BITS Pilani
Pilani Campus

Computer Networks (CS F303)

Virendra Singh Shekhawat
Department of Computer Science and Information Systems

innovate achieve lead

BITS Pilani
Pilani Campus

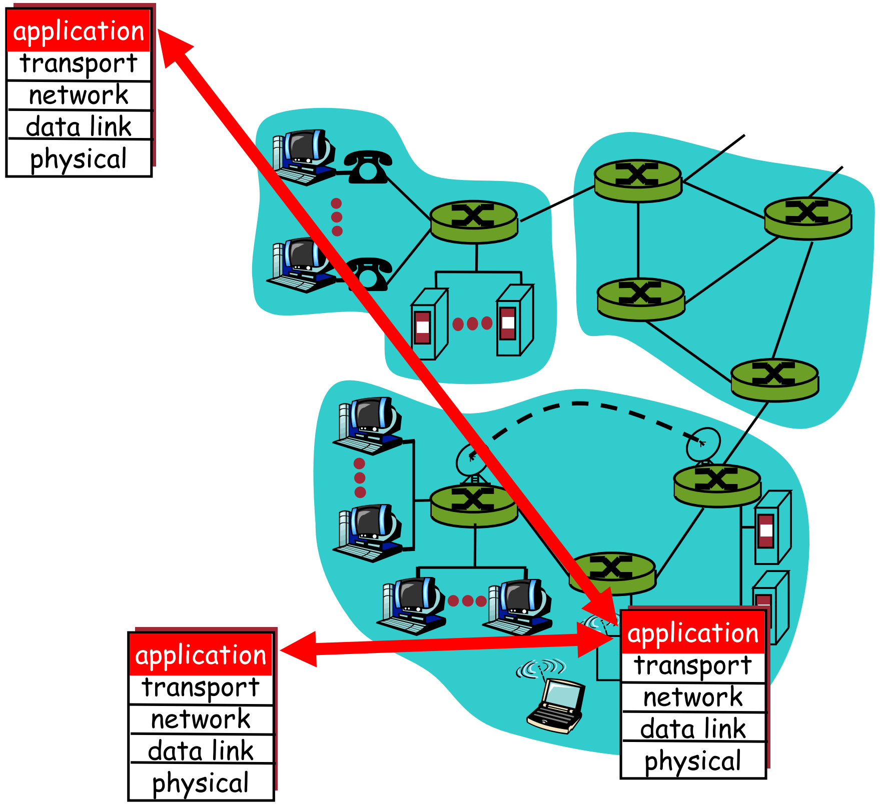
Second Semester 2020-2021
Module-2 Application Layer

What is a Network Application?

- Programs that run on different end systems and communicate over a network
  - e.g., Web: Web server software communicates with browser software

- Network core devices do not run user application code

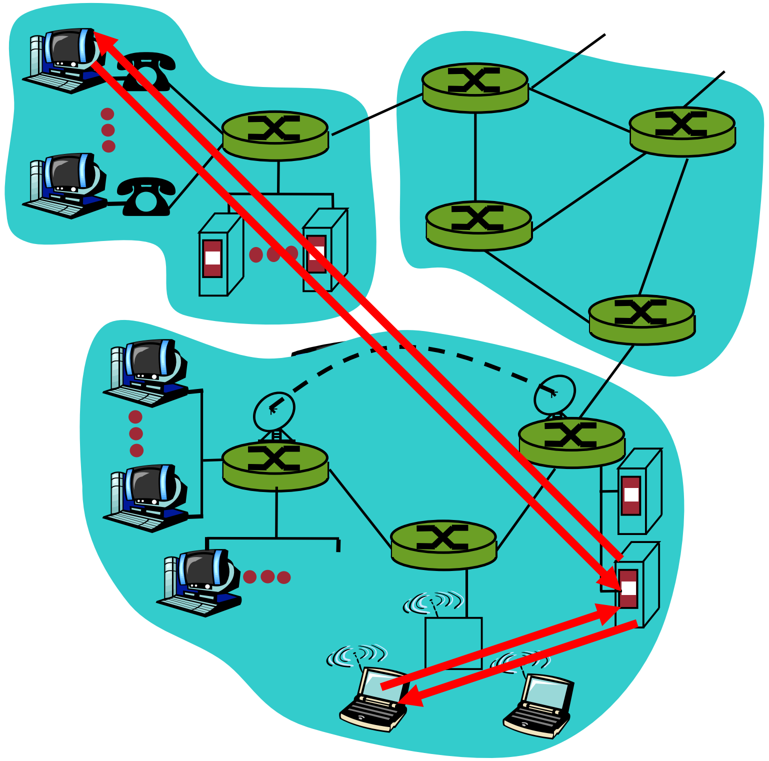
- Application on end systems allows for rapid application development



Application architectures

- Client-server
- Peer-to-Peer (P2P)
- Hybrid of client-server and P2P

Client-Server Architecture



Server:

- "always-on" host
- Permanent IP address
- For scaling, data center is used to create large powerful virtual server

