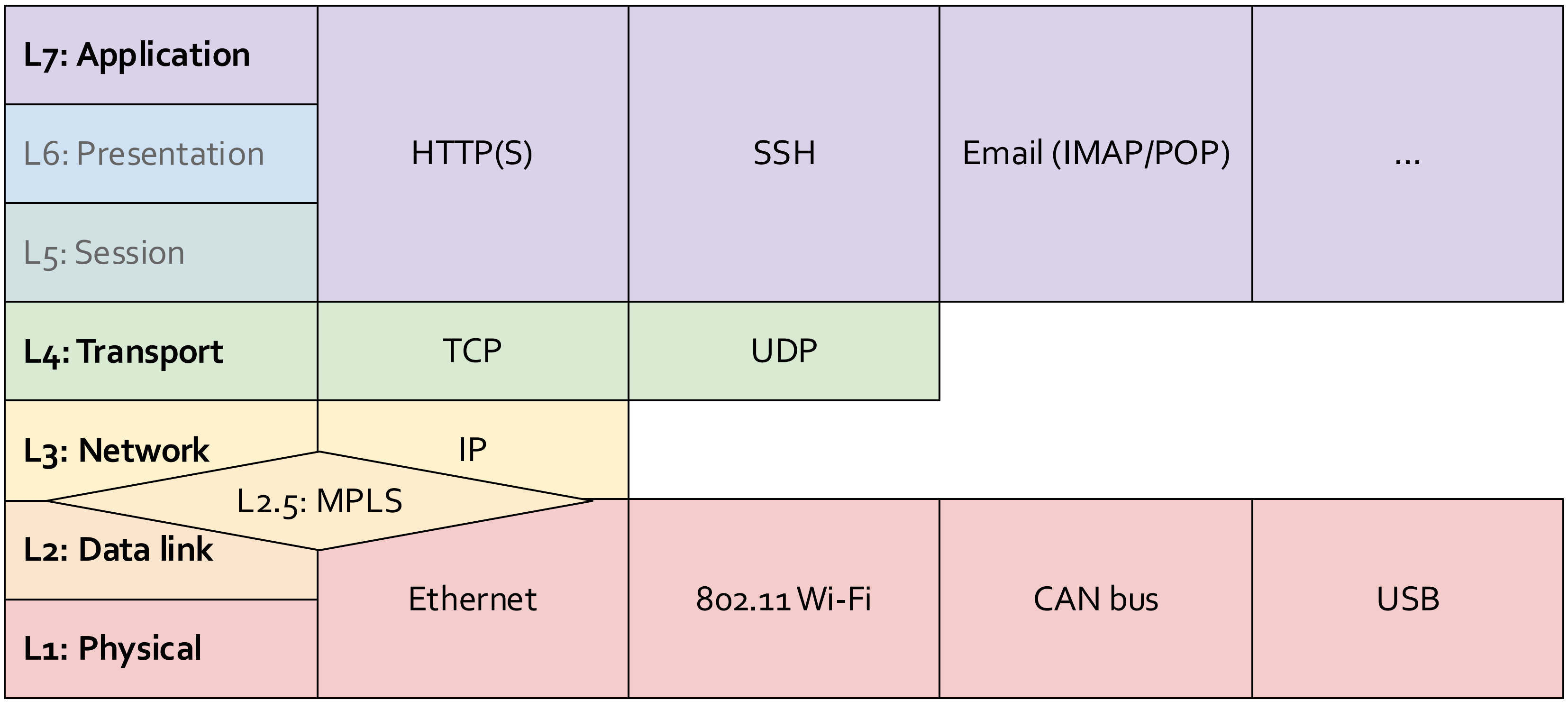
Web caches/ cookies and the DNS

CE 352, Computer Networks

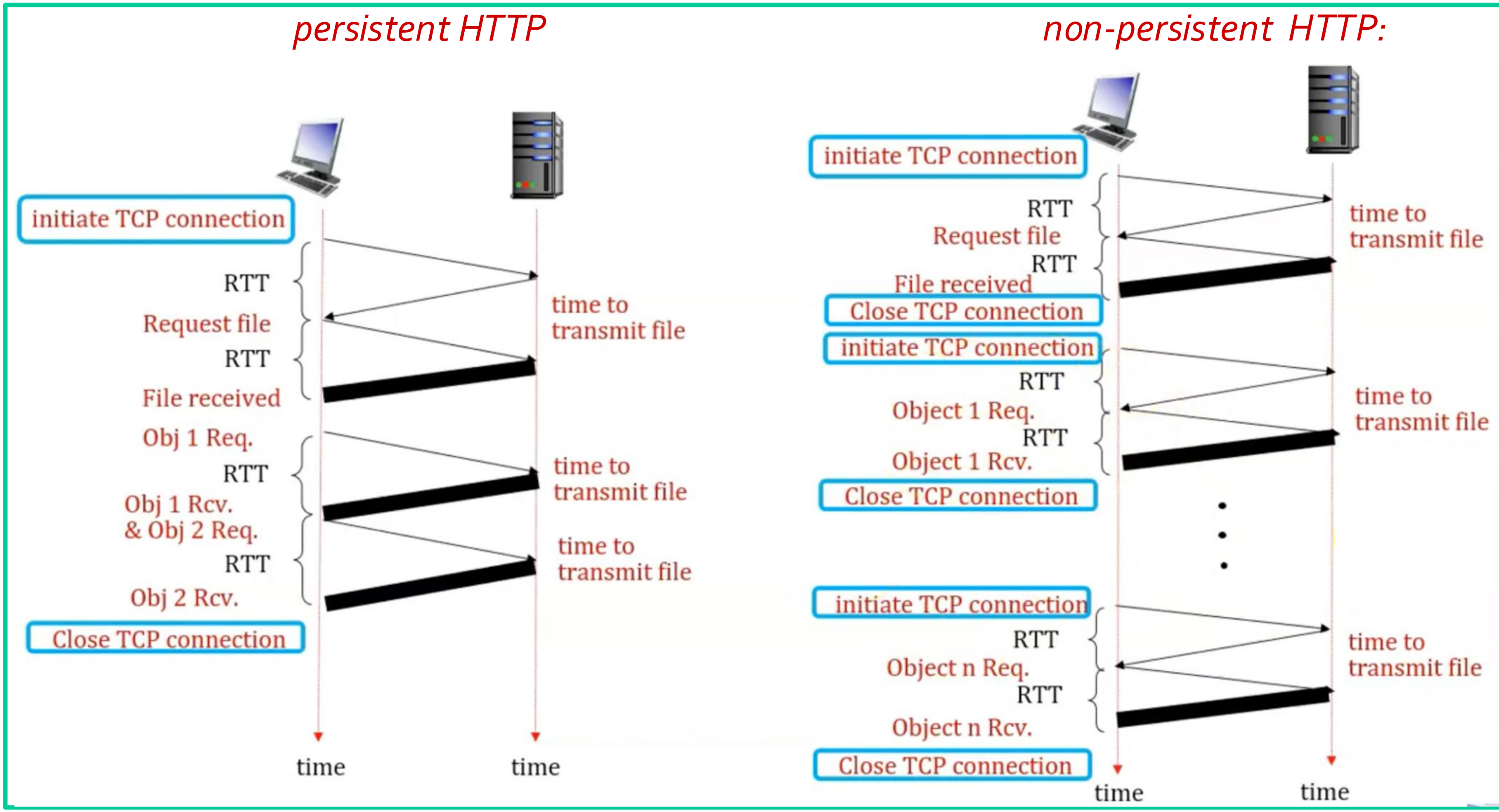
Salem Al-Agtash

Lecture 5

Recap (Internet Protocol Stack)



Recap (Persistent vs no-persistent http)



