CS 43: Computer Networks

Naming and DNS

September 24, 2019

SWARTHMORE COLLEGE

Last class

- DNS: Domain Name System
  - Core Internet Functionality
  - Application Layer Protocol – E2E design!
  - Client-Server Architecture
  - Hierarchical, Distributed

Today

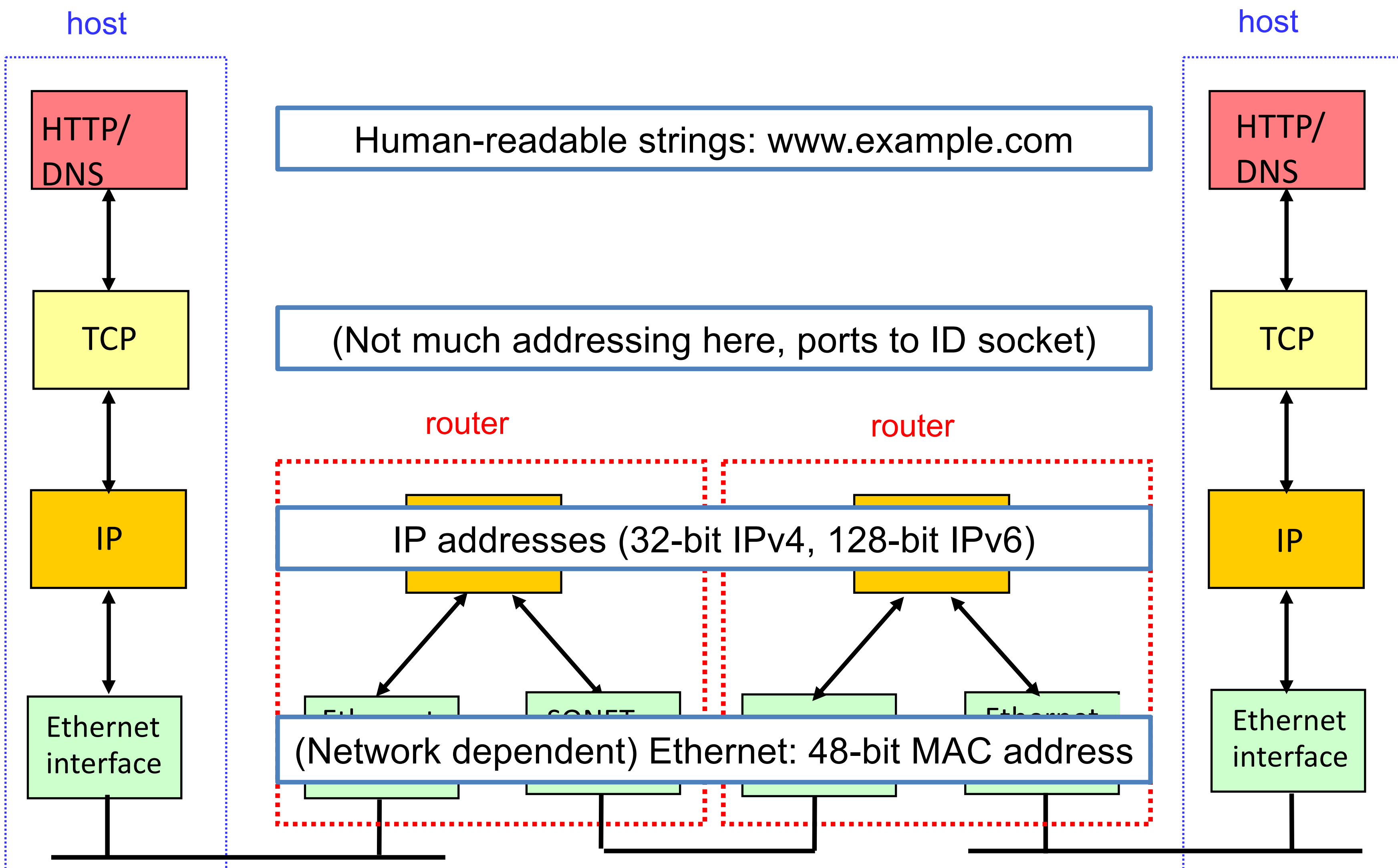
- Domain Name System
  - Query sequences: Record types
  - Caching: Load Balancing
  - Security

