CS3.301 Operating Systems and Networks

Networking - Application Layer

Karthik Vaidhyanathan

https://karthikvaidhyanathan.com

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

SERC
Software Engineering Research Centre

Acknowledgement

The materials used in this presentation have been gathered/adapted/generate from various sources as well as based on my own experiences and knowledge -- Karthik Vaidhyanathan

Sources:

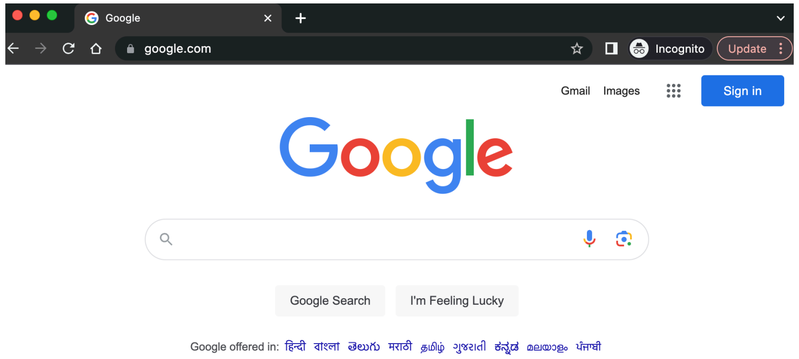
- Computer Networks, 6e by Tanebaum, Teamster and Wetherall
- Computer Networks: A Top Down Approach by Kurose and Ross
- Instructor Materials on Computer Networks: A Top Down Approach, Kurds and Ross

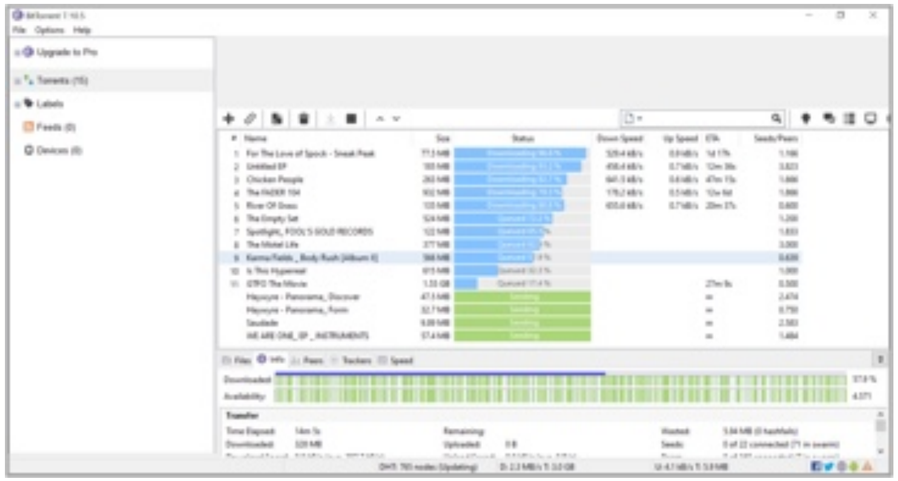
What are the different Network Applications that you come across?

- Streaming stored video (eg: Netflix, Youtube)
- VOIP apps (eg: Skype)
- Web browser (eg: Chrome, Firefox)
- Social Networking (eg: Facebook)
- Instant Messaging (eg: Whatsapp)
- P2P File Sharing (eg: Bit Torrent)
- Real-time Video Conferencing (eg: Zoom)
- ….

Many Processes run on the OS

Rather applications!





Browser - Type any URL and you get the page - How?

The process on the other side needs to provide the page

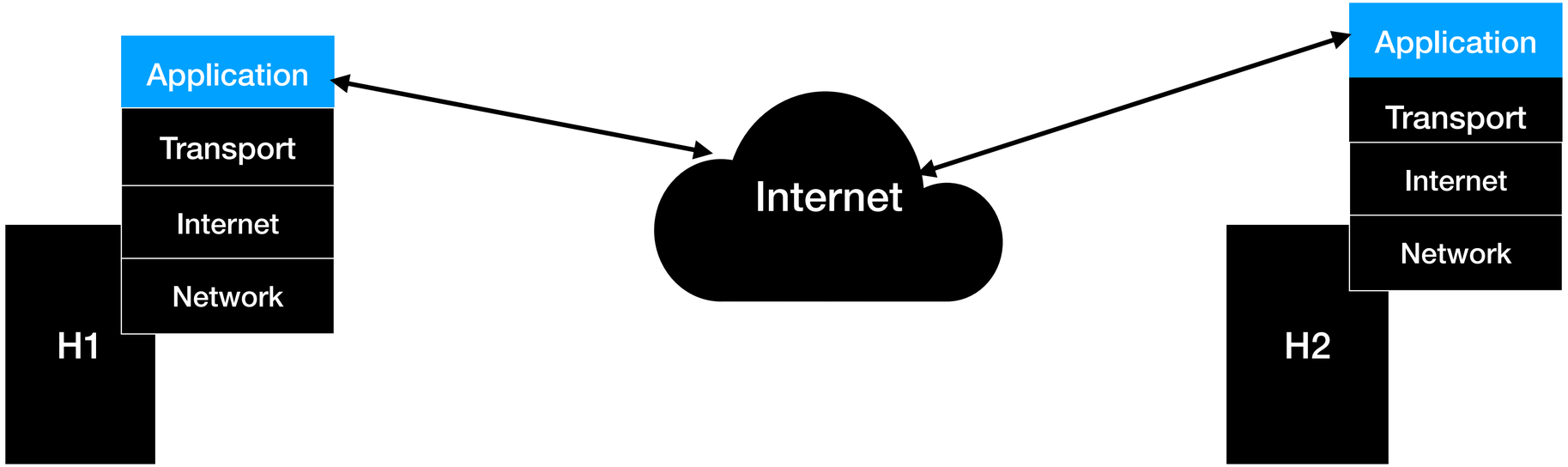
Large videos may have to be served to different users

There are also programs like torrent -

How does that work?

Is transport layer enough?

- Process communicates over the network
- The real communication happens through the application layer



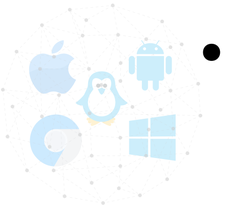
Building a Network Application

- **Write programs:**
  - That can run on different systems
  - Communicate over the network
  - Eg: Web server communicates with browser (Apache)
- **Core-network device software:**
  - Network devices do not run user applications

httpd://apache.org

Network Application Architectures

- From a developer perspective, network architecture is fixed
  - Application architecture is something that can be controlled
- Two main types are available: Client-server and P2P
- **Client-Server**
  - Host that is always on, serving the clients - Server
  - Host that requests for services - Clients
  - Eg: Web client and Web server



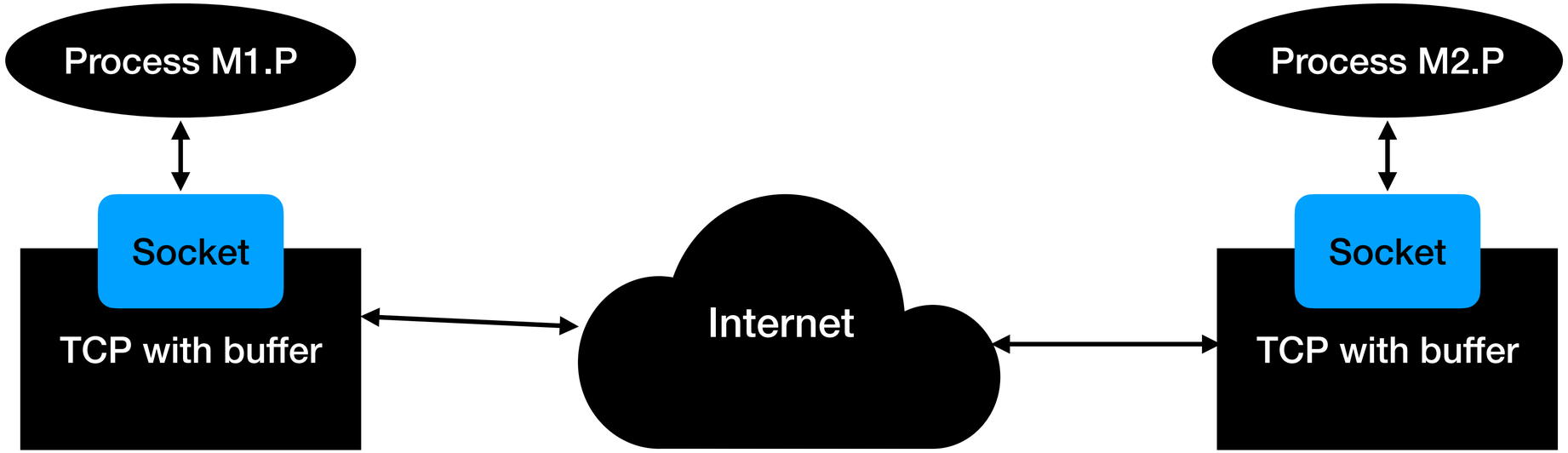
Network Application Architecture

- Server has mostly a fixed IP address (or rather domain)
- Clients can always connect by sending packet to server IP address
  - Eg: Web browser, FTP, e-mail
- Often a single server may not be enough - leverage data centres
- Data centres have hundreds or thousands of servers that must be processed and maintained - **Energy!!**
  - **Spare a thought for the carbon foot print - Can we do something?**