Topics of today

- Part1

  1- Stateless and stateful web servers

  2- Cookies to maintain state in stateless protocol

  3- Uses of cookies

  4- Web caches and caching examples

- Part2

  1- What is a DNS and its hierarchy

  2- The DNS 13 root name servers

  3- The TLD authoritative servers

  4- The local DNS name server

  5- The DNS name resolution example

  6- DNS records: host and dig commands

  7- DNS caching and updating records

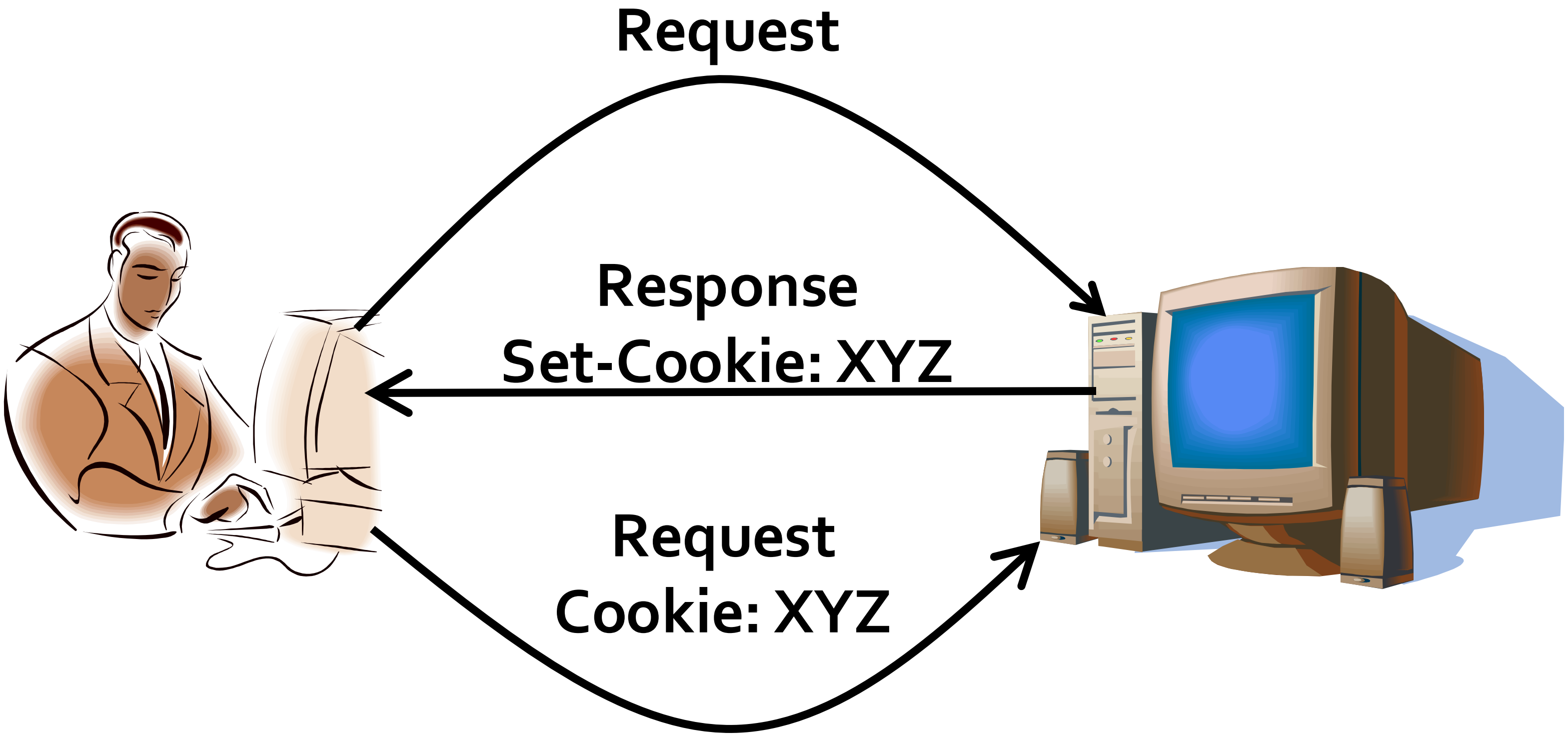
  8- DNS properties and attacks

Cookies: State in a Stateless Protocol

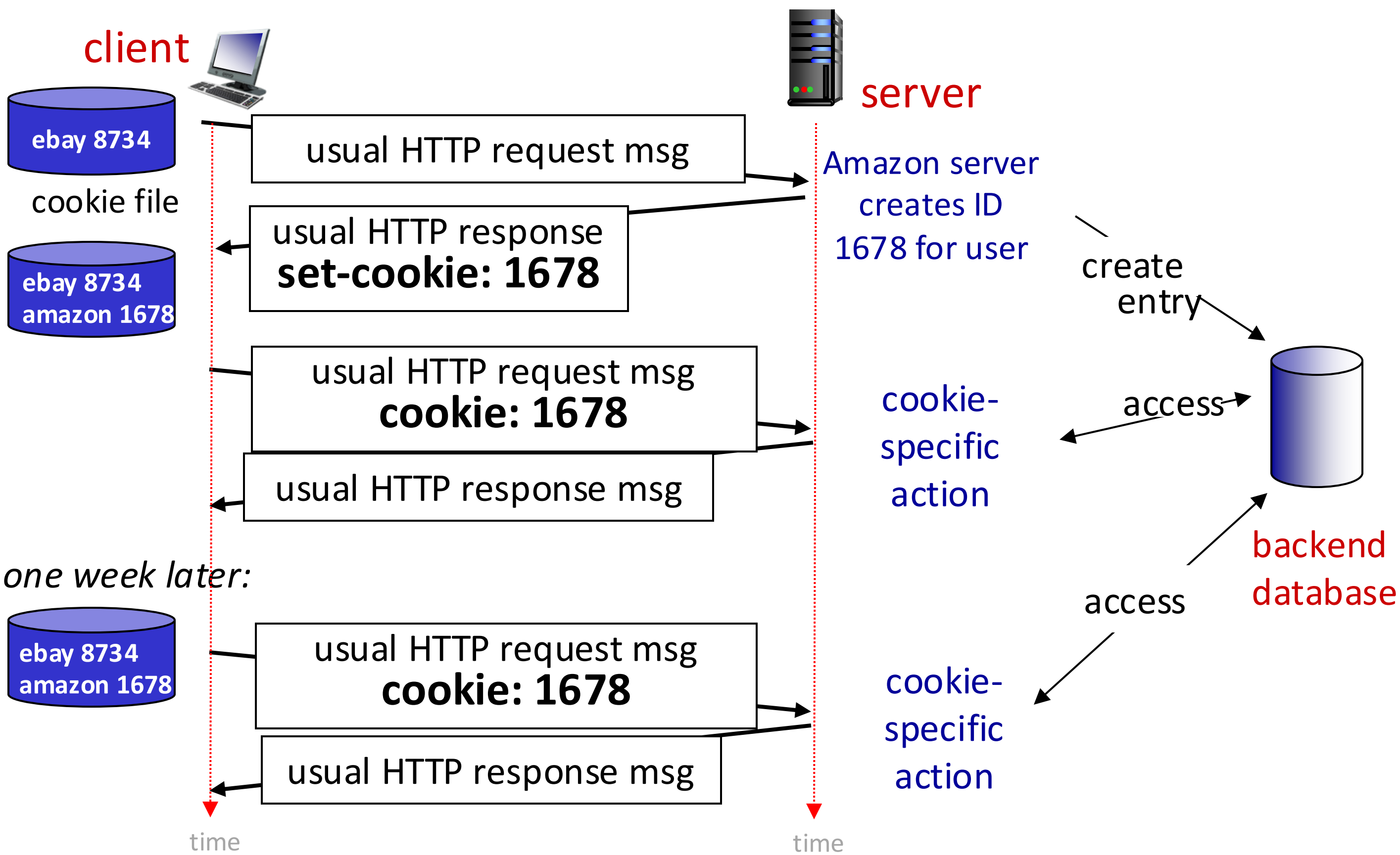
Client-side state maintenance

- Client stores small state on behalf of server
- Client sends state in future requests to the server

Can provide authentication