DNS: domain name system

Input: www.google.com (hostname)

Output: 8.8.8.8 (IP address)

Recall: TCP/IP Protocol Stack



DNS protocol, messages

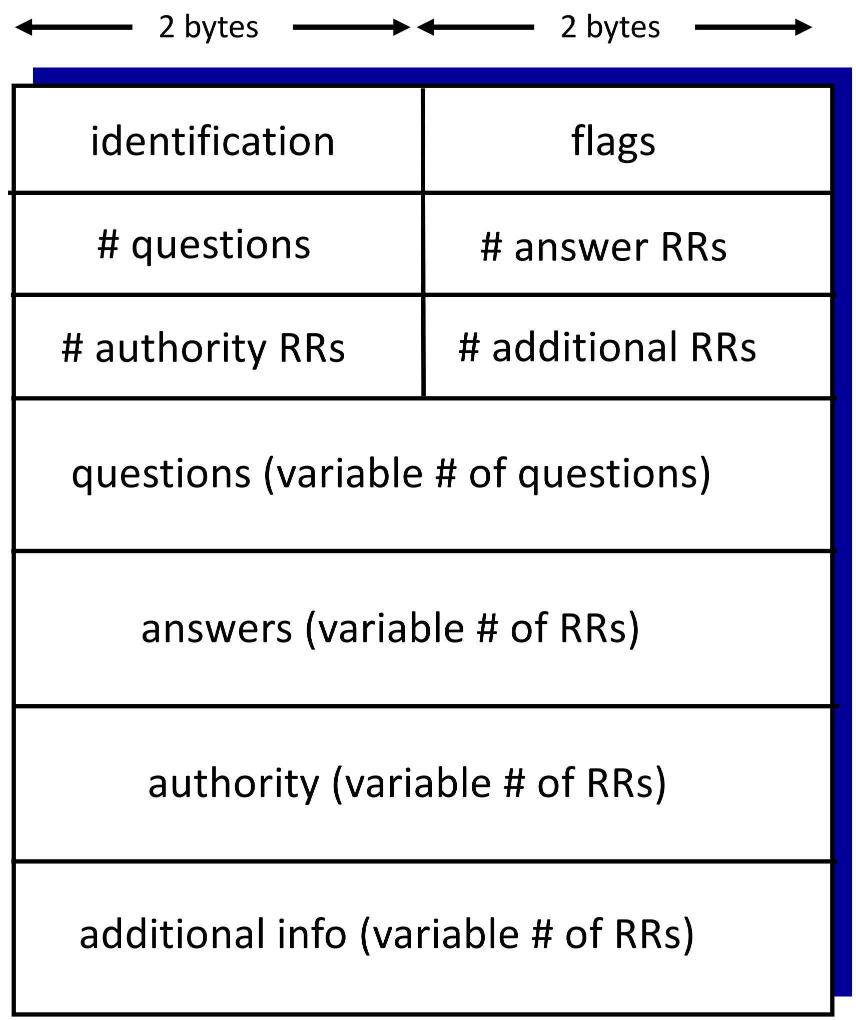
- <u>query and reply messages, both with same message format!</u>

Binary Protocol!

- Delimiters: pre-defined lengths/field
- Names: <len><name>

Sent via UDP (User Datagram Protocol)