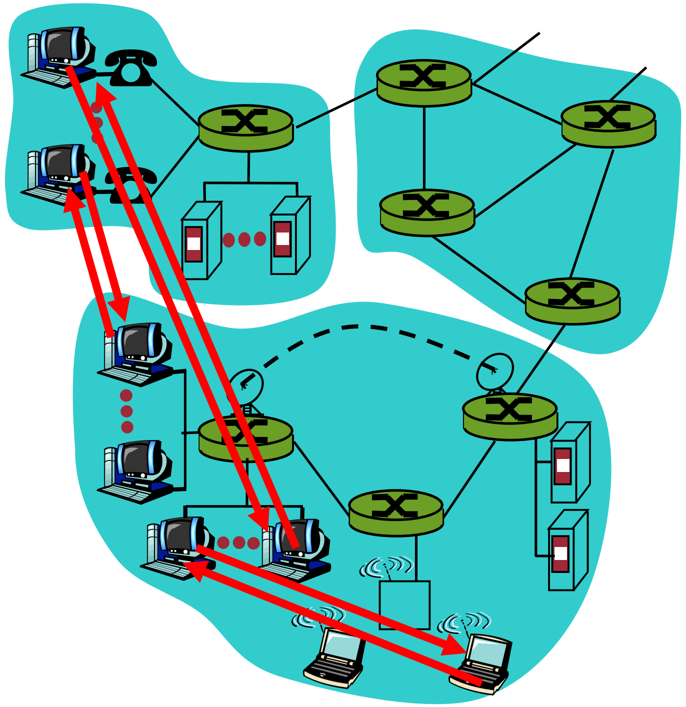
Clients:

- Communicate with server
- May be intermittently connected
- May have dynamic IP addresses
- Clients do not communicate directly with each other

Pure P2P Architecture

- No “always-on” server
- Arbitrary end systems directly communicate
- Peers are connected and change IP addresses
  - example: Freenet and BitTorrent (File Sharing Apps)

Highly scalable but difficult to manage!!!



Hybrid of client-server and P2P

Skype

- Internet telephony application
- Finding address of remote party: centralized server (s)
- Client-client connection is direct (not through server)

Instant messaging

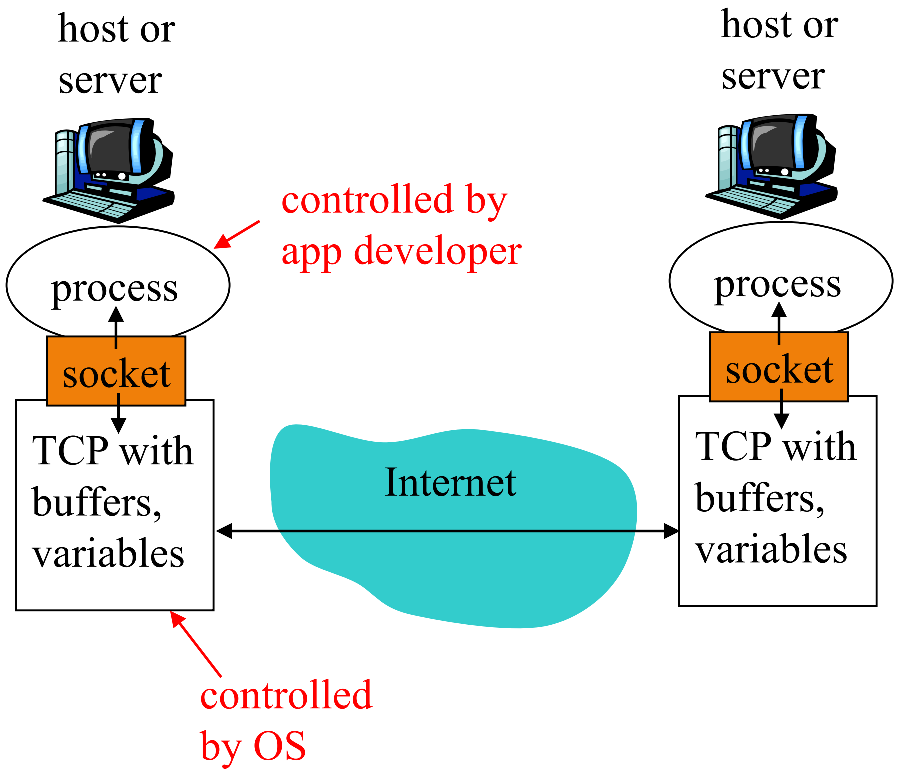
- Chatting between two users is P2P
- Presence detection/location centralized:
  - User registers its IP address with central server when it comes online
  - User contacts central server to find IP addresses of buddies

How Network Applications Communicate?