Network Application Architecture

- **Peer-to-Peer (P2P) Architecture**
  - There is minimal or no reliance on dedicated servers (No always on Servers)
  - Peers communicate among each other
  - Peer technically acts as a client and a server
  - Not owned by any service provider and does not pass through dedicated server
  - Advantage: Self-scalability, cost effective
  - Challenges: Security, Performance, reliability, etc.
  - Eg: Bit Torrent, Skype

Lets take a step back



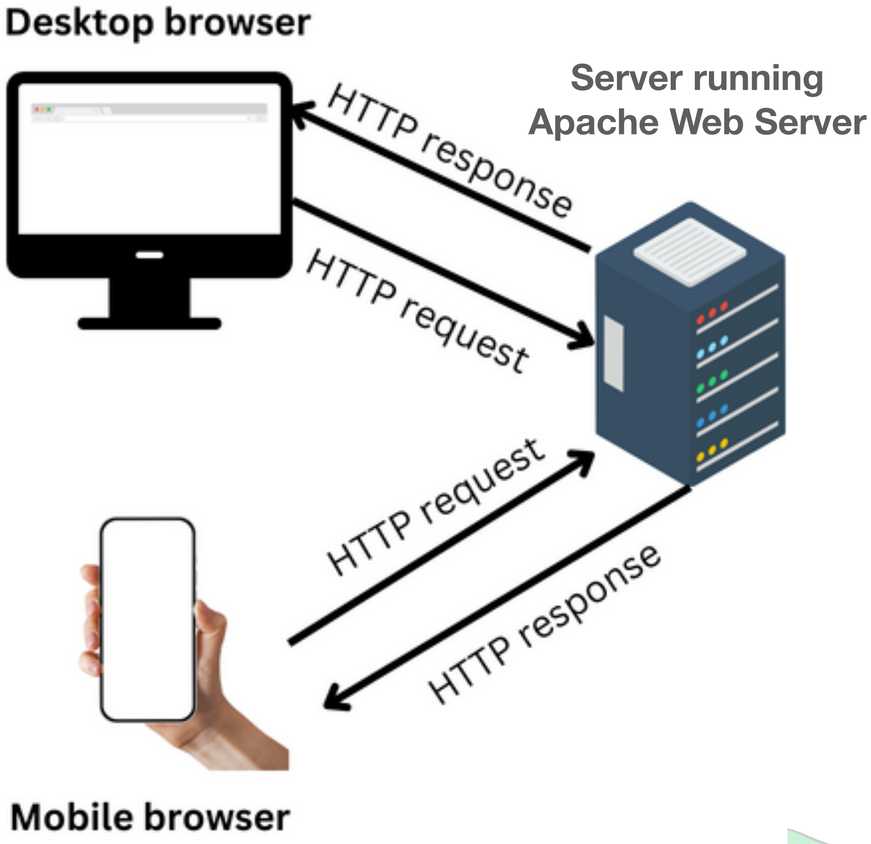
- Process sends to and receives messages from network using socket interface
- The developer has more control on the application side than socket
- Application just sends message with IP address and port - Rest other layers

Application Layer Protocol
What does it mean?

- Application layer protocol defines the following:
  - Types of message exchanges (request/response)
  - Syntax of various message types
  - Semantics of the fields
  - When and how the process sends and responds to messages
- Some protocols: HTTP, SMTP, DNS, etc.

HTTP: Hyper Text Transfer Protocol

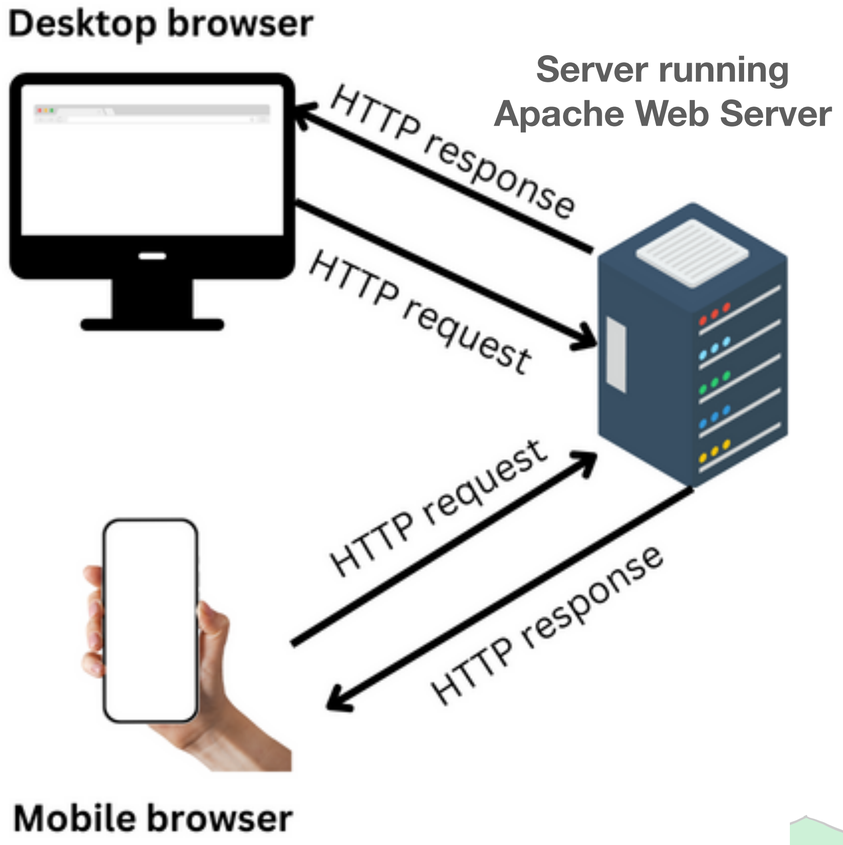
- Application layer protocol of the web
- Implemented in two programs: Client and Server
- HTTP protocol defines structure of messages
- **Client:** browser that sends requests, receives and displays web objects (using HTTP protocol)
- **Server:** Web server that sends objects in response to requests (using HTTP protocol)



https://iiit.ac.in/samplePage.html
Host name: iiii.ac.in
Object: samplePage.html

HTTP: Hyper Text Transfer Protocol

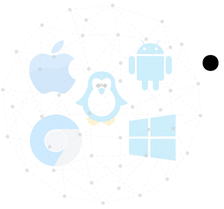
- HTTP uses **TCP at transport layer**
  - Client initiates TCP to server (Port 80)
  - Server accepts TCP connection from client
  - HTTP messages are exchanges
  - Connection is closed
  - HTTP is reliable - why?
- HTTP is **stateless**
  - Server maintains no information about client



Types of HTTP Connection

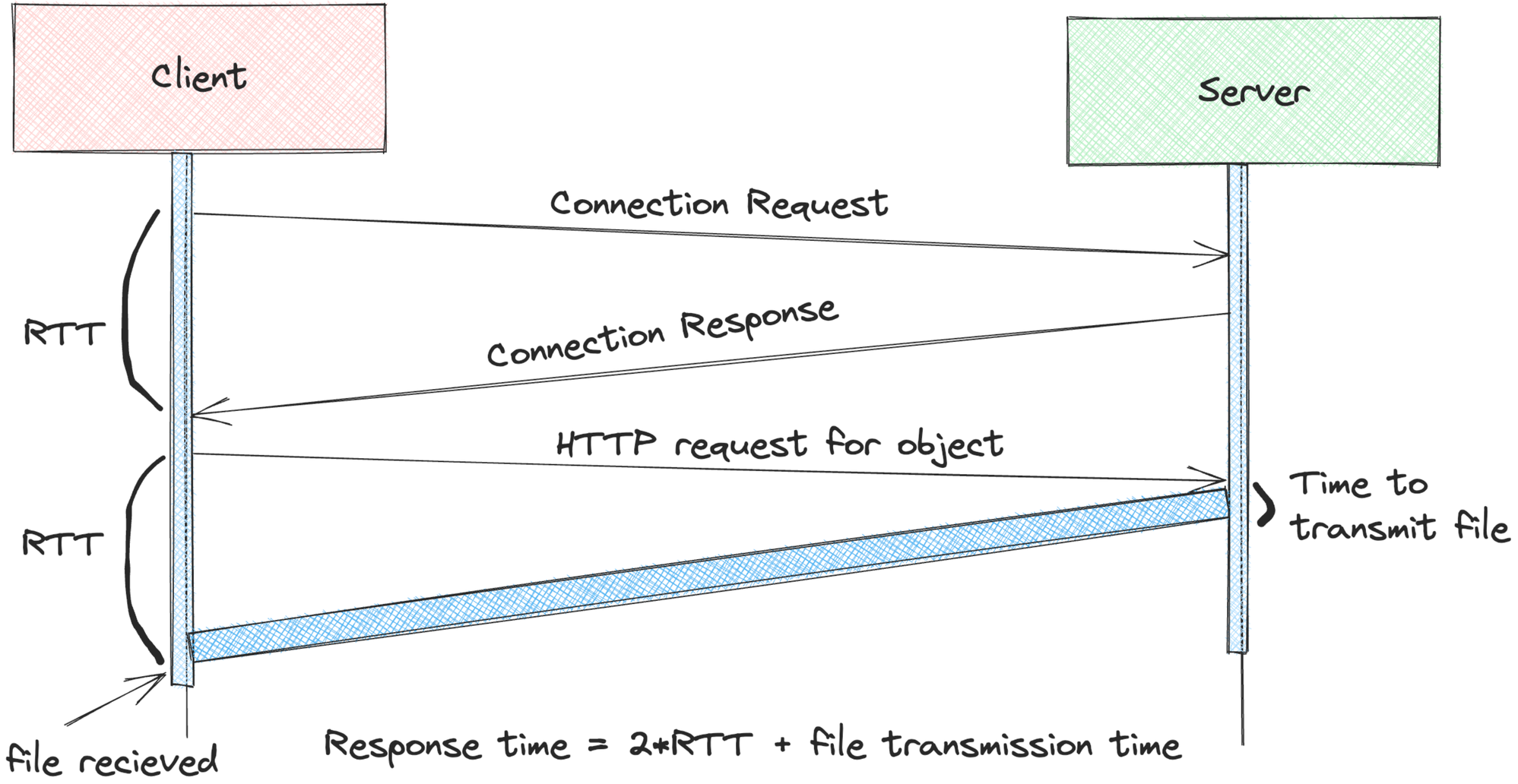
Persistent and Non-Persistent Connections

