Welcome to

CS 3516: Computer Networks

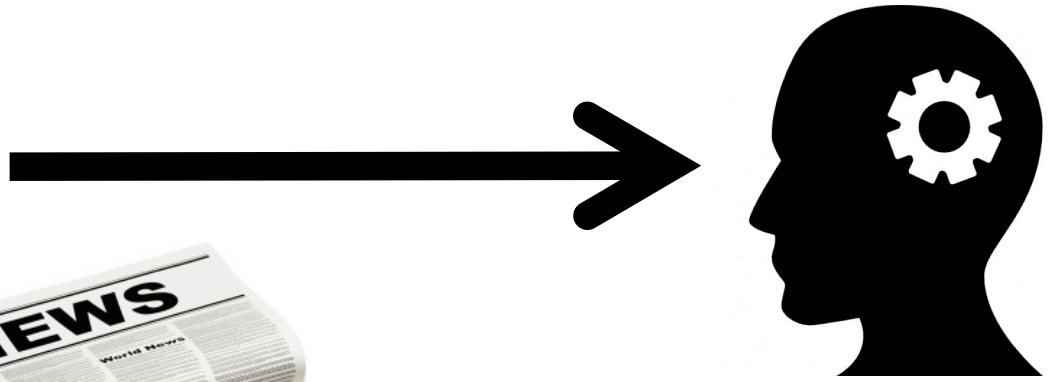
Prof. Yanhua Li

Time: 9:00am –9:50am M, T, R, and F

Zoom Lecture

Fall 2020 A-term

Web changes how people receive/publish information



One Way

On demand

Chapter 2: outline

2.1 principles of network applications

- app architectures
- app requirements

2.2 Web and HTTP

- Overview
- Persistent vs non-persistent
- HTTP message formats
- Project 1 demo
- Web cookies
- Web Proxy

Web and HTTP


orcester Polytechnic Institute [US]
https://www.wpi.edu
Industria...
News
Web
MENU
Worcester Polytechnic Institute
SEARCH
More Than "Wicked Smaht"
At WPI, we are scientists, engineers, and future business leaders. We are also athletes, artists, thespians, entrepreneurs, musicians, and spirited fans. (And you'll hear more than just a Boston accent.) Welcome home.

Web and HTTP

First, a review…

- *web page* consists of *objects*
- object can be HTML file, JPEG image, Java applet, audio file,…
- web page consists of *base HTML-file* which includes *several referenced objects*
- each object is addressable by a *URL,* e.g.,

`www.someschool.edu/someDept/pic.gif`

host name | path name

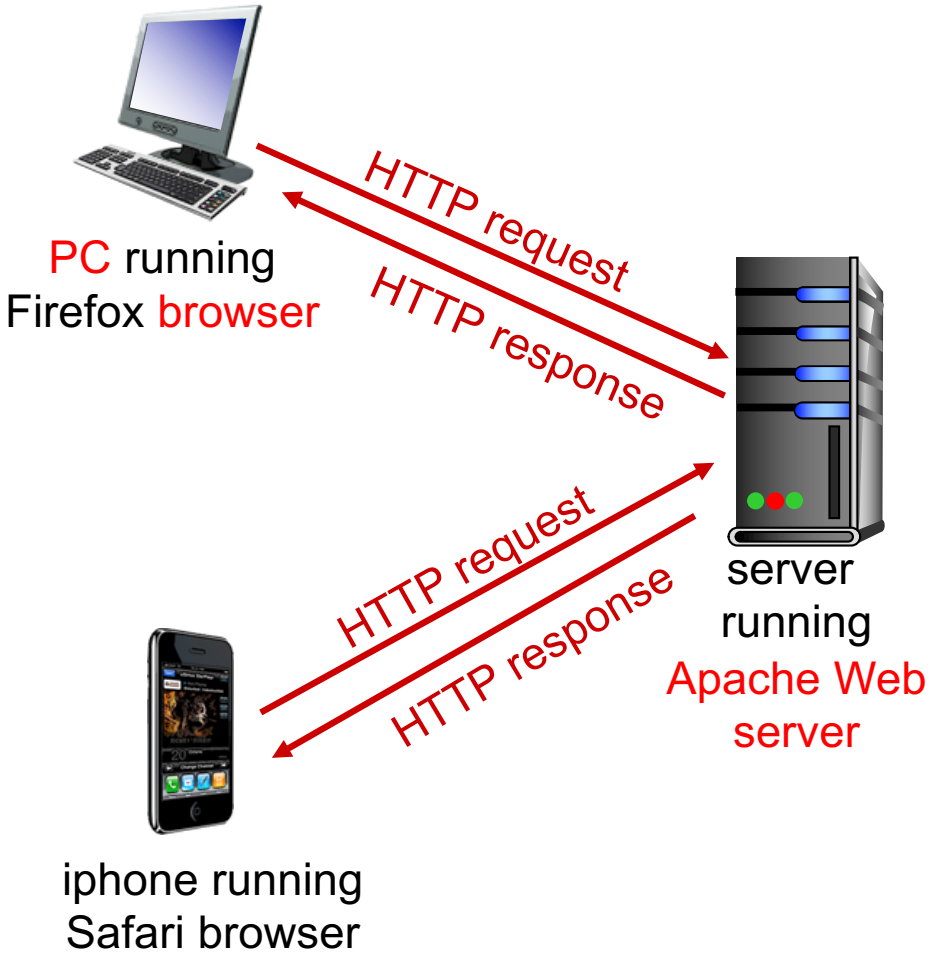
Web and HTTP


orcester Polytechnic Institute [US]
https://www.wpi.edu
Industria...
News
Web
MENU
Worcester Polytechnic Institute
SEARCH
More Than "Wicked Smaht"
At WPI, we are scientists, engineers, and future business leaders. We are also athletes, artists, thespians, entrepreneurs, musicians, and spirited fans. (And you'll hear more than just a Boston accent.) Welcome home.