- **Process** sends/receives messages to/from its **Socket**
  - **Socket** is the interface between the application layer and the transport layer within the host

- Within same host, two **processes** communicate using **inter-process communication**

- **Processes** in different hosts communicate by exchanging **messages**

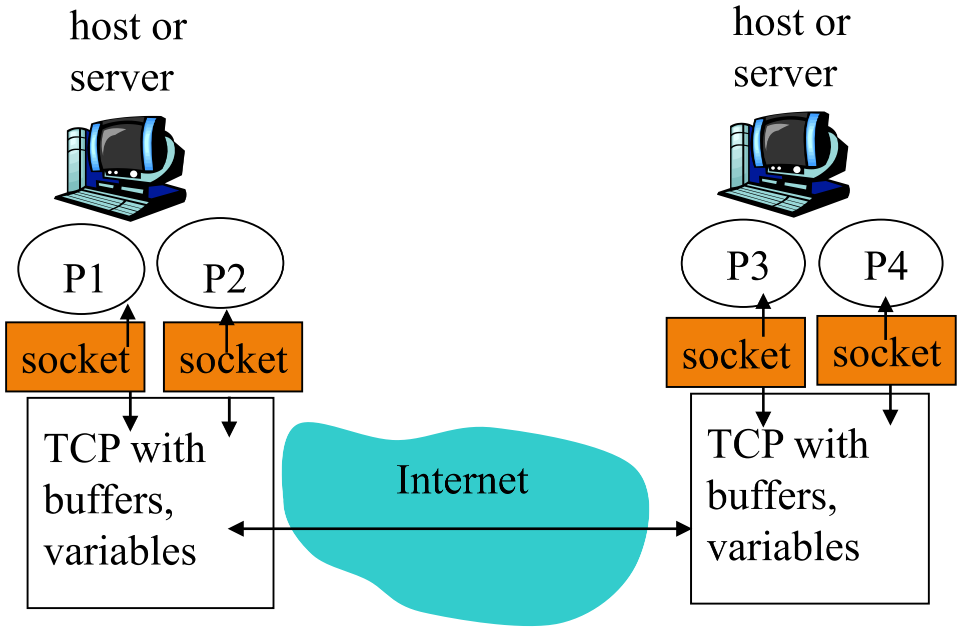

host or server
process
socket
TCP with buffers, variables
controlled by app developer
Internet
controlled by OS
host or server
process
socket
TCP with buffers, variables

How to identify a process running on a machine?

- To receive messages, process must have *identifier*

- IP address of host on which process runs is not sufficient for identifying the process. Why?

- *Process identifier* = IP address + port number
  - e.g., HTTP server: 80, Mail server (SMTP): 25
  - List of well known port numbers is available at **http://www.iana.org**



What transport service does an app need?

- Data loss
  - Some apps (e.g., audio, video) can tolerate some loss
  - Other apps (e.g., file transfer, telnet) require 100% reliable data transfer
- Bandwidth
  - Some apps (e.g., multimedia) require minimum amount of bandwidth to be “effective”
  - Other apps (“elastic apps”) make use of whatever bandwidth they get
  - ex. E-mail, File Transfer
- Timing
  - Some apps (e.g., Internet telephony, interactive games) require low **delay** to be “effective”

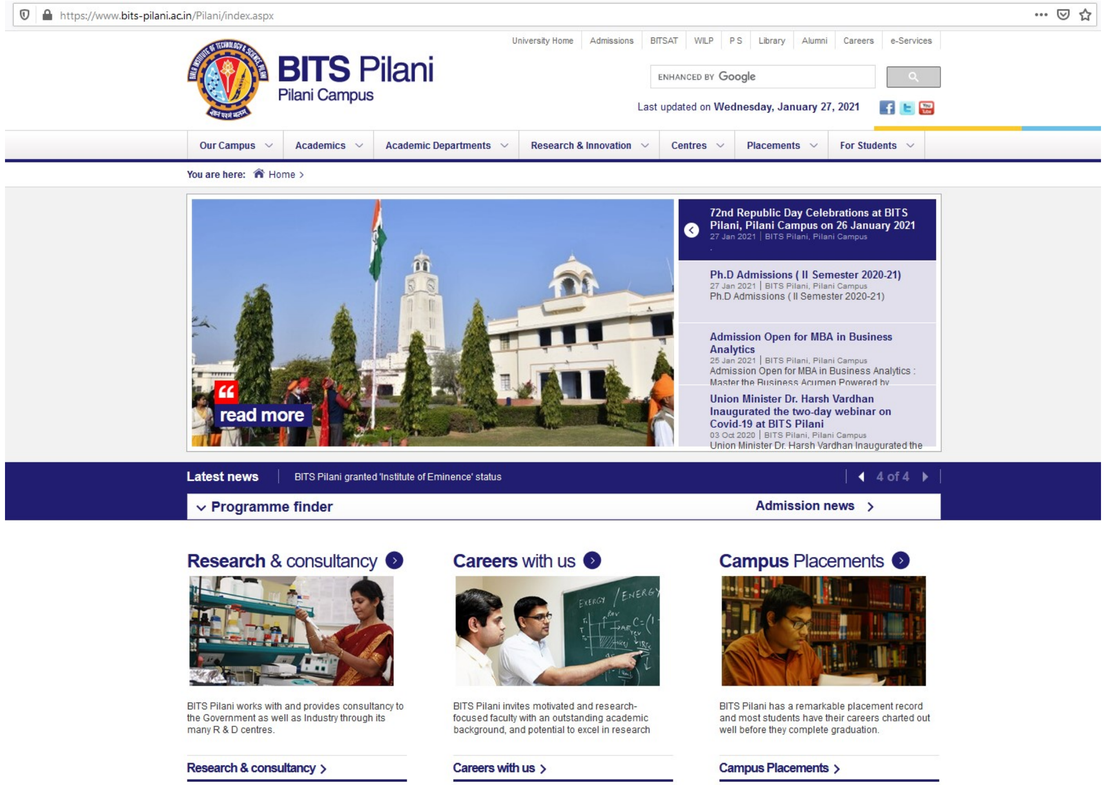
Web and HTTP [1994]