- **Non-persistent HTTP Connection (HTTP/1.0)**
  - For every connection, the client has to create a request (one page may require multiple objects)
  - Downloading multiple objects requires multiple connections
  - Opened connection is closed after each request response
- **Persistent HTTP Connection (HTTP/1.1)**
  - One connection for all the objects
  - The opened connection is maintained



Non Persistent Connection: Response time

HTTP 1.0



HTTP Request Message

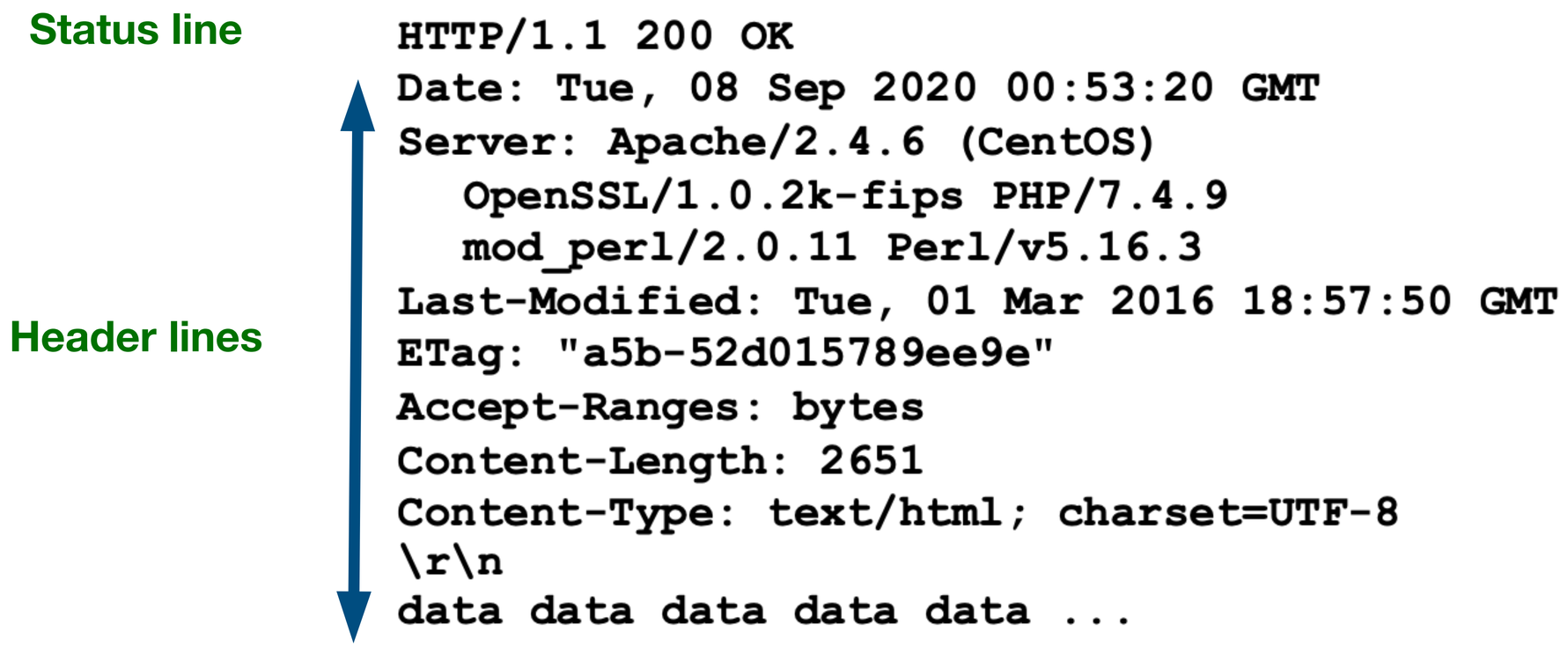
- Two types of HTTP messages: **request** and **response**
- HTTP request message: ASCII (human-readable format)
- HTTP supports different methods - GET, POST, PUT, HEAD, DELETE
- Request line: Method, URL and Version

Request line

Header lines

```
GET /index.html HTTP/1.1\r\n
Host: www.iiit.ac.in/centers \r\n
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
   10.15; rv:80.0) Gecko/20100101 Firefox/80.0 \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
Connection: keep-alive\r\n
\r\n
```

HTTP Response Message

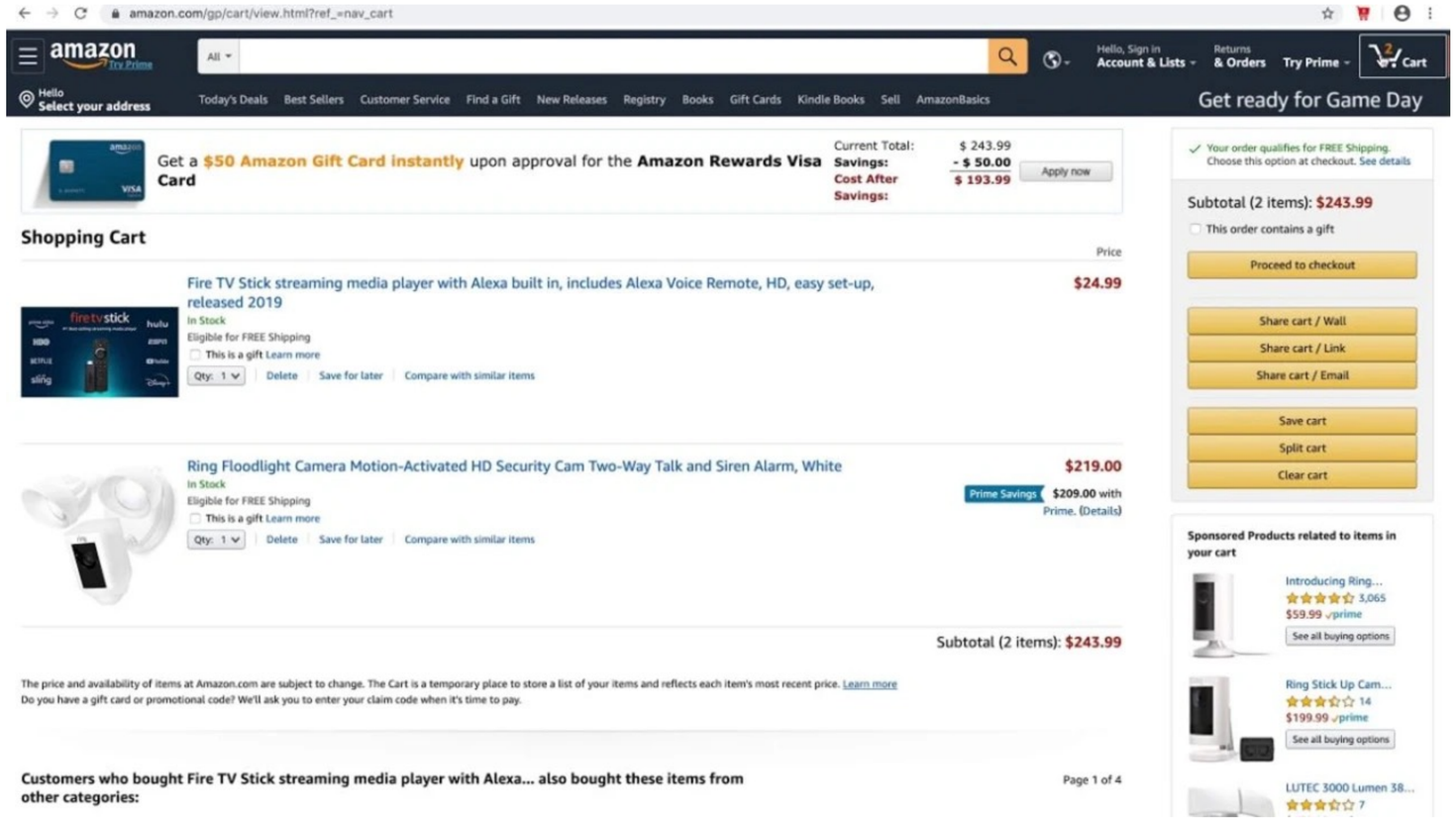


- Status line: version and status code
- Last-modified in header can help in caching - **How?**