HTTP overview

HTTP: hypertext transfer protocol

- 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 overview (continued)

uses TCP:

- 1. client initiates TCP connection (creates socket) to server, port 80
- 2. server accepts TCP connection from client
- 3. HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- 4. TCP connection closed

HTTP is "stateless"

- server maintains no information about past client requests

aside

protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, must be reconciled

Chapter 2: outline

2.1 principles of network applications

- app architectures
- app requirements

2.2 Web and HTTP

- Overview
- Persistent vs non-persistent
- HTTP message formats
- Project 1 demo
- Web cookies
- Web Proxy

HTTP connections

non-persistent HTTP

- at most one object sent over TCP connection
  - connection then closed
- downloading multiple objects required multiple connections

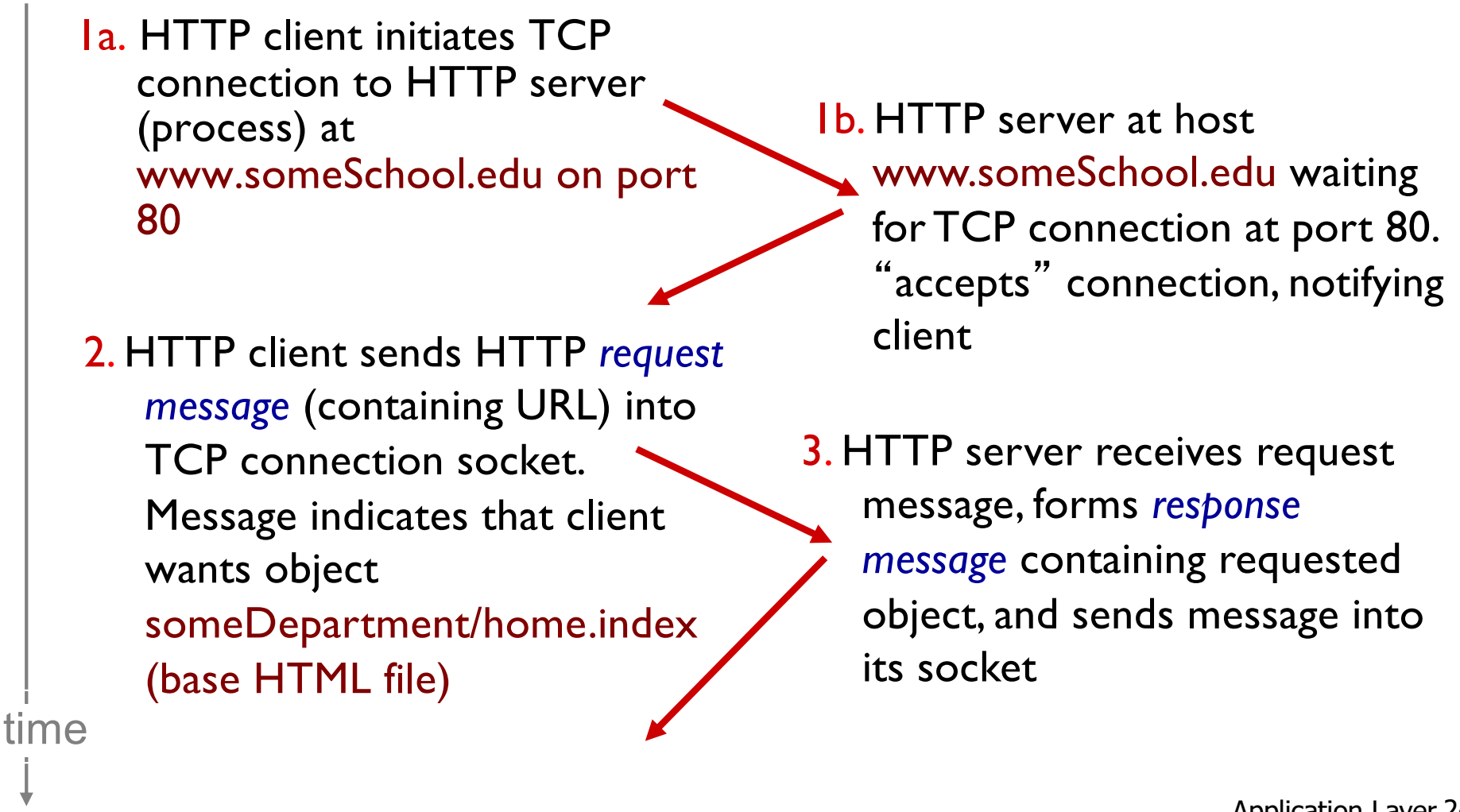
persistent HTTP

- multiple objects can be sent over single TCP connection between client, server

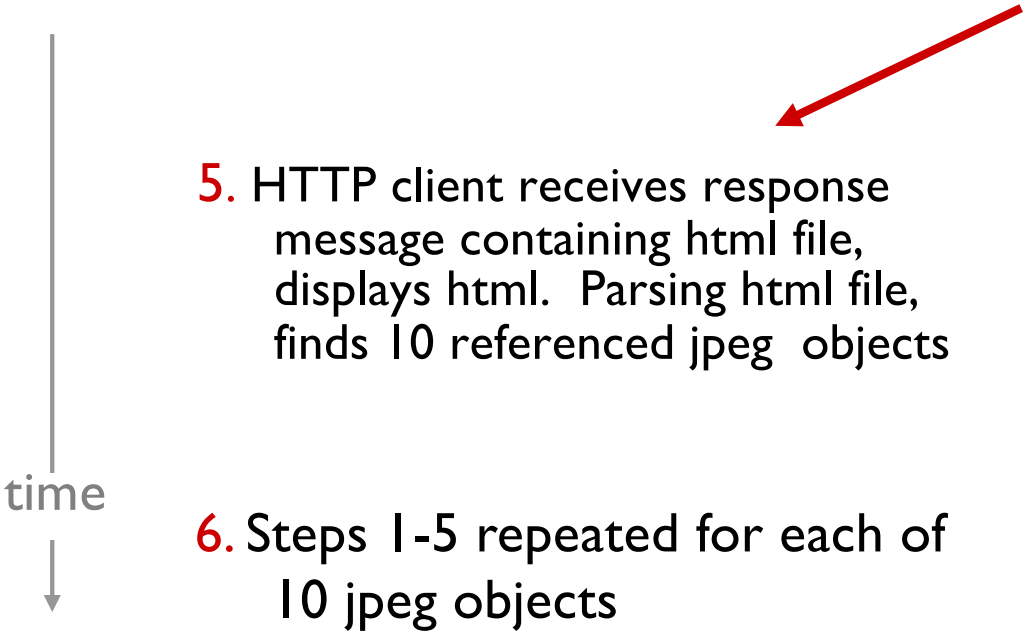
Non-persistent HTTP

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

(contains text, references to 10 jpeg images)



Non-persistent HTTP (cont.)



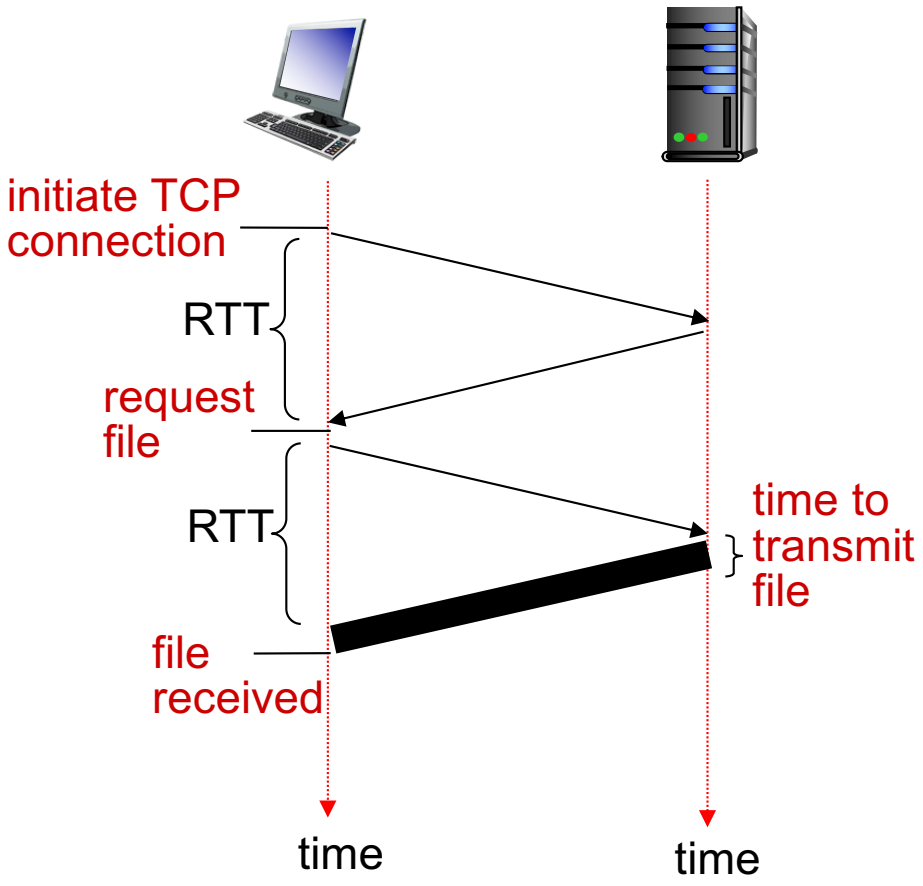
Non-persistent HTTP: response time

RTT (round-trip time): time for a small packet to travel from client to server and back

HTTP 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
- non-persistent HTTP response time =

  2RTT+ file transmission time



Persistent HTTP (Default)