Cookies: keeping "state"



Web sites and client browser use *cookies* to maintain some state between transactions

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

Use of cookies

what cookies can be used for:

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

aside

cookies and privacy:

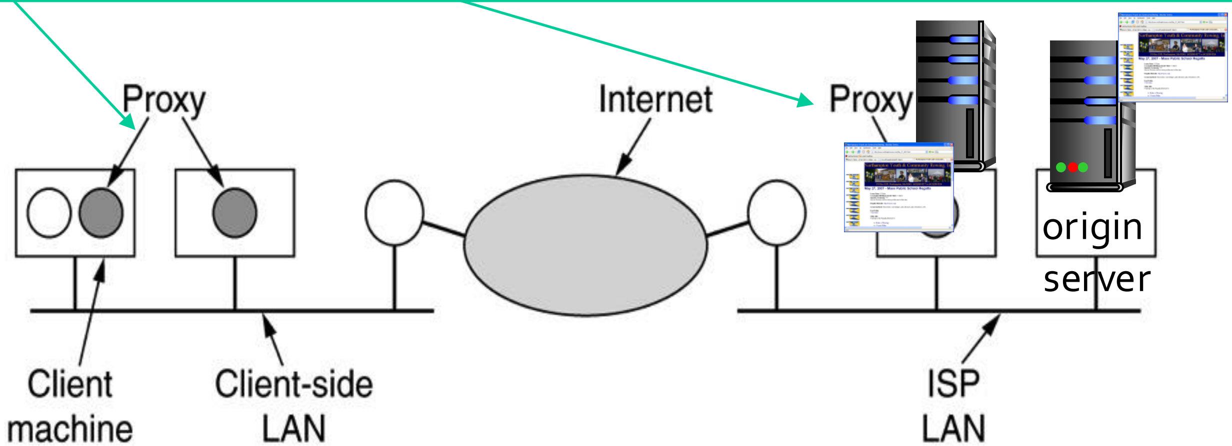
- cookies permit sites to learn a lot about you
- third party persistent cookies (tracking cookies) allow common identity (cookie value) to be tracked across multiple web sites to sites

Web caches (proxy server)

- Responding to client requests without involving origin server
- Web users
  - Availability and fast downloads
- Web content providers
  - More users, cost-effective infrastructure, and non-congested network

Why Web caching?