HTTP Status Codes

Status Code	Description
200	Request succeeded, status ok
301	Moved permanently, requested object moved, new location in the location: field (header)
400	Bad request, request message not understood by server
404	Document does not exist
505	HTTP version not supported

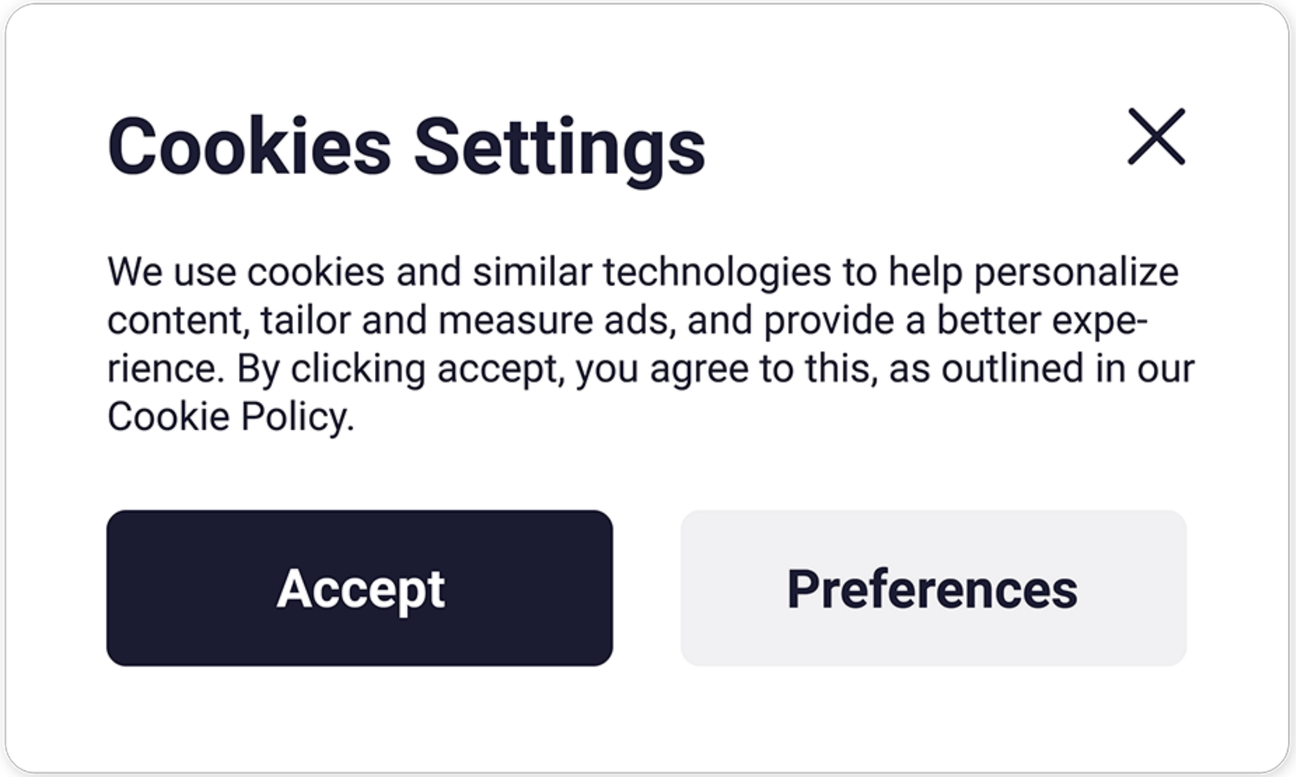
Can this happen in stateless protocol



- Even after closing and opening the website, the cart has the items
- How's website like Amazon, ebay, etc able to do this?

Image source: https://en.softonic.com/articles/how-to-use-share-a-cart-for-amazon

Why not cookies!



Source: https://boom visibility.com

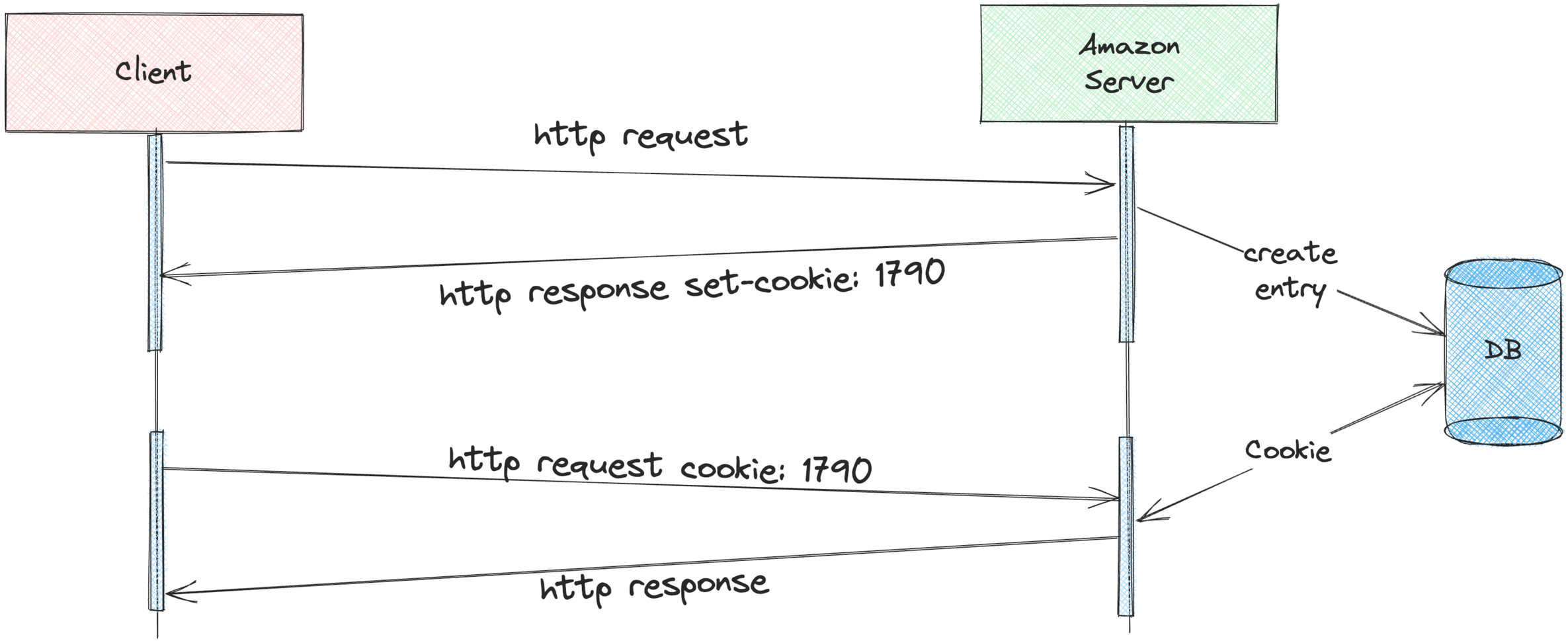
Maintaining User/Server State: Cookies

- HTTP server is **stateless**
- Helps in supporting thousands of simultaneous connections
- Each connection is treated separately
- Website may want to identify users for various reasons
  - Keep session information
  - Recommend similar products

Cookies

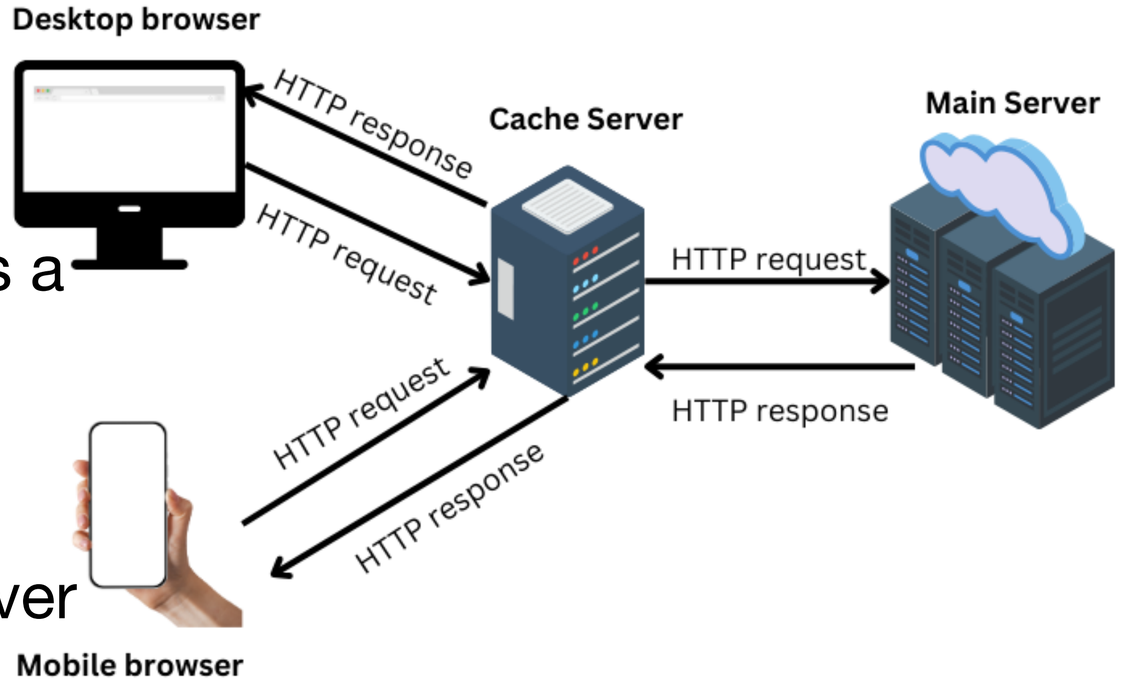
- HTTP header consists of information for cookies
- Consists of four components:
  - Cookie header line in HTTP response
  - Cookie header line in HTTP request message
  - Cookie file kept in clients system
  - Backend database on the server/website
  - Cookies can be used to create user session on top of HTTP - **Can be invasion of Privacy!**

