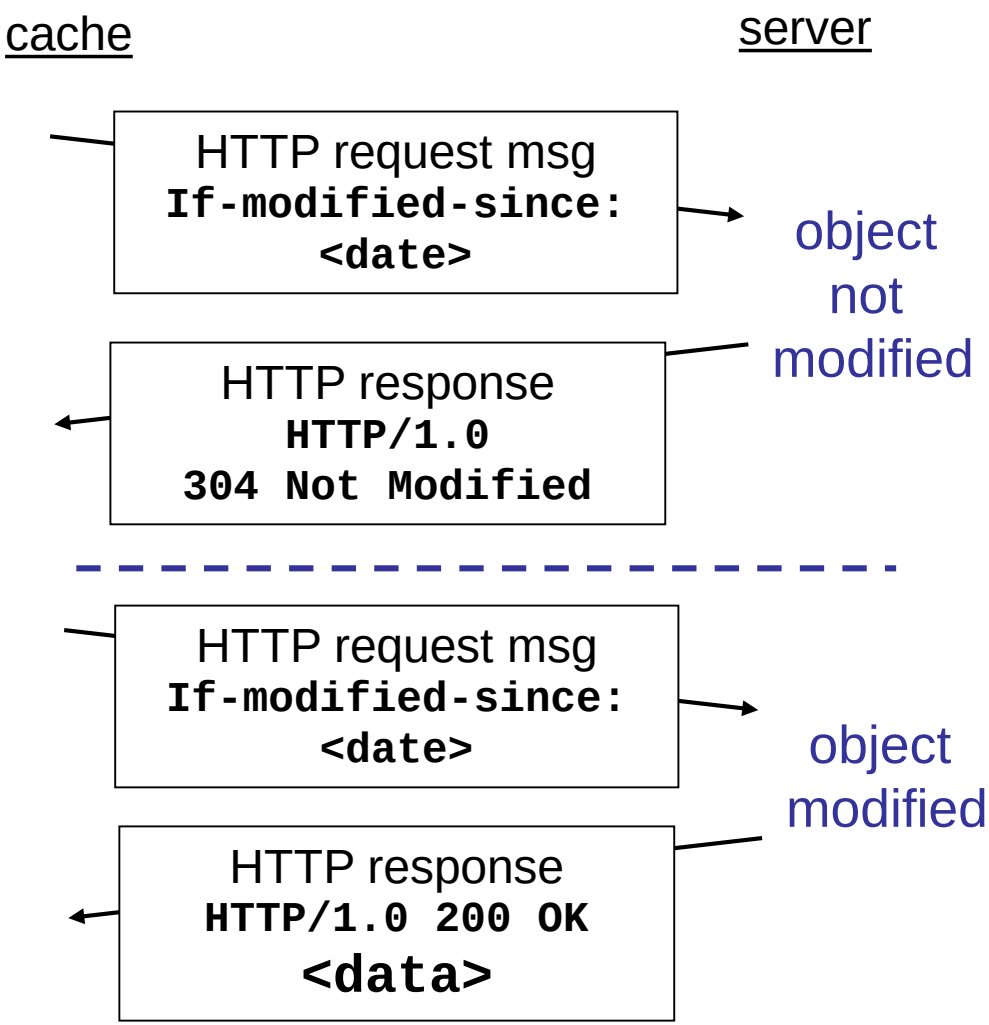
The copy of an object in the cache may be outdated.

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/1.0 304 Not Modified`



Outline

- HTTP
  - Non-persistent and persistent connections
  - Messages
  - Cookies
  - Web caching

References

- [KR3] James F. Kurose, Keith W. Ross, *Computer networking: a top-down approach featuring the Internet*, 3rd edition.
- [PD5] Larry L. Peterson, Bruce S. Davie, *Computer networks: a systems approach*, 5th edition
- [TW5] Andrew S. Tanenbaum, David J. Wetherall, *Computer network*, 5th edition
- [LHBi]Y-D. Lin, R-H. Hwang, F. Baker, *Computer network: an open source approach*, International edition

Acknowledgements

- All slides are developed based on slides from the following two sources:
  - Dr DongSeong Kim's slides for COSC264, University of Canterbury;
  - Prof Aleksandar Kuzmanovic's lecture notes for CS340,Northwestern University, https://users.cs.northwestern.edu/~akuzma/classes/CS340-w05/lecture_notes.htm