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
- Example URLs:

  https://www.bits-pilani.ac.in/pilani/computerscience/ProgrammesOffered

  https://www.bits-pilani.ac.in/pilani/computerscience/Faculty



Last updated on Wednesday, January 27, 2021

72nd Republic Day Celebrations at BITS Pilani, Pilani Campus on 26 January 2021
27 Jan 2021 | BITS Pilani, Pilani Campus

Ph.D Admissions (II Semester 2020-21)
27 Jan 2021 | BITS Pilani, Pilani Campus
Ph.D Admissions (II Semester 2020-21)

Admission Open for MBA in Business Analytics
25 Jan 2021 | BITS Pilani, Pilani Campus
Admission Open for MBA in Business Analytics : Master the Business Acumen Powered by

Union Minister Dr. Harsh Vardhan Inaugurated the two-day webinar on Covid-19 at BITS Pilani
03 Oct 2020 | BITS Pilani, Pilani Campus
Union Minister Dr. Harsh Vardhan Inaugurated the

BITS Pilani granted 'Institute of Eminence' status

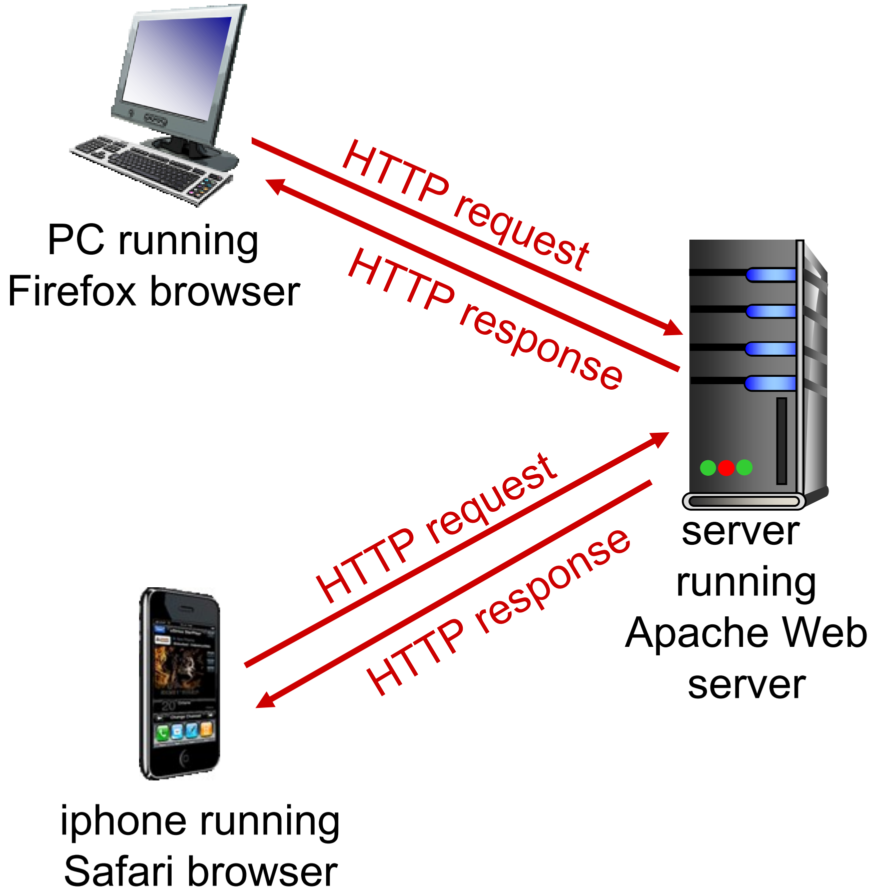
BITS Pilani works with and provides consultancy to the Government as well as Industry through its many R & D centres.

BITS Pilani invites motivated and research-focused faculty with an outstanding academic background, and potential to excel in research

BITS Pilani has a remarkable placement record and most students have their careers charted out well before they complete graduation.