Cookie Illustration



Web Caches

- Not every time we need to access the main (original) web server
- We can have proxy server that satisfies request on behalf of main server
- Browser can be controlled to point towards a cache (mentioned in response header)
  - If cache hit: return object from cache
  - Else cache request object from main server and returns it

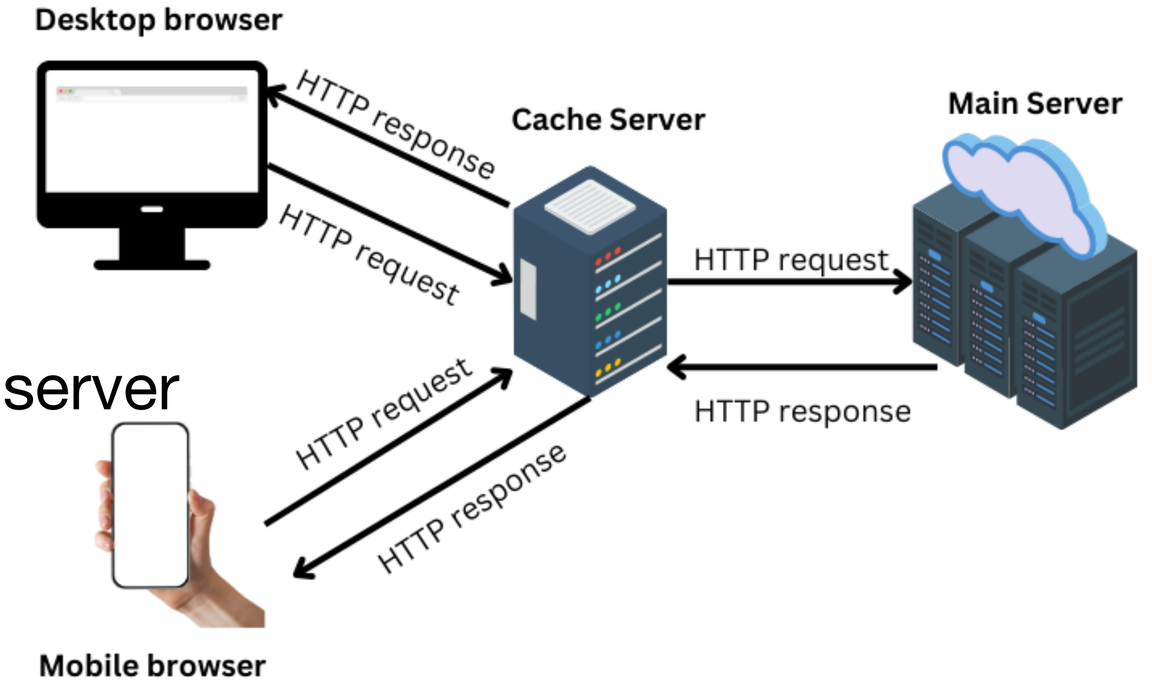



```
Cache-Control: max-age=<seconds>
```

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

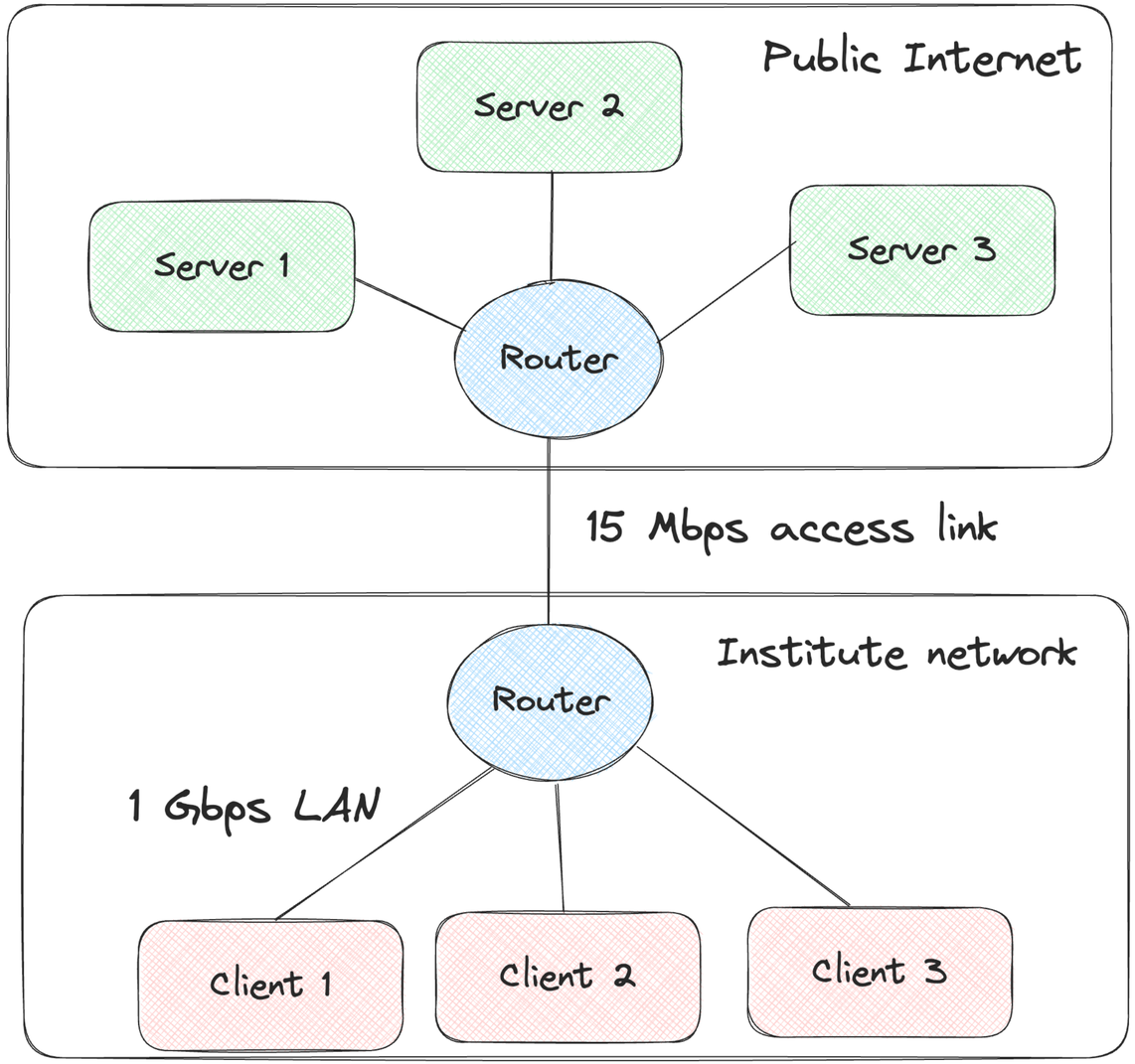
Caching Benefits

- Reduce response time for a client request
  - Cache is closer to client
  - There may be bandwidth constraints
- Reduce traffic on the internet and to main server
  - Significantly improve performance
- Internet is dense with caches
  - Effective delivery of content