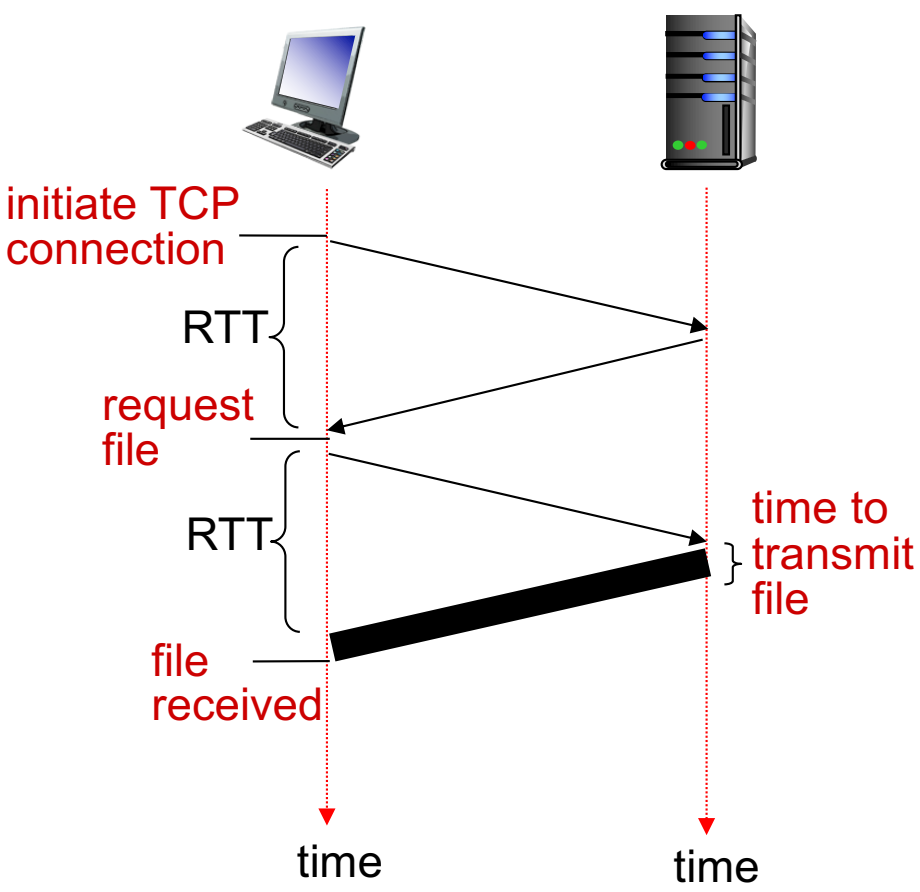
non-persistent HTTP issues:

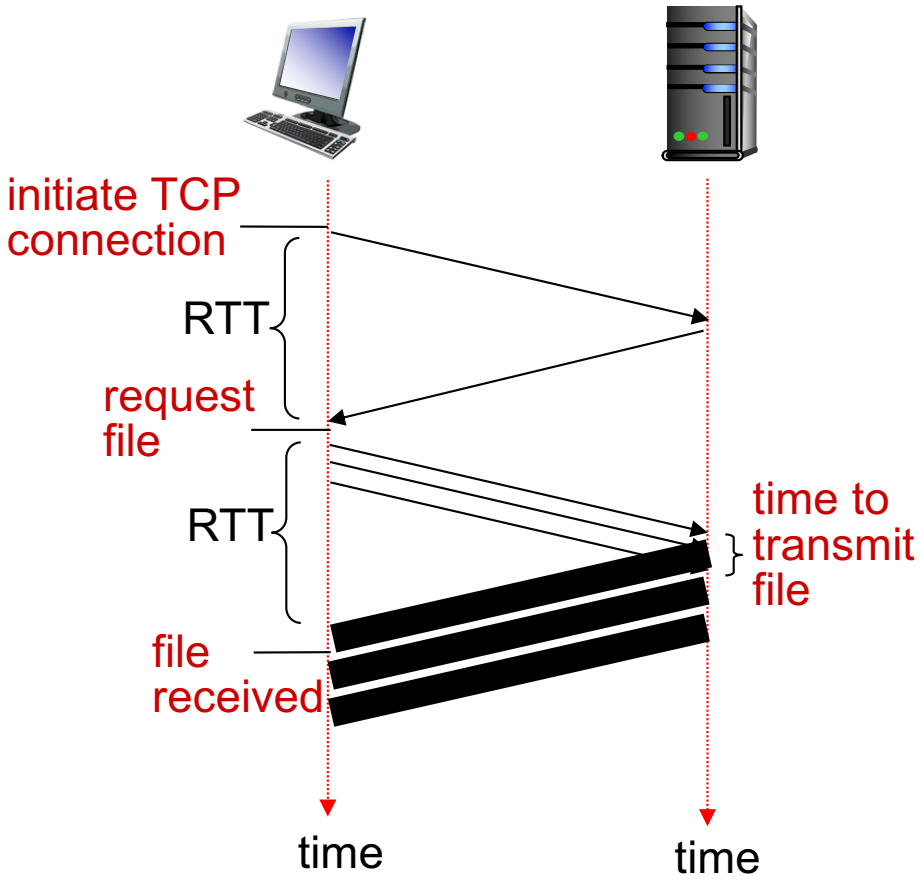
- requires 2 RTTs per object
- OS overhead for *each* TCP connection
- browsers often open parallel TCP connections to fetch referenced objects
  - 5-10
  - Adjustable

persistent HTTP:

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- Back-to-back requests/responses: client sends requests as soon as it encounters a referenced object
  - as little as one RTT for all the referenced objects