- reduce response time for client request
  - cache is closer to client
- reduce traffic on an institution's access link
- Internet is dense with caches
  - enables "poor" content providers to more effectively deliver content



- Proxies: Caching and replication
- Content delivery networks (CDN): economies of scale

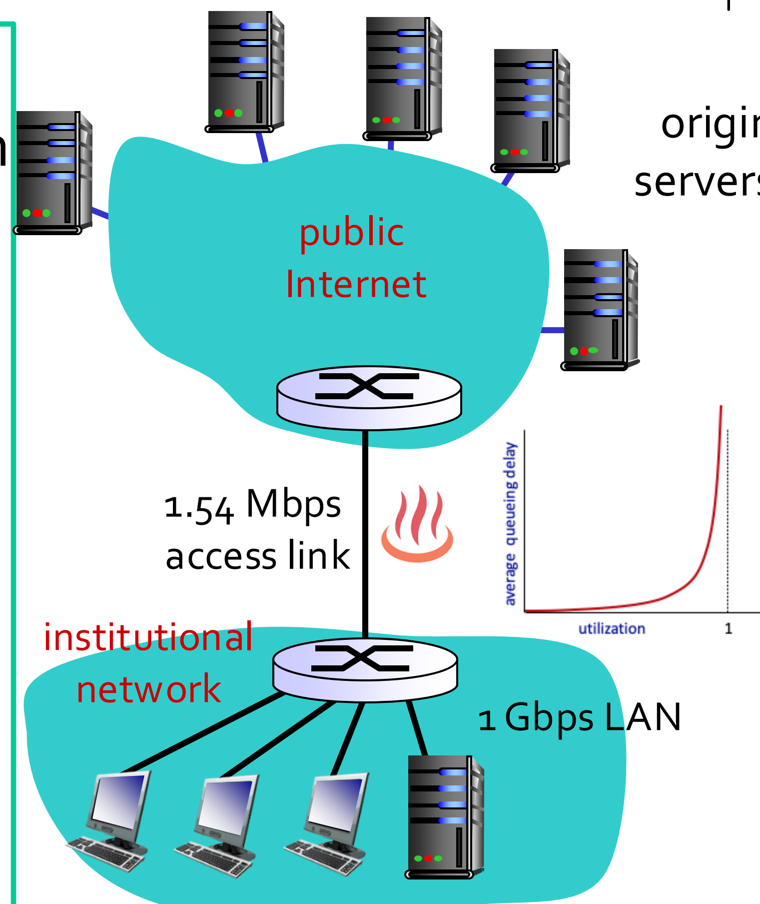
Caching example:

assumptions:

- avg request rate from browsers to origin servers:1.5Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 1.54 Mbps

consequences:

- LAN utilization:?
- access link utilization:?
- total delay =



Caching example:

assumptions:

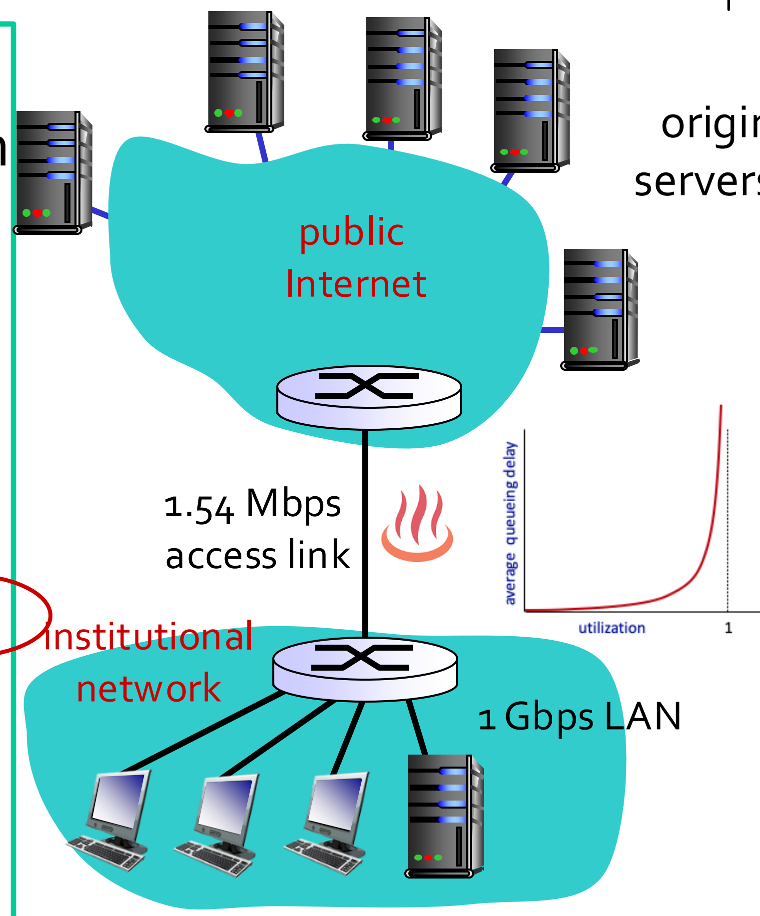
- avg request rate from browsers to origin servers:1.5Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 1.54 Mbps

consequences:

- LAN utilization: (1.5Mbps)/1 Gbps = 0.15%
- access link utilization: 1.5 Mbps/1.54Mbps = 97%
- total delay = Internet delay + access delay + LAN delay