University Scenario

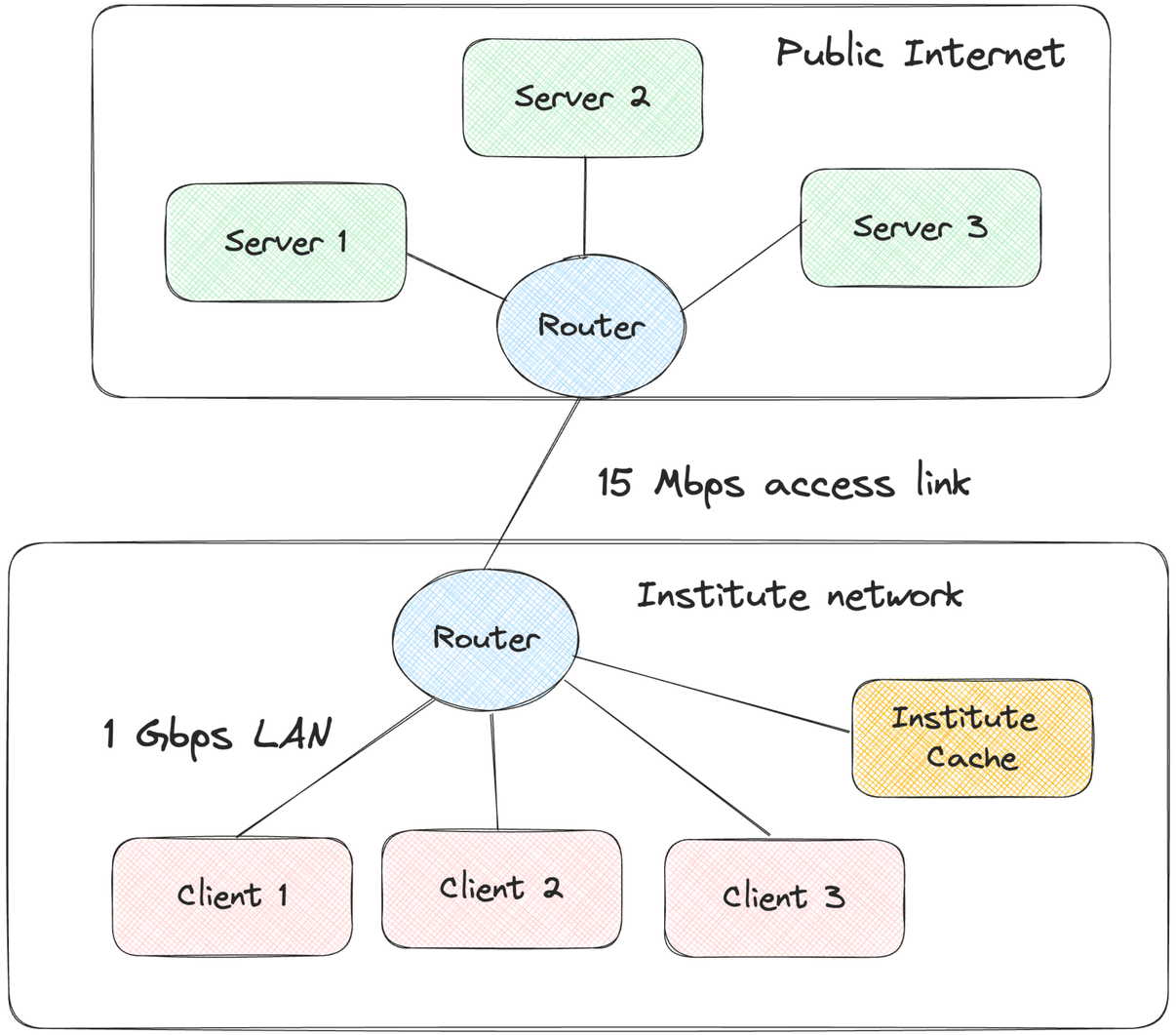
- Assume that:
  - servers to router in public internet incurs ~ 2 sec delay
  - Data is 1Mbits
  - Request interval from client ~ 15 requests/ second
- End-to-end delay = Internet delay + access link delay + LAN delay
  - ~ 2 sec + ~ 1 min + ~ u sec (High!!)



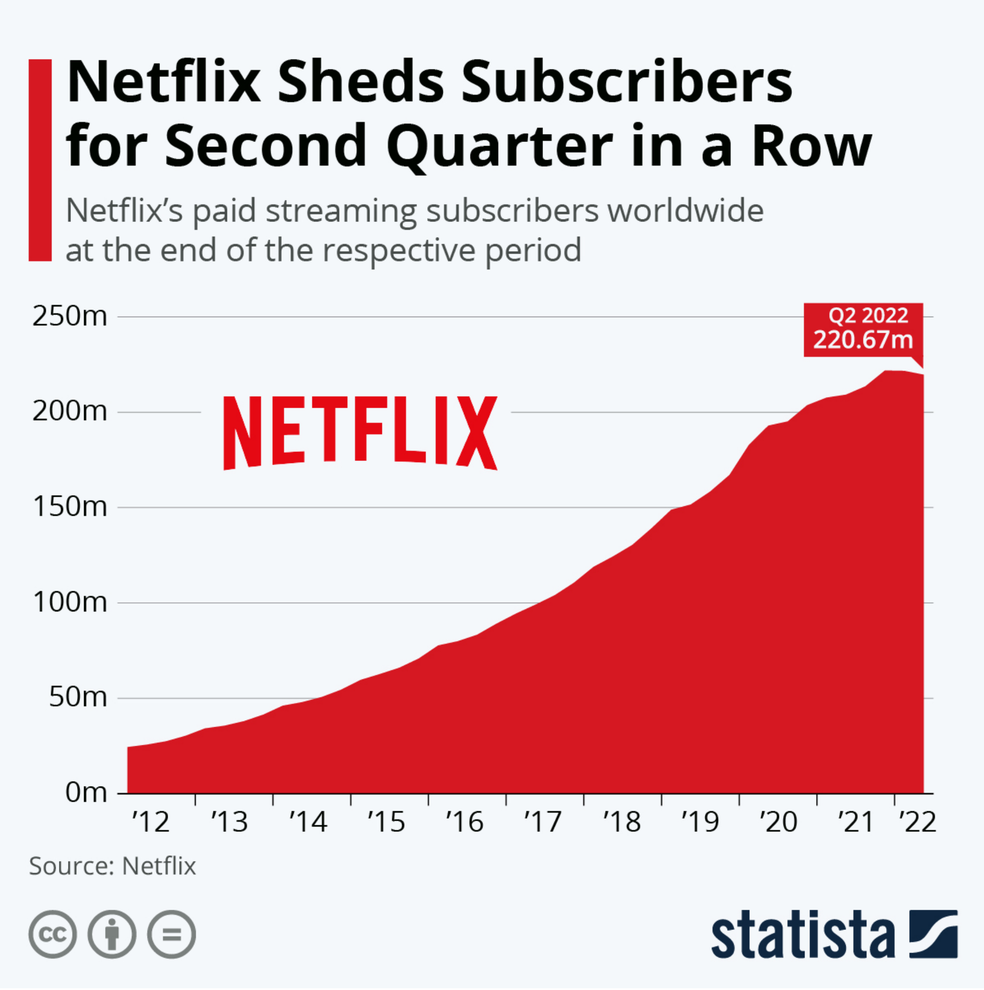
University Scenario

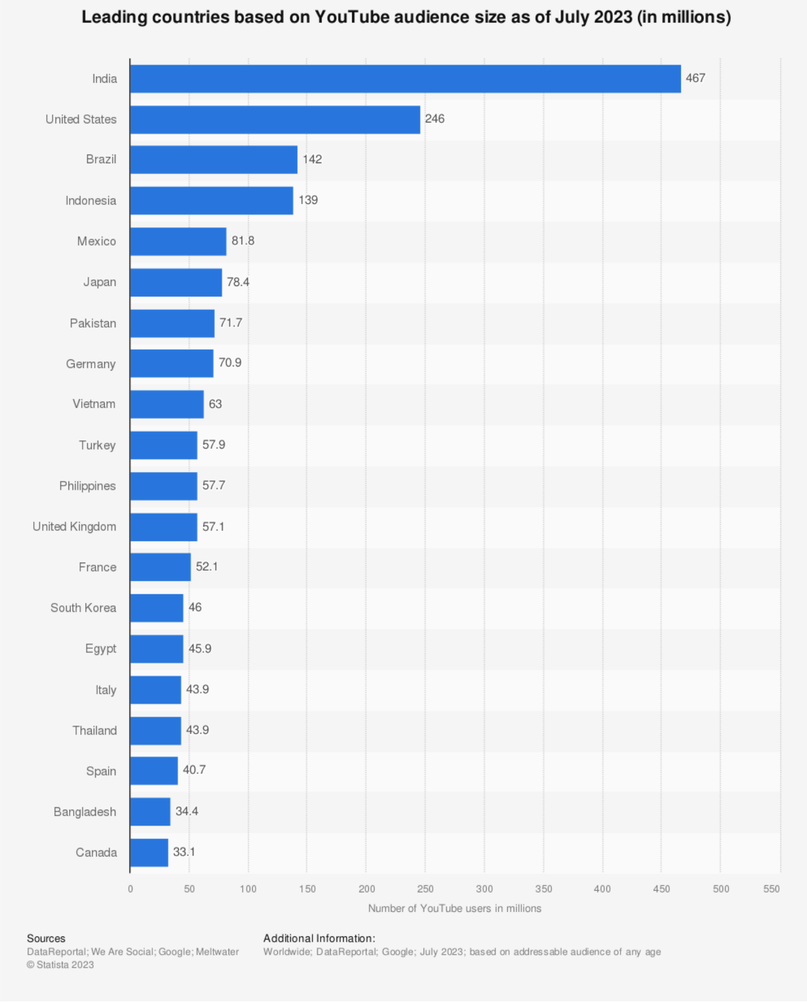
- Easy way out is to increase bandwidth
  - Costly solution, request rate may increase
- Introduce Cache server at Institute level
  - Even if there is 40% hits => significant improvement
- Delay = 0.6*(2) + (0.4)*(0.01)