HTTP Overview [.1]

- Types of messages exchanged
  - e.g., request, response
- Message syntax:
  - What fields in messages & how fields are delineated
- Message semantics
  - Meaning of information in fields
- Rules for when and how processes send & respond to messages

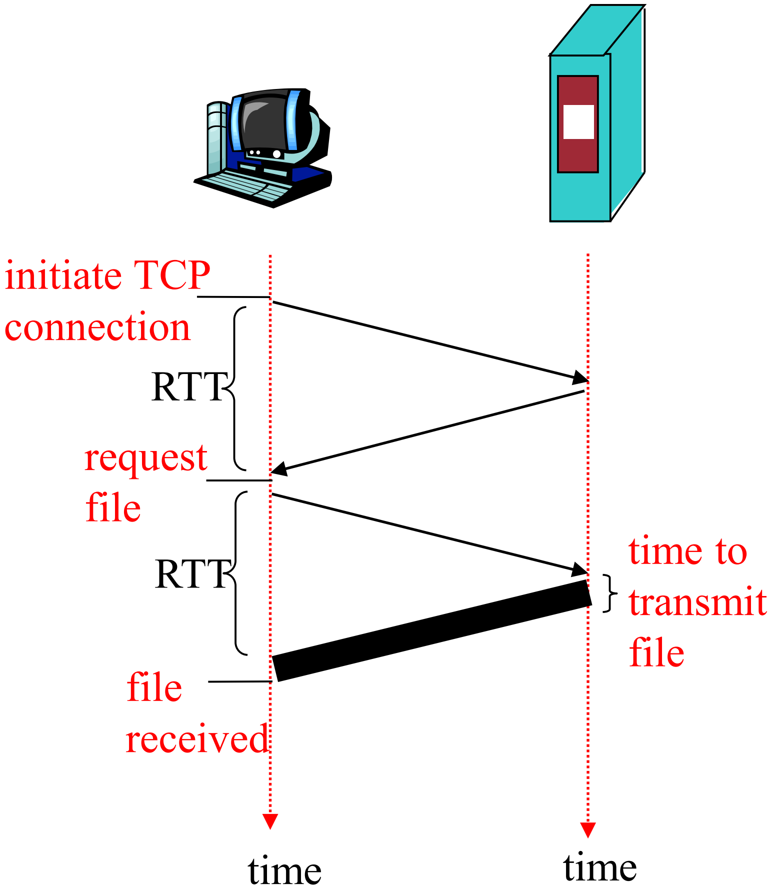


PC running Firefox browser
HTTP request
HTTP response
server running Apache Web server
HTTP request
HTTP response
iphone running Safari browser

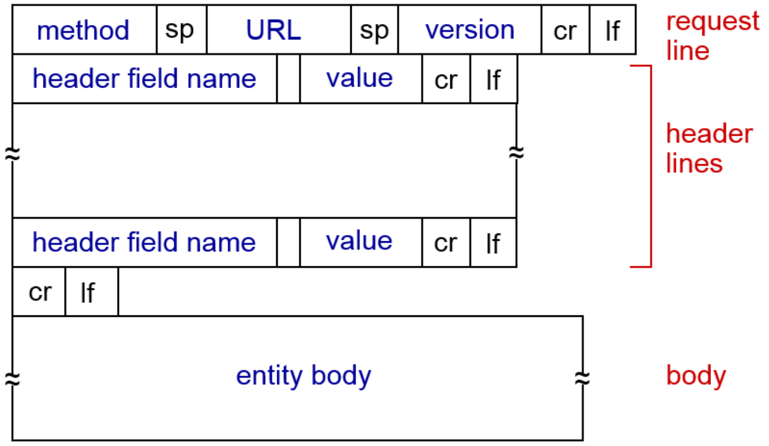
HTTP Overview [..2]

Uses TCP:

- Client initiates TCP connection (creates socket) to server, port 80
- Server accepts TCP connection from client
- HTTP messages exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed



HTTP Request Message



carriage return character

line-feed character

request line (GET, POST, HEAD commands)

header lines

carriage return, line feed at start of line indicates end of 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
data data data data data ...
```

Response Message

status line
(protocol
status code
status phrase)