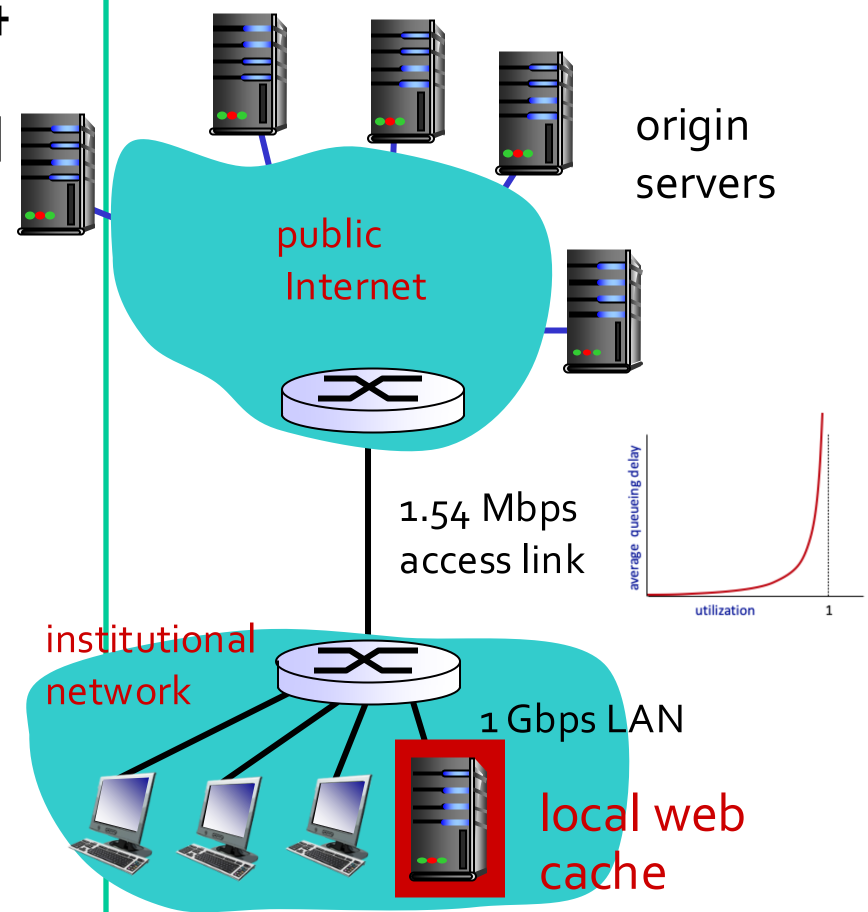
Assuming local access 0.01 sec, then: total delay = 2 + 0.1 = 2.01 sec
As access link approaches 100% utilization, link upgrade in needed for users to get better internet response!

This delay becomes very large when the link approached 100% utilization, unacceptable for institution's users. So: Either upgrade the link or install a local cache



Install local cache

- suppose cache hit rate is 0.4
  - 40% requests satisfied at cache, 60% requests satisfied at origin
- access link utilization:
  - 60% of requests use access link
- data rate to browsers over access link =?
- utilization =?
- total delay = ?



Install local cache

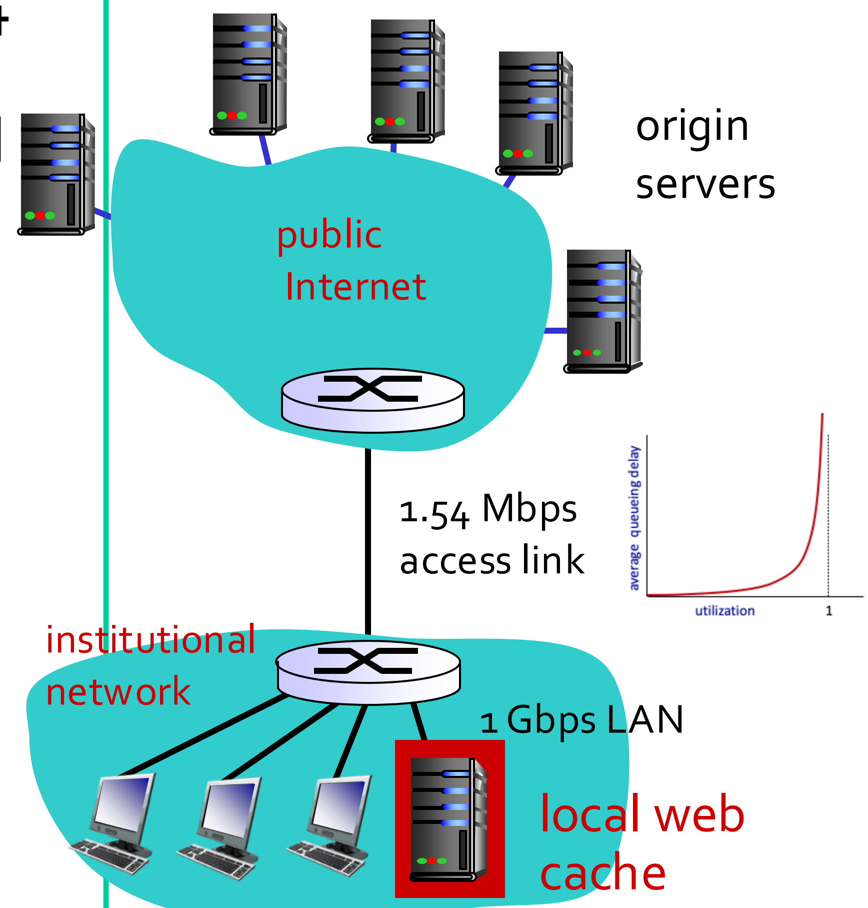
- suppose cache hit rate is 0.4
  - 40% requests satisfied at cache, 60% requests satisfied at origin
- access link utilization:
  - 60% of requests use access link
- data rate to browsers over access link

  = 0.6*1.50 Mbps = .9 Mbps
- utilization = 0.9/1.54 = 58%

- total delay

  = 0.6 * (delay from origin servers) +0.4 * (delay when satisfied at cache)

Assuming local access 0.01 sec, then: total delay = 0.6 (2.01) + 0.4 (0.1) = 1.21 sec
So with no link upgrade, users still get better internet response!