  = ~1.2 secs
- Cheaper solution, lower end-to-end delay



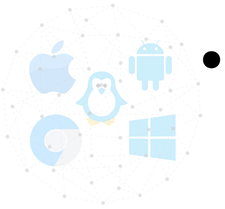
Web Cache has broader Applications





Content Distribution Networks (CDN)

- Many internet companies are distributing on-demand video streams to millions of users on daily basis
  - Youtube (~2.7 Billion users), Netflix (~238 Million subscribers), ..
  - Distributed across the world - Having one large data centre may not work - Why?
- To support growing demand, scale - distribute over different CDNs
  - Servers are distributed among different geographical locations
  - Requests are redirected to CDNs (private or third party)



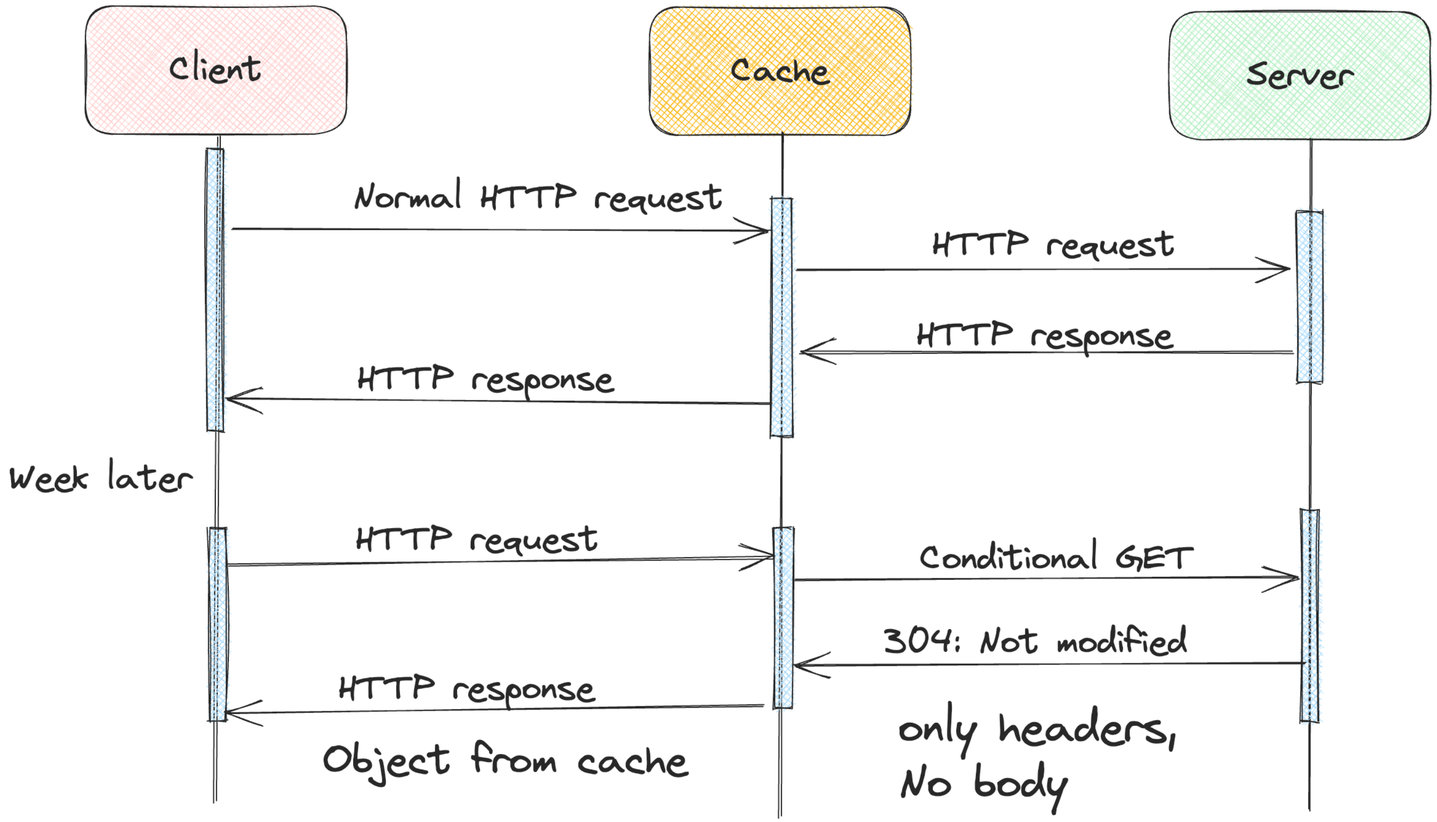
Content Distribution Networks (CDN)

- CDNs adopt two different server placement strategies
  - **Enter Deep:** Deploy server clusters in all access ISPs
    - High maintenance, higher throughput and lower delays
  - **Bring Home:** Building larger cluster at smaller number of sites (eg: 10s)
    - Lower maintenance, lower throughput and higher delays
- Client needs to pick a cluster - Geographically closest or using real-time information (dynamic choices based on heuristics)

Conditional GET

Cache does not contain originals

- If things are kept in cache, the copy of original may become stale!
- HTTP provides mechanism to verify if object is up-to-date (conditional GET)
- HTTP request header - use field “If-modified-since: “
- Server provides “Last-Modified” in the response header



HTTP 1.1 and Beyond

- **HTTP 2.0**
  - Standardised in 2015
  - As of 2020, 40% of top 10 million websites support HTTP/2
  - Primary goals
    - Enable request response multiplexing over single TCP
    - Request prioritisation
    - Server push
    - Compression of HTTP header fields
- **HTTP 3.0** underway (drafts as of 2020)

Domain Name System (DNS)

- Students have roll numbers but addressed by names
  - Similar is the case with citizens - Aadhar, SSN
- What about the internet?
  - Google.com, Facebook.com, YouTube.com - Are they enough?
  - How to locate them? Eg: iiit.ac.in - Provides info that it is in India but exactly where?
  - IP address can help - They have a hierarchical representation

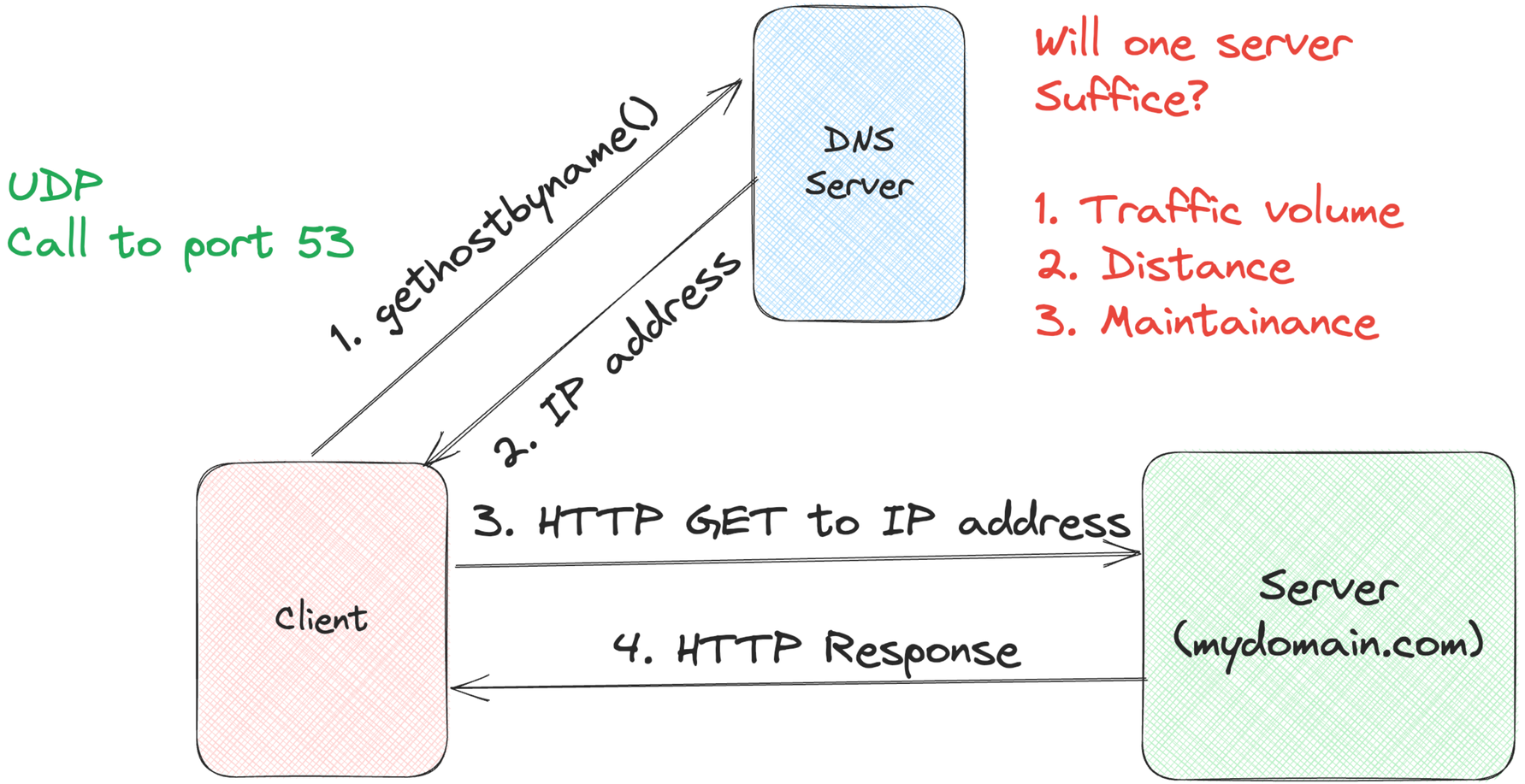
Domain Name System (DNS)