```
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 ...
```

header
lines

data, e.g.,
requested
HTML file

HTTP Response status 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

- the **HTTP** version used in the request is not supported by the server.

Working of HTTP

- Let’s assume a web page consists of a base HTML file and 10 JPEG images.
  - https://www.bits-pilani.ac.in/Pilani/SustainableEnvironment

HTTP Connections

Non-persistent HTTP

- At most one object is sent over a TCP connection
- HTTP/1.0 uses non-persistent HTTP

Persistent HTTP

- Multiple objects can be sent over single TCP connection between client and server.
- Persistent with Pipeline vs. Persistent without Pipeline
- HTTP/1.1 uses persistent connections in default mode

HTTP Method Types

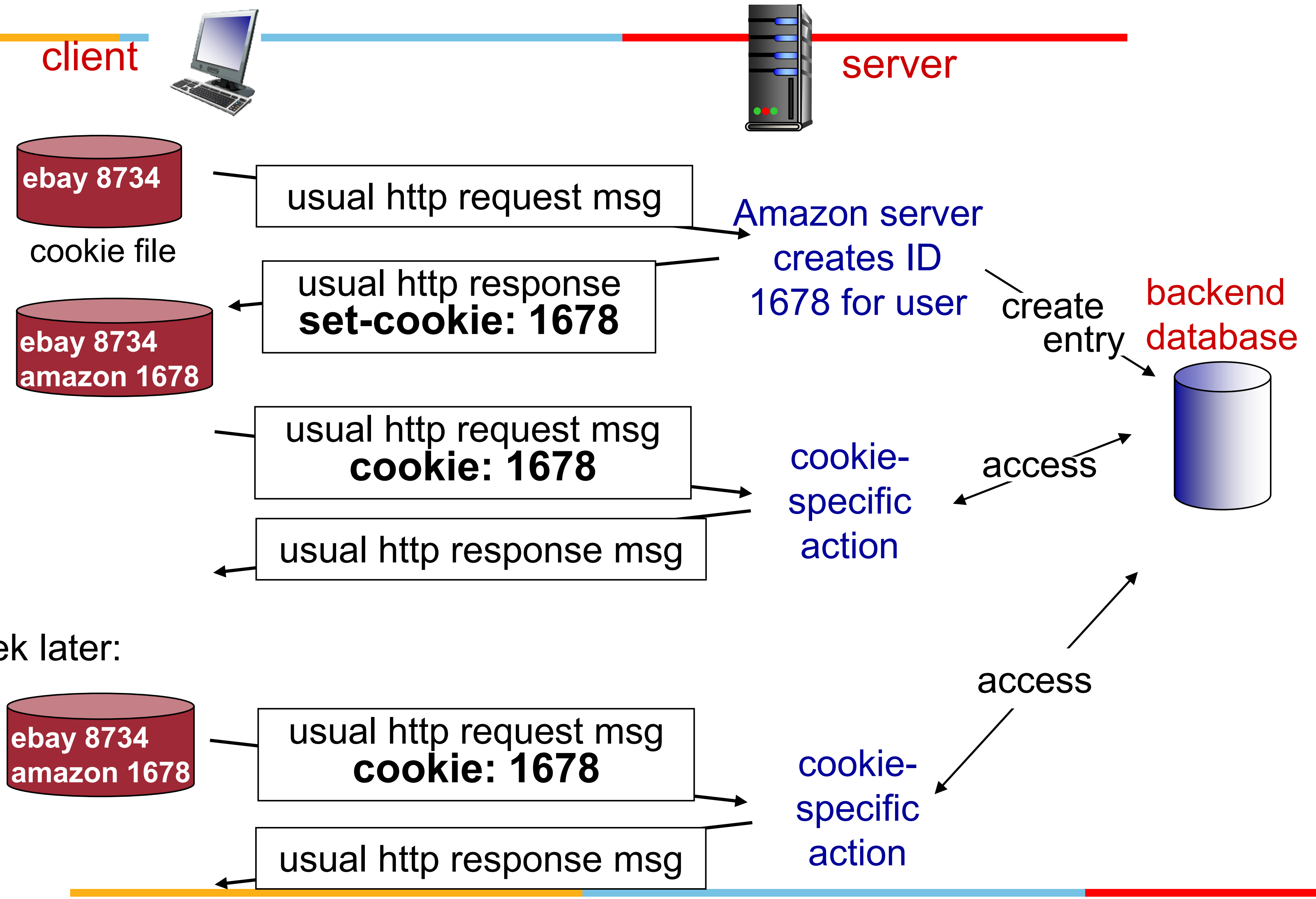
HTTP/1.0:

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

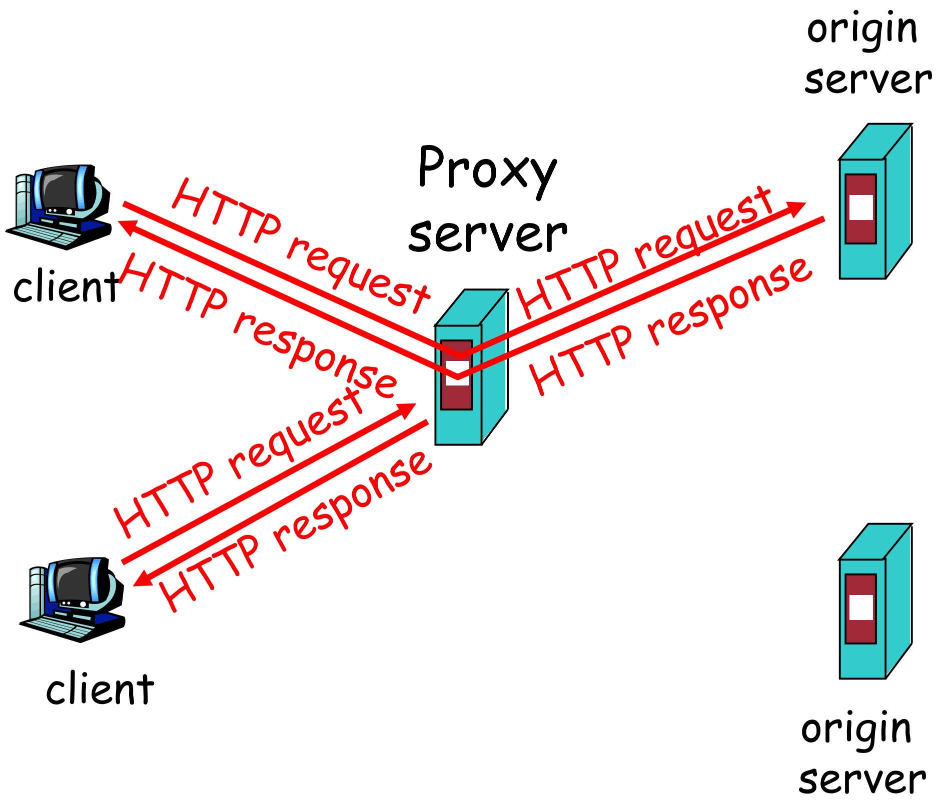
HTTP/1.1:

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