The DNS: domain name system

Internet hosts, routers:

- IP address (32 or 128 bits) - used for addressing datagrams
- “name”, e.g., www.yahoo.com - used by humans

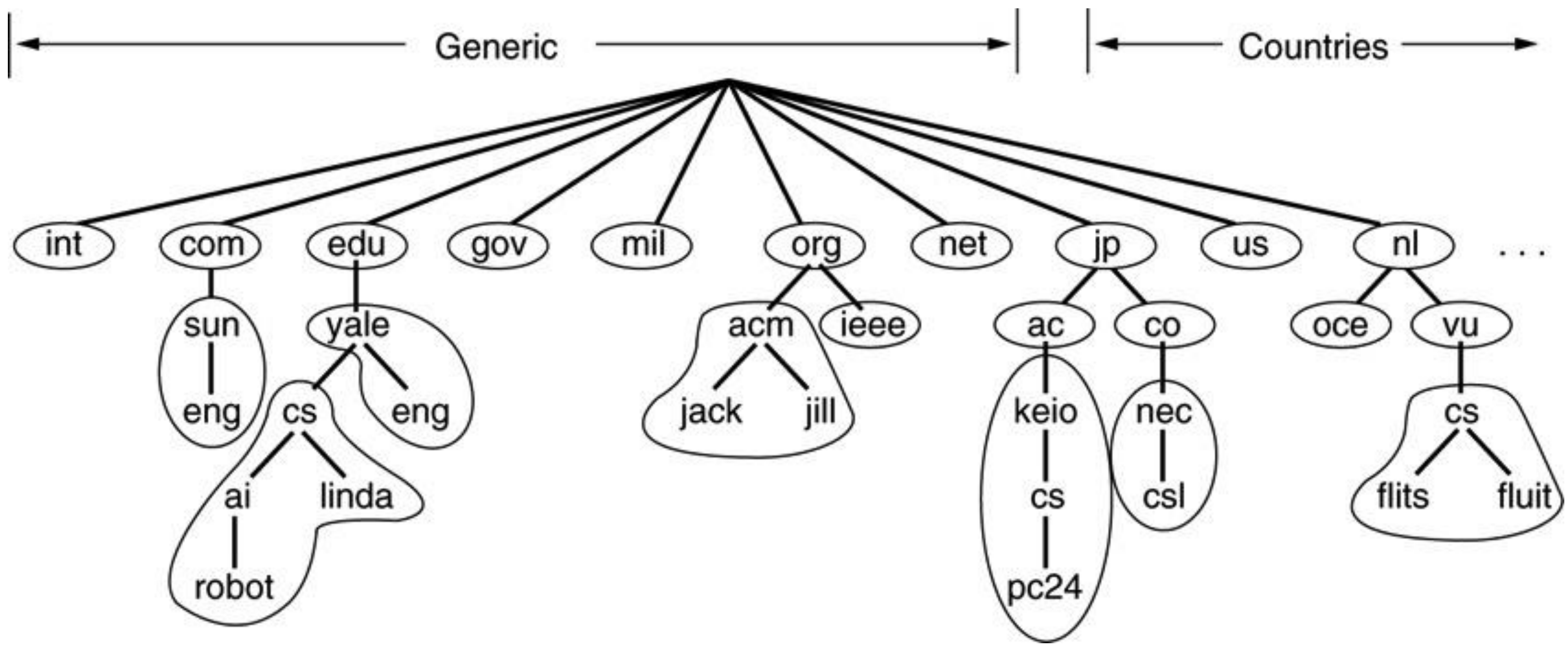
Domain Name System:

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol:* hosts, name servers communicate to *resolve* names (address/name translation)

handles many *trillions* of queries/day:

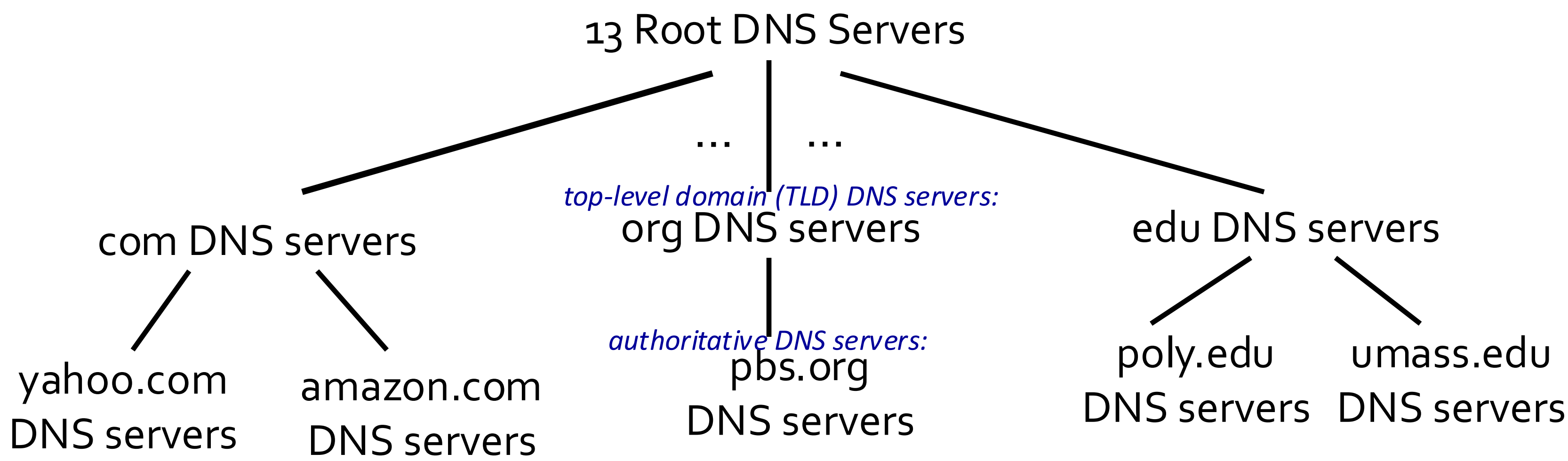
- *many* more reads than writes
- *performance matters:* almost every Internet transaction interacts with DNS - msecs count!