- Directory service of the internet that translates hostnames to IP addresses
- People prefer mnemonic names, routers prefer IP addresses
- Application layer protocol, allows hosts to query a distributed database
- DNS servers are UNIX machines running **Berkley Internet Name Domain** (BIND) software
- DNS runs over **UDP** and uses port **53**
- Leveraged by protocols like HTTP and SMTP to translate hostnames to IP addresses

Domain Name System (DNS) Services

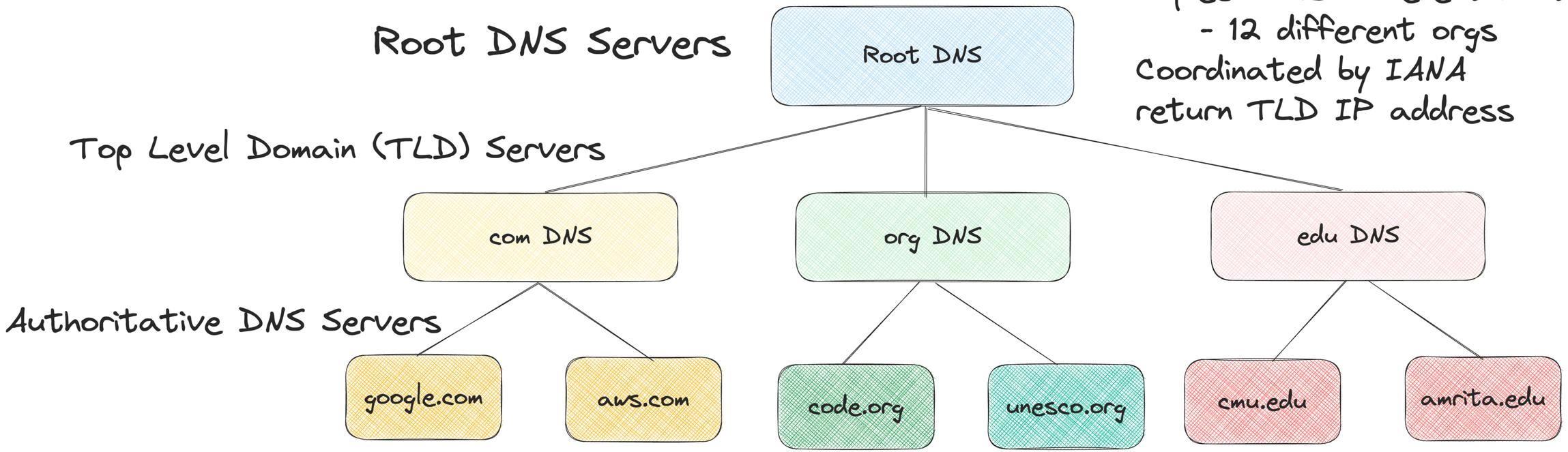
- **Host aliasing**
  - Same hosts can have multiple aliases, resolve the names (get canonical names of host)
  - Eg: 1231242s-us-west.aws.com: mydomain.com, e-markt.com
- **Mail Server Aliasing**
  - Mail servers may also have aliases
  - DNS can provide canonical names of mail server to mail clients
- **Load distribution**
  - Perform distribution among replicated servers
  - Eg: Amazon may be replicated to multiple servers (IP address), keep giving back IP in different order

DNS: How does it work?



DNS: Distributed Hierarchical Database

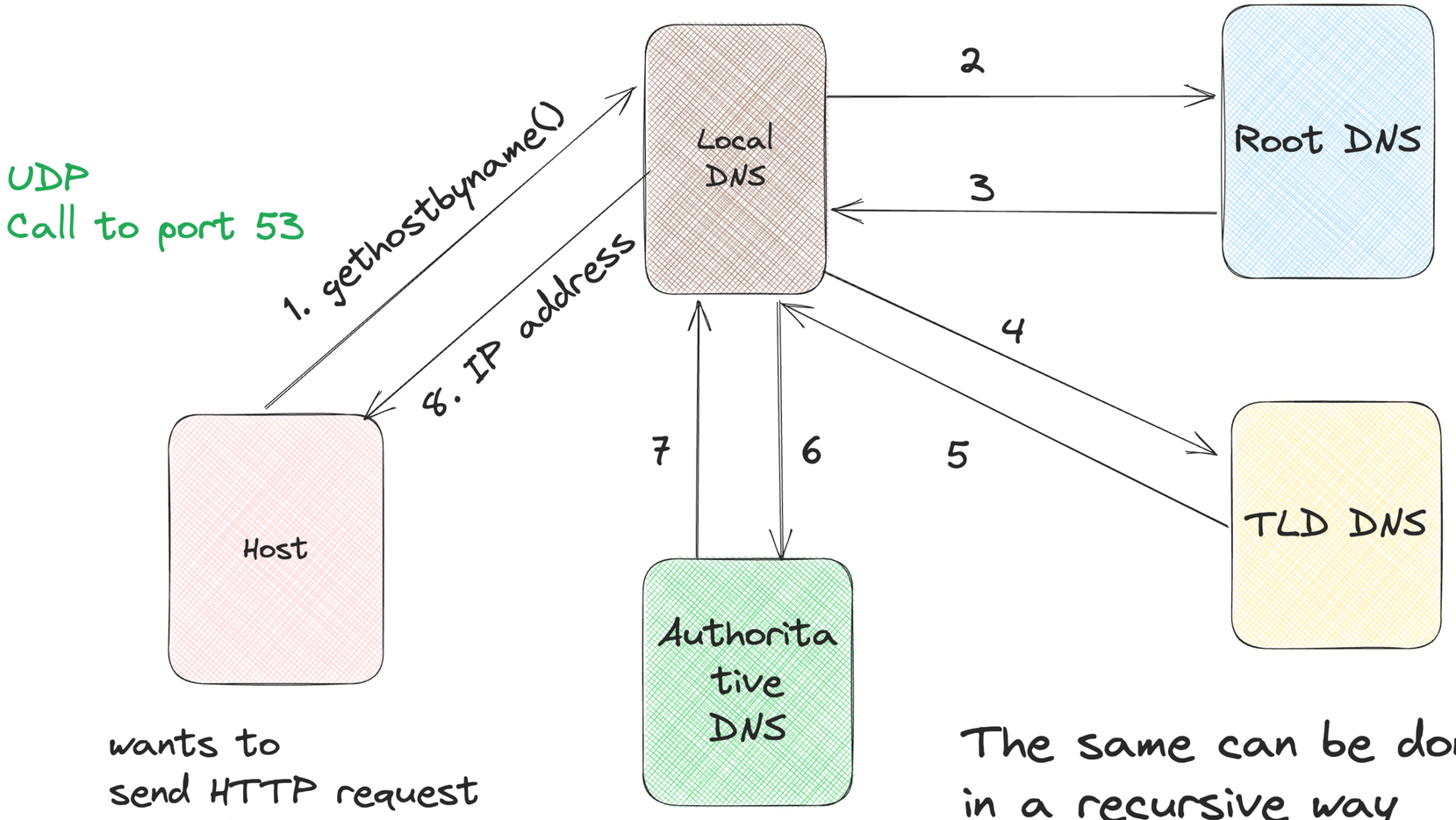
More than 1000 root servers
Copies of 13 different root servers
- 12 different orgs
Coordinated by IANA
return TLD IP address



TLD Servers - can be maintained by orgs, provide IP of authoritative DNS Servers
Authoritative DNS servers - Orgs can choose to implement their own or go for third party
All DNS records have to be made public - that maps hosts to IP address

Local DNS

Each ISP can have DNS and clients can connect to that



Caching can be done at the ISP Local DNS

DNS Records

DNS Servers stores Resource Records (RR)

- Each RR is a tuple: **(name, value, type, ttl)**
- **type = A**
  - name is hostname
  - value is IP address
  - (abc.com, 122.x.x.x, A, 3600)
- **type = NS**
  - name is domain
  - value is hostname of authoritative
  - (abc.com, ns.host.com, NS, 86400)
- **type = CNAME**
  - name is alias of canonical name
  - value is canonical name
  - (abc.com, west-abc.com, CNAME, 86400)
- **type = MX**
  - name is domain
  - value is name of SMTP mail server
  - (abc.com, mail.abc.com, MX, 3600)

More questions needs to be answered

How does Network layer route the traffic?

Before that: Its not that only one process will be running at one time?

Thank you

Course site: karthikv1392.github.io/cs3301_osn
Email: karthik.vaidhyanathan@iiit.ac.in
Twitter: @karthi_ishere