Non-persistent vs persistent





Chapter 2: outline

2.1 principles of network applications

- app architectures
- app requirements

2.2 Web and HTTP

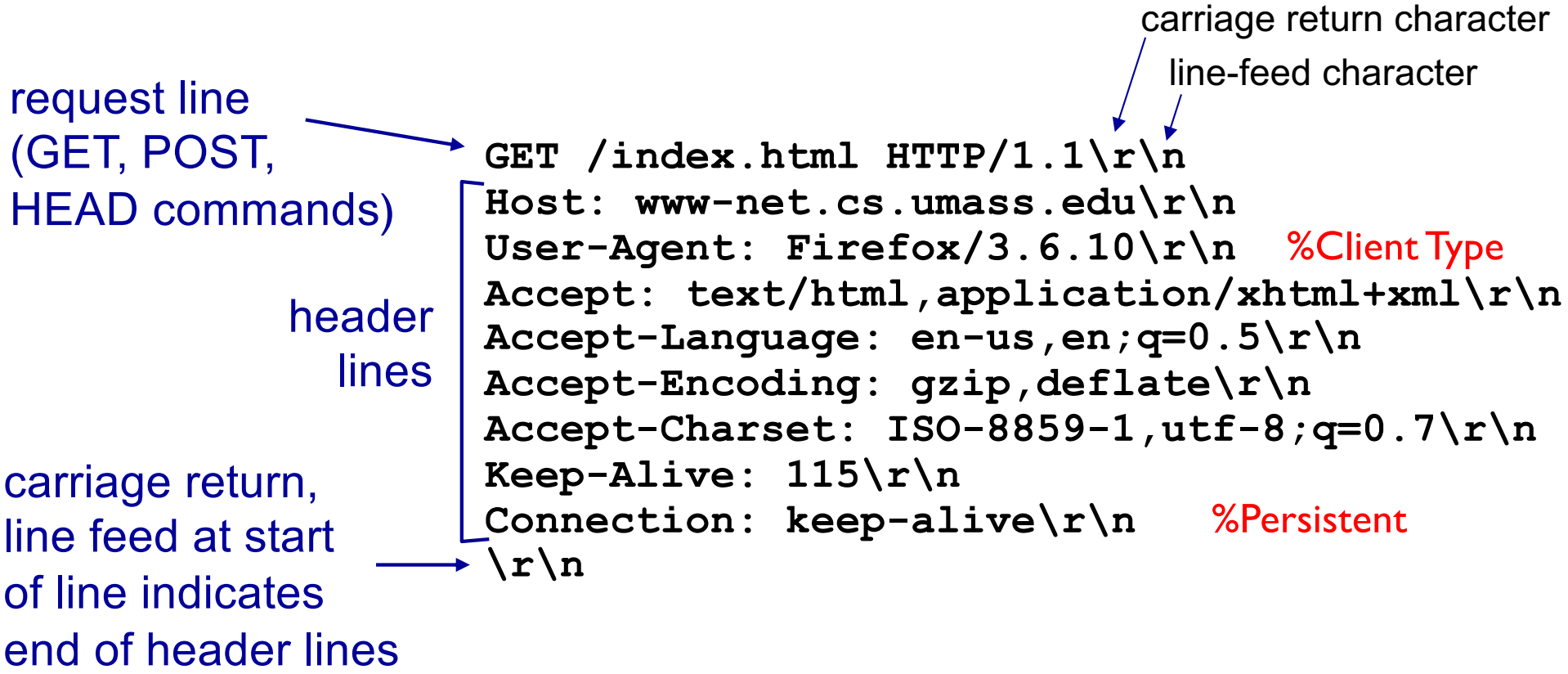
- Overview
- Persistent vs non-persistent
- HTTP message formats

- Demo of Project 1

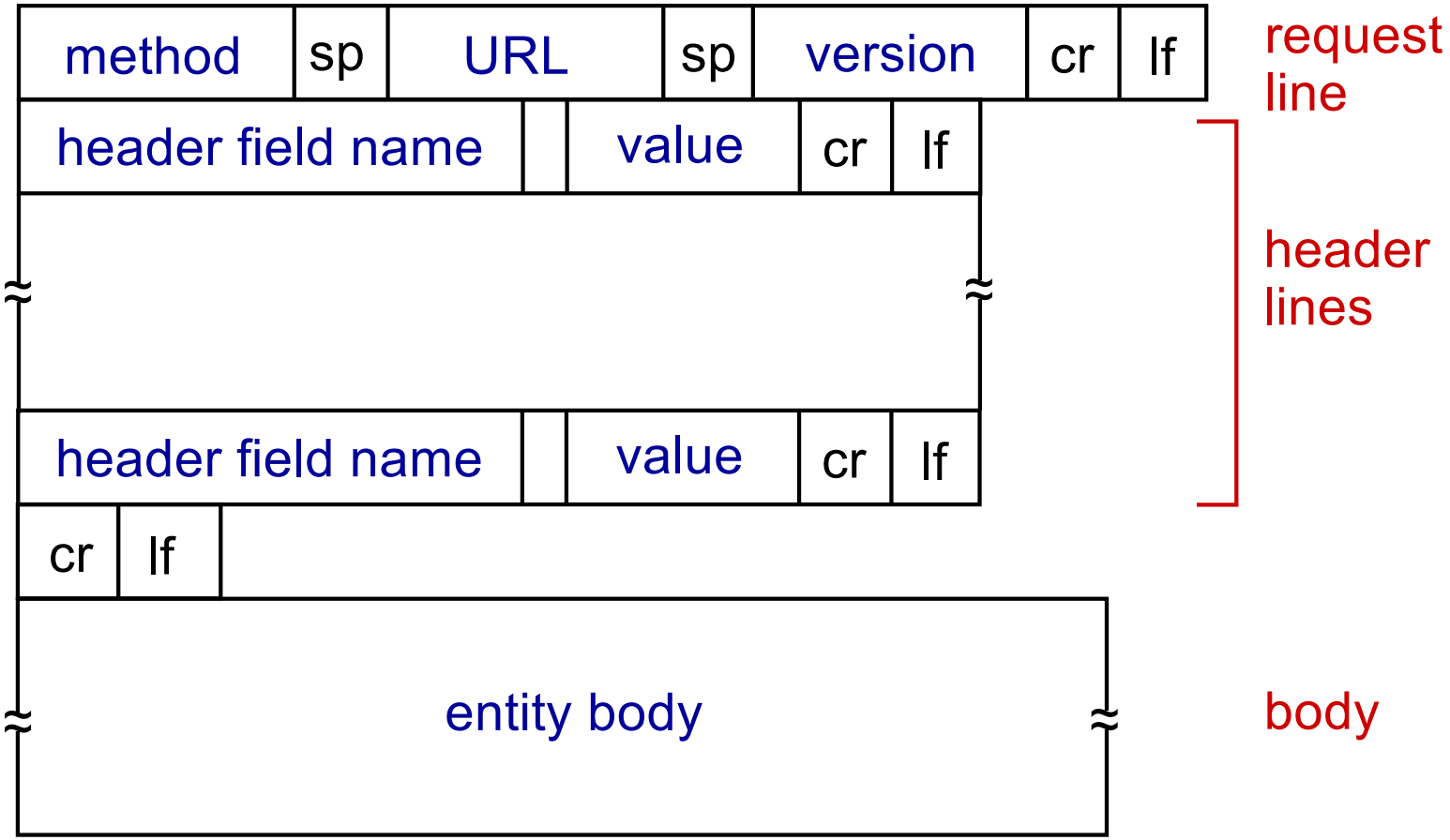
- Web cookies
- Web Proxy

HTTP request message

- two types of HTTP messages: *request, response*
- HTTP request message:
  - ASCII (human-readable format)