DNS hierarchy



- DNS name space is divided into zones.
- Each zone contains some part of the tree and contains name servers holding information about the that zone

DNS: a distributed, hierarchical database



client wants IP for www.amazon.com; 1st approximation:

- client queries amazon.com DNS server to get IP address for www.amazon.com
- client queries .com DNS server to get amazon.com DNS server
- client queries root server to find .com DNS server

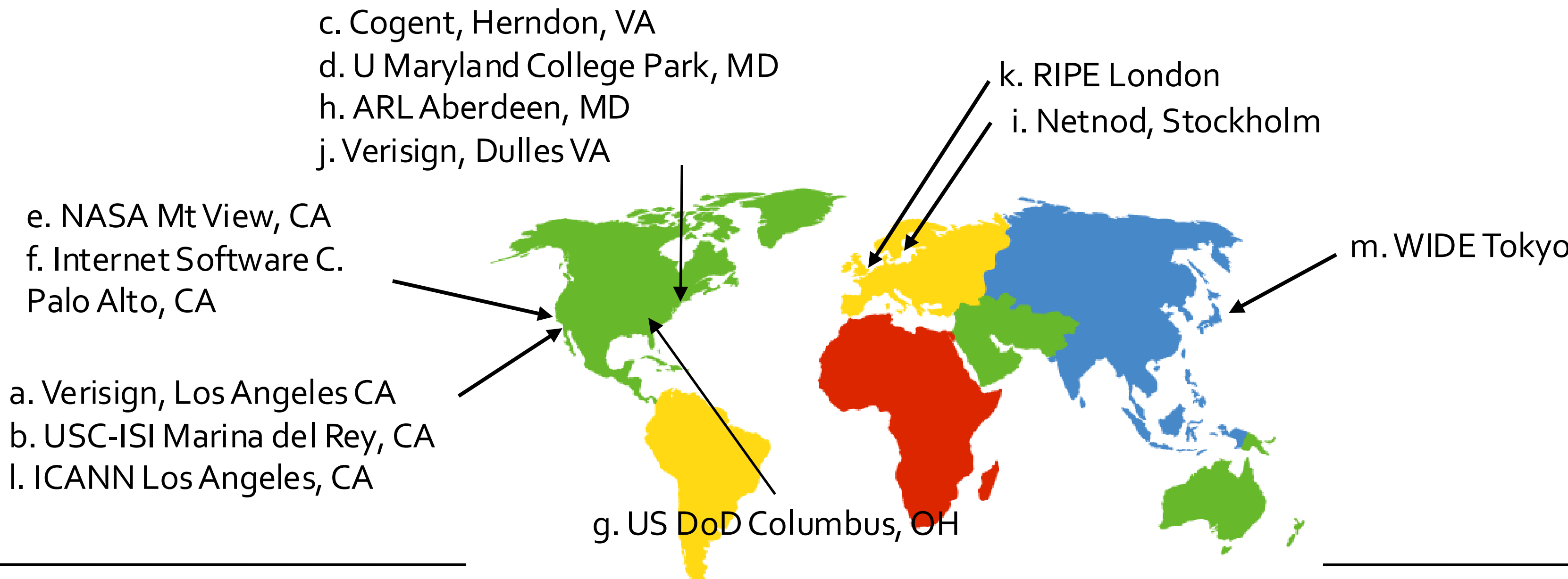
DNS: root name servers (1)

13 root servers (labeled A – M) http://www.root-servers.org/

contacted by local name server that can not resolve name

root name server:

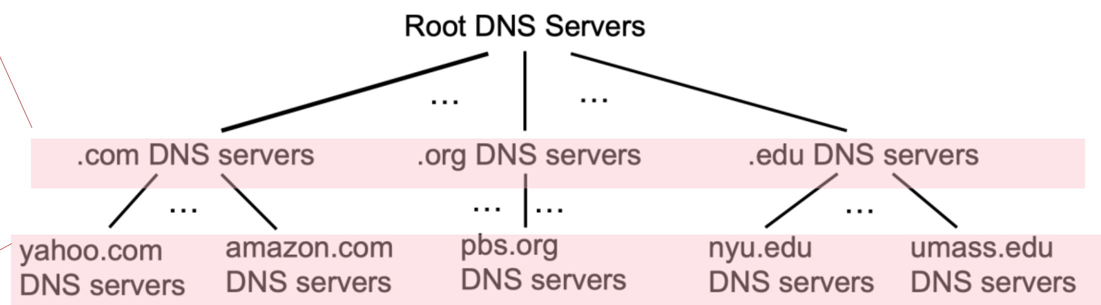
- contacts authoritative name server if name mapping not known
- gets mapping
- returns mapping to local name server



TLD (2), authoritative (3) servers

top-level domain (TLD) servers:

- responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
- Network Solutions maintains servers for .com TLD
- Educause for .edu TLD