State in HTTP using "Cookies"



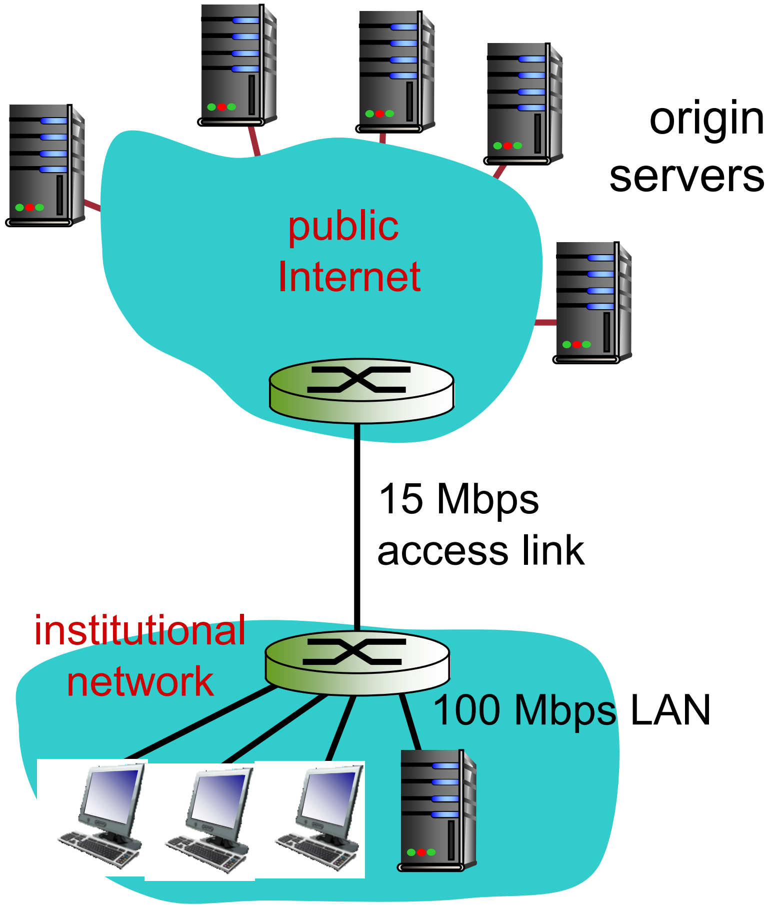
Web Caches (aka Proxy Server)



Proxy Server Example [.1]

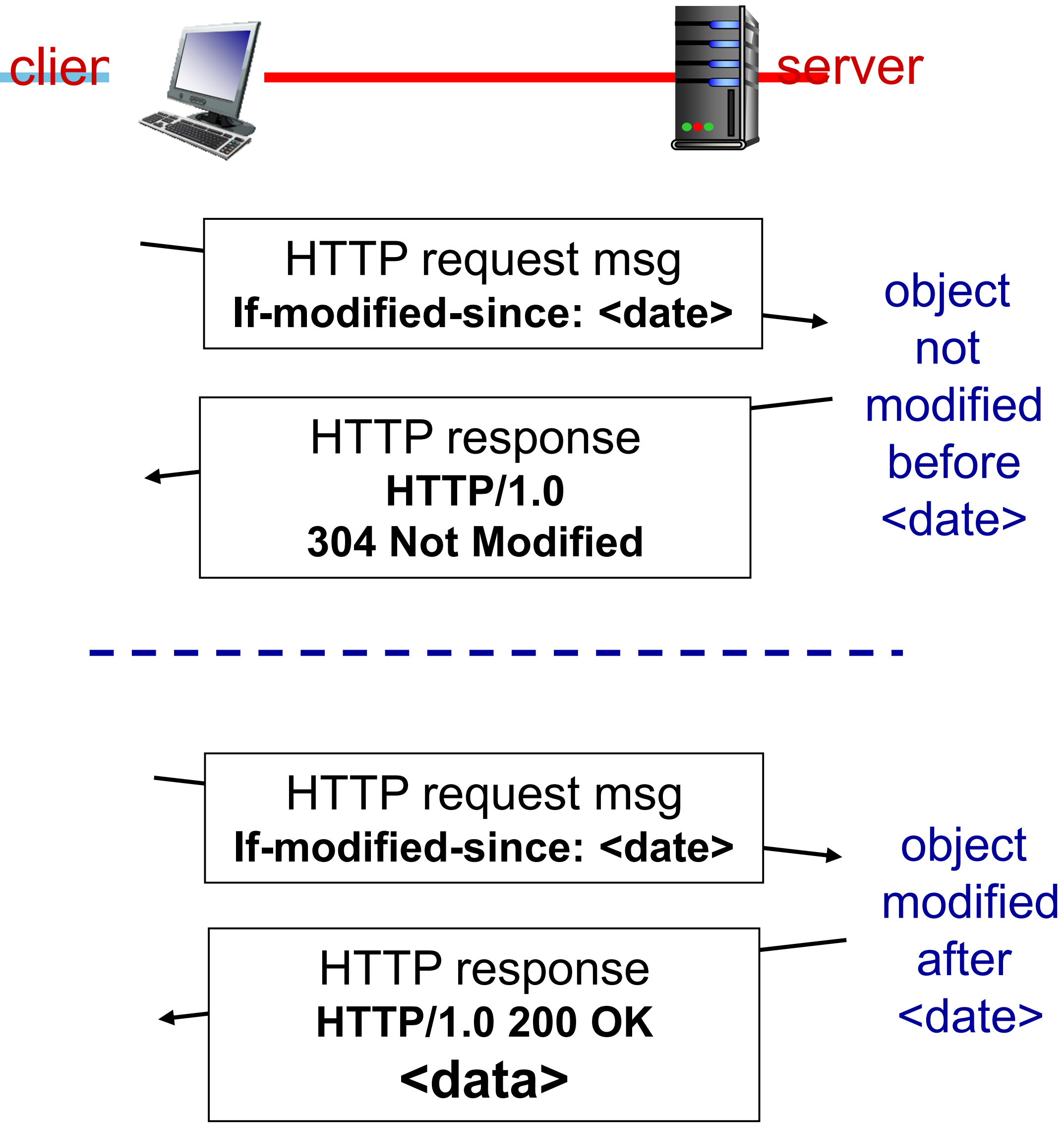
Assumptions:

- avg object size: 100K bits
- avg request rate from browsers to origin servers:15 req/sec
- avg data rate to browsers: 1Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 15 Mbps



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`



HTTP/2 [Proposed in 2015]

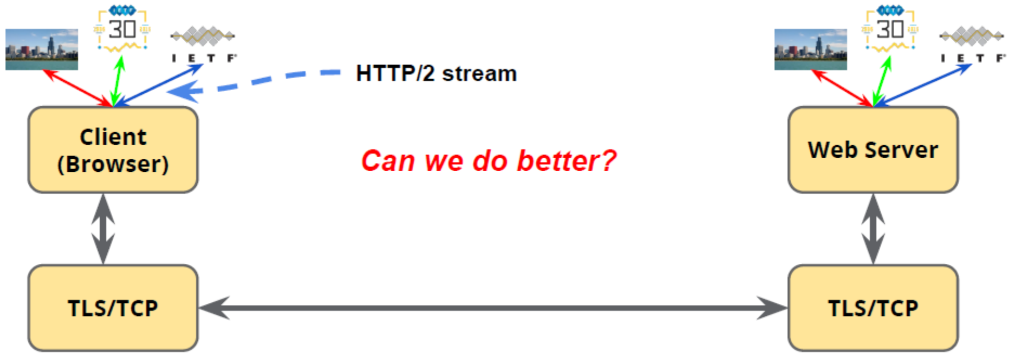
- Motivation
  - To improve internet user experience and effectiveness
  - Webpages comprise resource-intensive multimedia content
  - To make it more secure, reliable with improved performance
- It is an extension to its predecessor not replacing the older one
- Limitations of HTTP1.1
  - It processes only one outstanding request per TCP connection
  - Forcing browsers to use multiple TCP connections to process multiple requests simultaneously
  - HTTP1.x used to process text commands which makes it slower

HTTP/2Feature: Stream Multiplexing

- What is ***stream***?
  - Bi-directional sequence of text format frames sent over the HTTP/2 protocol exchanged between the server and client

- HTTP/1 is capable of transmitting only one stream at a time
  - Receiving large amount of media content via individual streams sent one by one is inefficient and resource consuming

- HTTP/2 allows transmission of parallel multiplexed requests and responses
  - A binary framing layer is created
  - This layer allows client and server to disintegrate the HTTP payload into small, independent and manageable interleaved sequence of frames
  - This information is then reassembled at the other end



HTTP/2 stream

Client (Browser)

Can we do better?

Web Server

TLS/TCP

TLS/TCP

HTTP/2 Feature: Server PUSH

- It allows the server to send additional cacheable information to the client that isn’t requested but is anticipated in future requests.

- This mechanism saves a request-respond round trip and reduces network latency.

Thank You!