HTTP request message: general format



Request from input (POST & GET)

POST method:

- web page often includes input
- input is uploaded to server in entity body

GET+URL method:

- uses GET method
- input is uploaded in URL field of request line:

```
www.somesite.com/animalsearch?monkeys&banana
```

Method types

HTTP/1.0:

- GET
- POST
- HEAD
  - asks server to leave requested object out of response

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

HTTP response message

status line (protocol status code status phrase)

header lines

data, e.g., requested HTML file

```
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
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\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 ...
```

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 msg

301 Moved Permanently

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

400 Bad Request

- request msg not understood by server

404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported

Chapter 2: outline

2.1 principles of network applications

- app architectures
- app requirements

2.2 Web and HTTP

- Overview
- Persistent vs non-persistent
- HTTP message formats
- Web cookies
- Web Proxy

User-server state: cookies

many Web sites use cookies

four components:

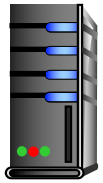
1) cookie header line of HTTP *response* message
2) cookie header line in next HTTP *request* message
3) cookie file kept on user's host, managed by user's browser
4) back-end database at Web site

example:

- Susan always access Internet from PC
- 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" (cont.)

client



server

ebay 8734

cookie file

usual http request msg

Amazon server creates ID 1678 for user

create entry

backend database

usual http response
set-cookie: 1678

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

cookie-specific action

access

usual http response msg

time

one week later:

access

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

cookie-specific action

usual http response msg

Cookies (continued)

what cookies can be used for:

- 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 supply name and e-mail to sites