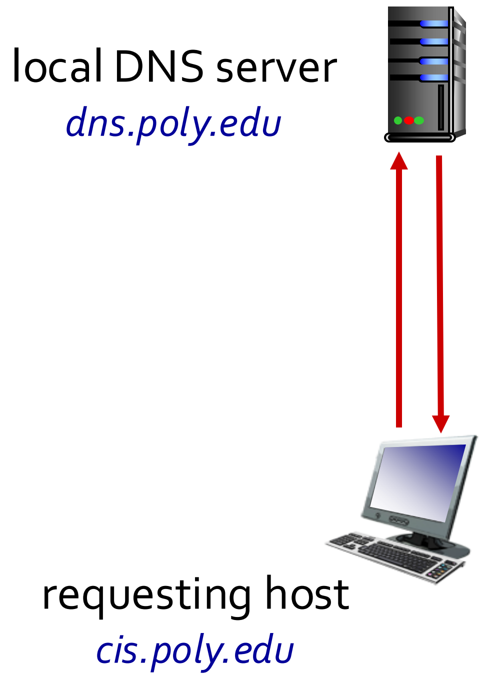


authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

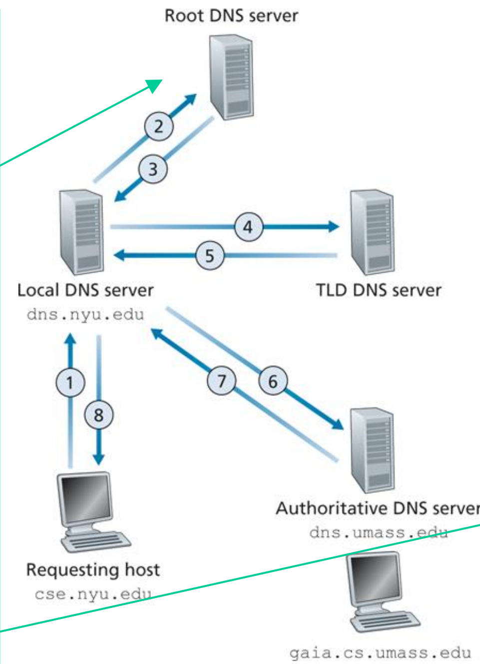
Local DNS name (4) server

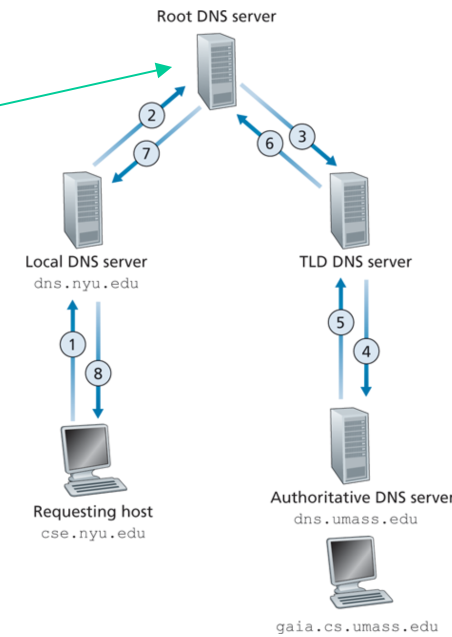
- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one also called "default name server"
- when host makes DNS query, query is sent to its local DNS server
- has local cache of recent name-to-address translation pairs (but may be out of date!)
- acts as proxy, forwards query into hierarchy



DNS name resolution example

- Client--server interaction on UDP Port 53
- host at cse.nyu.edu wants IP address for gaia.cs.umass.edu
- *iterative query:*
  - contacted server replies with name of server to contact
  - "I don't know this name, but ask this server"
  - Widely used in practice
- *recursive query:*
  - puts burden of name resolution on contacted name server
  - heavy load at upper levels of hierarchy?





DNS records

DNS: distributed database storing resource records (RR)

RR format: **(name, value, type, ttl)**

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

host – a www.gju.edu.jo

```
$host -a www.gju.edu.jo
Trying "www.gju.edu.jo"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32587
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 1


;; QUESTION SECTION:
;www.gju.edu.jo. IN ANY


;; ANSWER SECTION:
www.gju.edu.jo. 1140 IN A 87.236.233.242

;; AUTHORITY SECTION:
gju.edu.jo. 1543 IN NS dns1.junet.edu.jo.
gju.edu.jo. 1543 IN NS mail-relay.gju.edu.jo.
gju.edu.jo. 1543 IN NS ns1.gju.edu.jo.


;; ADDITIONAL SECTION:
ns1.gju.edu.jo. 1753 IN A 87.236.233.230

Received 132 bytes from fe80::1%5#53 in 7 ms
```

Try:

- dig www.gju.edu.jo
- dig www.gju.edu.jo any
- dig www.gju.edu.jo MX
- dig www.gju.edu.jo SOA

Inserting records into DNS

- Example: new startup "Network Utopia"
- Get a block of space from ISP
    - e.g. 220.44.8.128/25
- Register networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
    - provide registrar with names and IP addresses of authoritative name server (primary and secondary)
    - registrar inserts two RRs into .com TLD server:
        - **(networkutopia.com, dns1.networkutopia.com, NS)**
        - **(dns1.networkutopia.com, 220.44.8.129, A)**

- create authoritative server dns1.networkutopia.com:
    - type A record for www.networkuptopia.com;
    - type MX record for networkutopia.com

DNS: caching, updating records

- Performing DNS queries takes time
- Caching can reduce overhead
  - once (any) name server learns mapping, it *caches* mapping
  - cache entries timeout (disappear) after some time "*time to live*" (TTL)
- TLD servers typically cached in local name servers thus root name servers not often visited
  - cached entries may be *out-of-date* (best effort name-to-address translation!)
  - if name host changes IP address, may not be known Internet-wide until all TTLs expire
- update/notify mechanisms proposed IETF standard RFC 2136

DNS properties and attacks

- Important properties of DNS
  - Easy unique naming and reasonable trust model
  - Caching lends scalability and performance
- “Distributed Denial of Service” DDoS attacks
  - bombard root servers with traffic
    - local DNS servers cache IPs of TLD servers, allowing root server bypass
  - bombard TLD servers, potentially more dangerous
- redirect attacks
  - man-in-middle, Intercept queries
  - DNS poisoning
    - Send fake records to DNS server, which caches
- exploit DNS for DDoS
  - send queries with spoofed source address: target IP
  - requires amplification

Summary

Today:

- Web caches and cookies
- Caching example
- The DNS
- DNS records
- DNS properties and attacks

Canvas discussion:

- Reflection
- Exit ticket

Next time:

- Read 2.3 and 2.5 of K&R
- follow on Canvas! Material and announcements

Any questions?