Chapter 2: outline

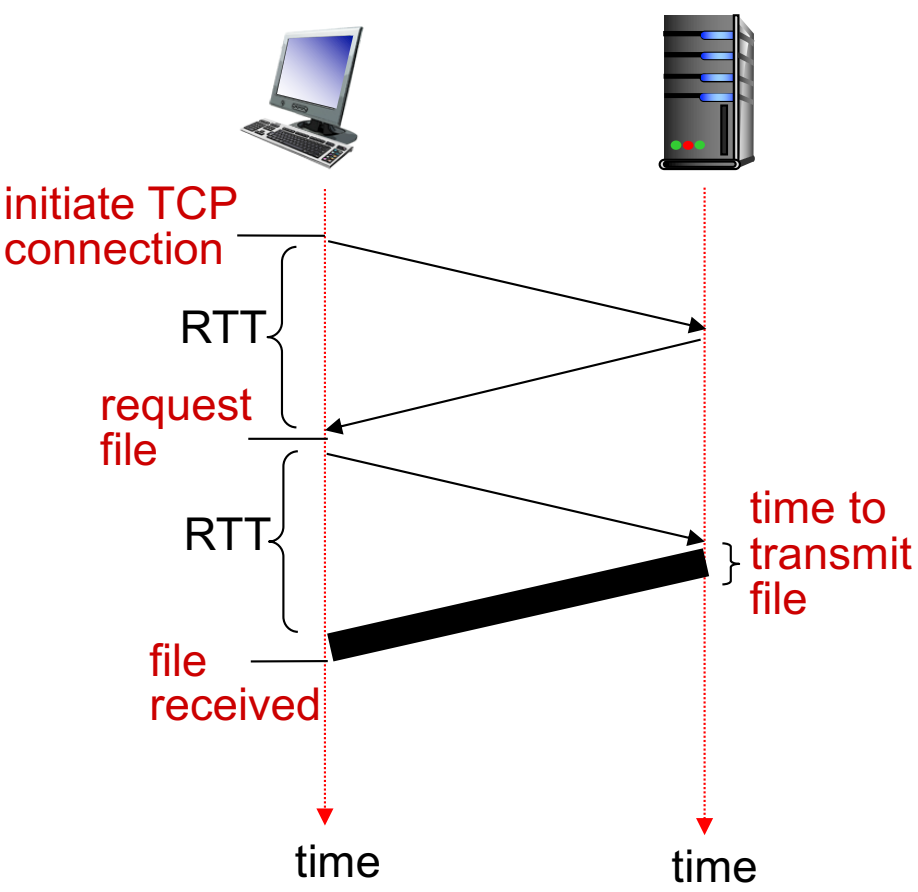
2.1 principles of network applications

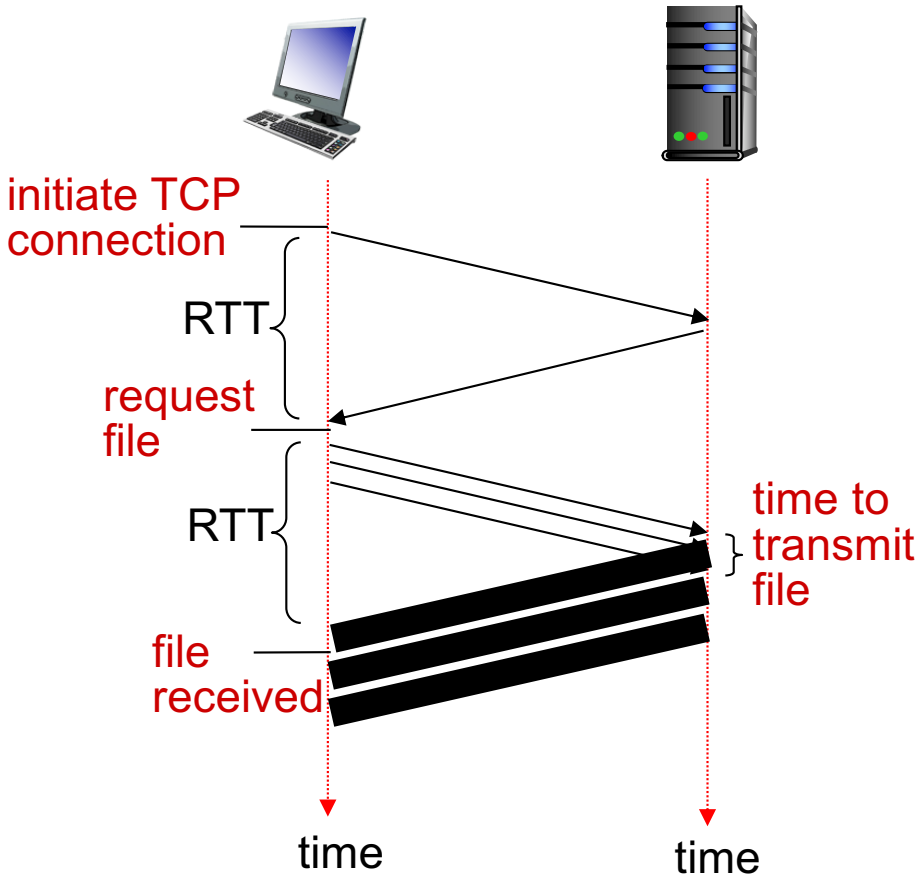
- app architectures
- app requirements

2.2 Web and HTTP

- Overview
- Persistent vs non-persistent
- HTTP message formats
- Web cookies
- Web Proxy

Non-persistent vs persistent





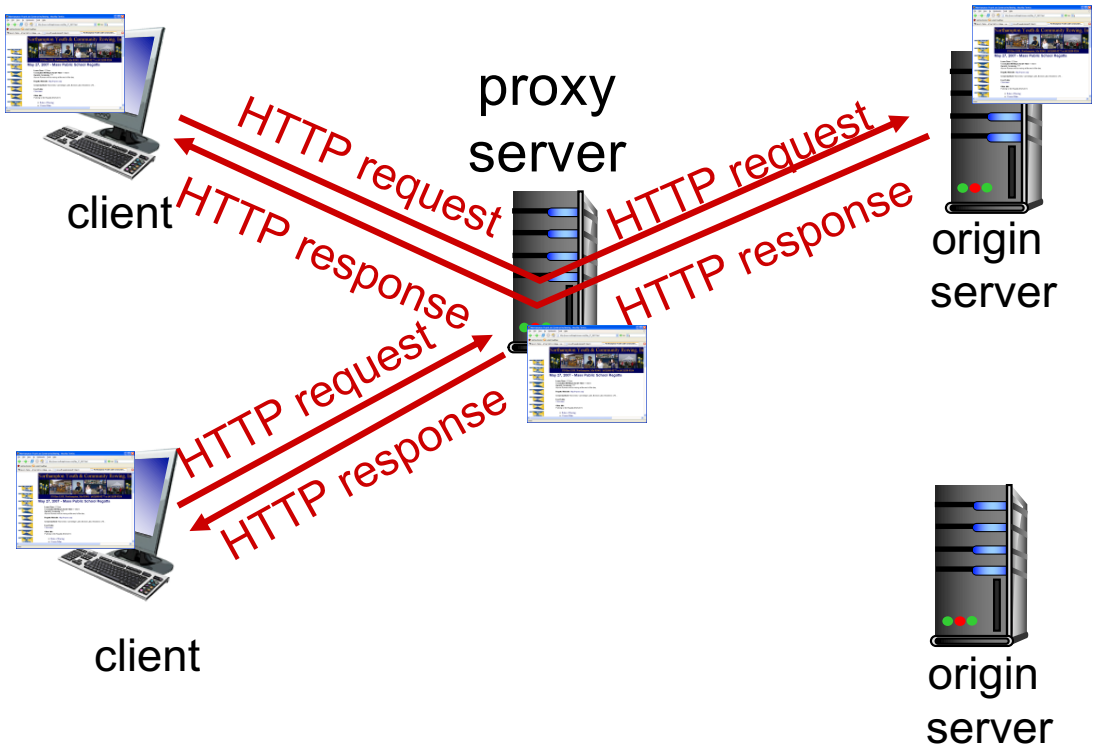
$T=N * (2 * RTT + T_{obj})$

$T=2 * RTT + N * T_{obj}$

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

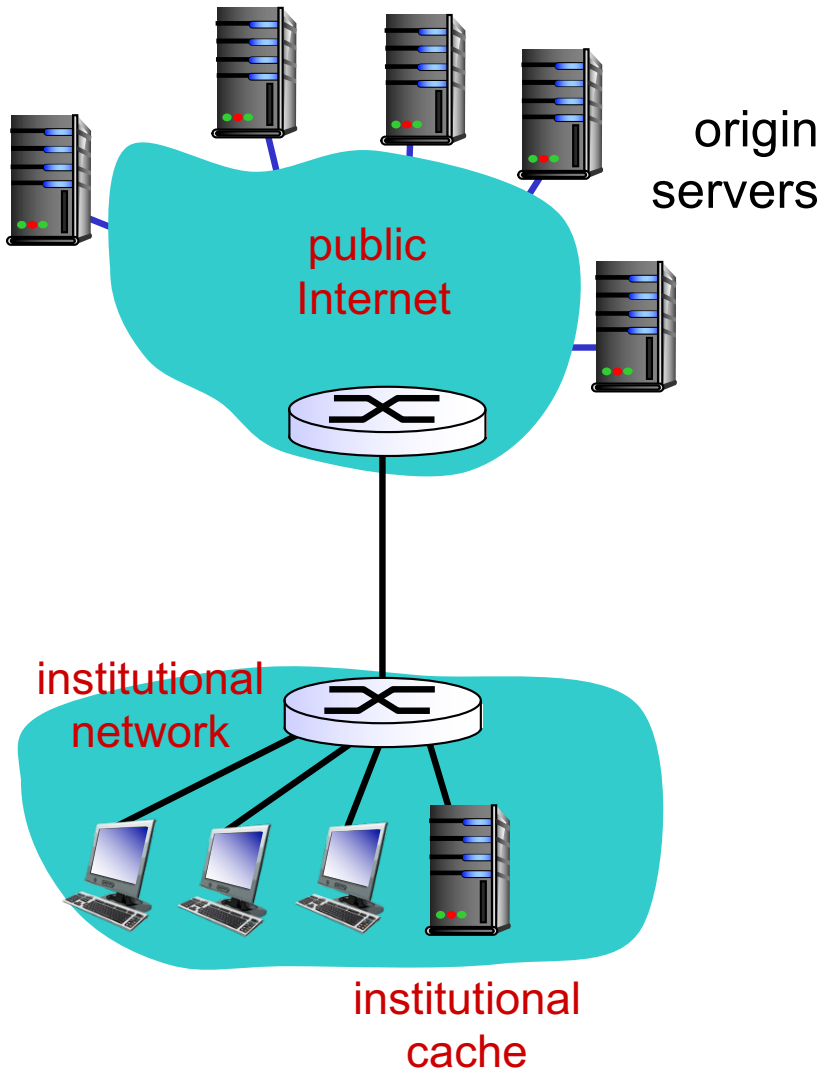


Caching example:

- total delay = Internet delay + access delay + Local area network (LAN) delay

- cache acts as both client and server
  - server for original requesting client
  - client to origin server

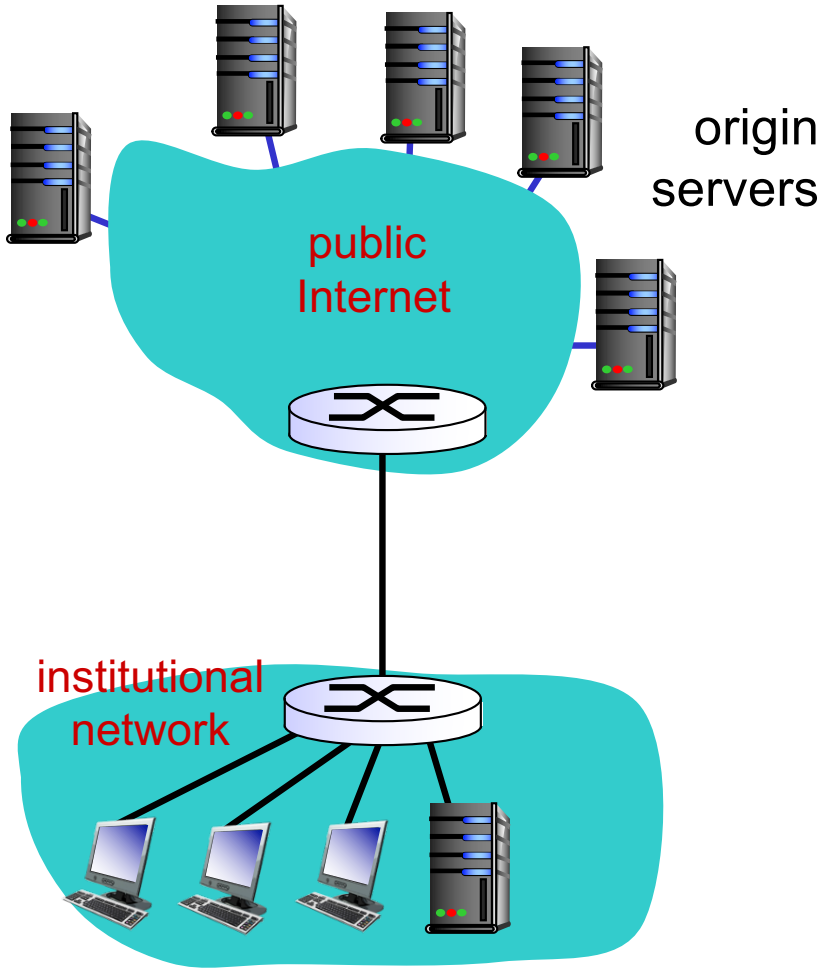
- typically cache is installed by access ISP (university, company, residential ISP)



More about Web caching

why Web caching?

- **Client:** reduce response time for client request
- **Access:** reduce traffic on an institution's access link
- **Internet:** Internet dense with caches: enables "poor" content providers to effectively deliver content (so too does P2P file sharing)



Concepts

- HTTP: HyperText Transfer Protocol
- HTML: HyperText Makeup Language
- URL: Uniform Resource Locator
- Web browser = Web Client
- RTT: Round-Trip Time
- TCP: Transmission control protocol

Questions?