- No connection established
- Not reliable

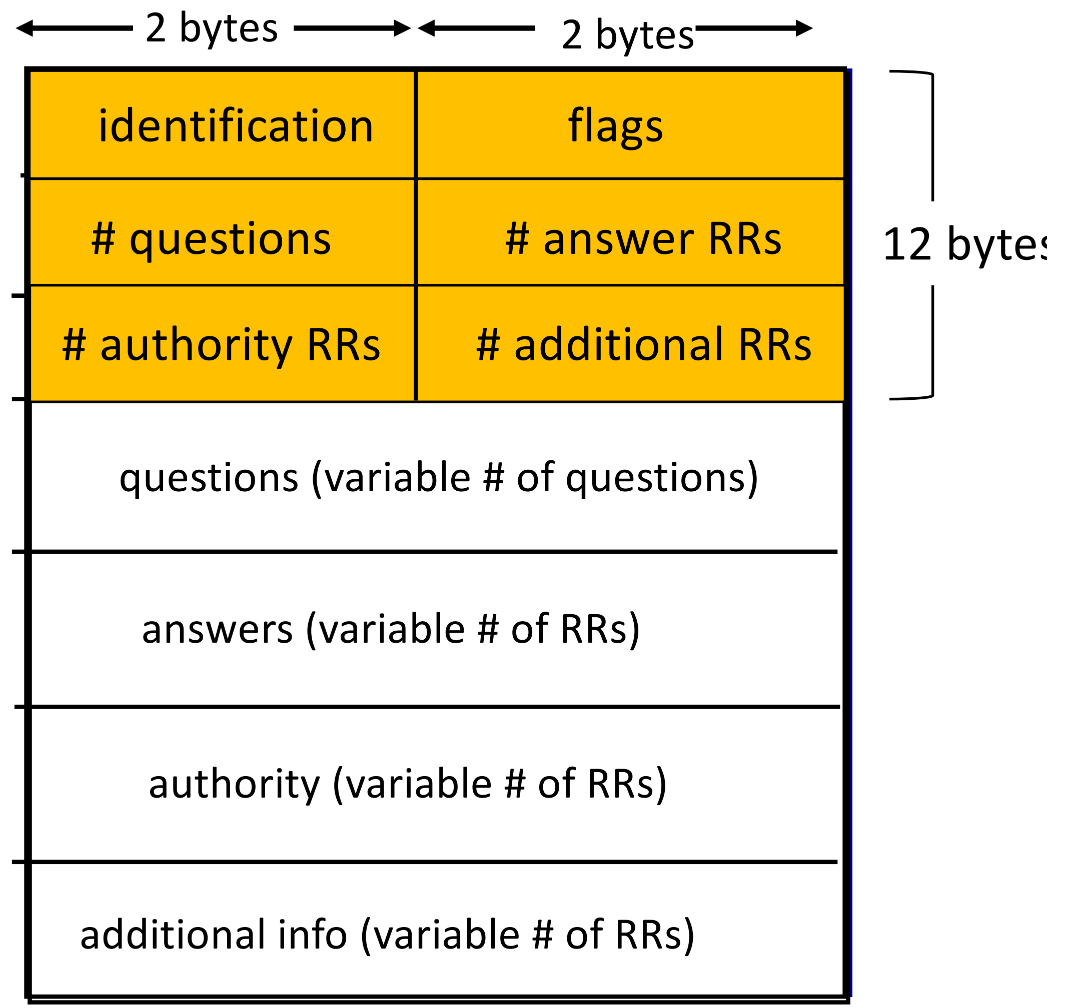


DNS protocol, messages

- query and reply messages, both with same message format

Message header

- identification: 16 bit id for query, reply to query uses same id.
- flags: recursion, query/reply
- # Resource Records to follow



DNS Services

DNS: distributed DB storing resource records (RR)

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

type=A

- **name** is hostname
- **value** is IP address

type=NS

- **name** is domain (e.g., foo.com)
- **value** is hostname of authoritative name server for this domain

type=CNAME

- **name** is alias name for some "canonical" (the real) name
- **www.ibm.com** is really
- servereast.backup2.ibm.com
- **value** is canonical name

type=MX

- **value** is name of mailserver associated with name

DNS Types

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

- Type = A / AAAA
  - Name = domain name
  - Value = IP address
  - A is IPv4, AAAA is IPv6

Query
Name: cs.swarthmore.edu
Type: A

Resp.
Name: cs.swarthmore.edu
Value: 130.58.68.9

- Type = NS
  - Name = partial domain
  - Value = name of DNS server for this domain
  - "Go send your query to this other server"

Query
Name: cs.swarthmore.edu
Type: NS

Resp.
Name: cs.swarthmore.edu
Value: 130.58.68.9

DNS Types, Continued

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

- Type = CNAME
  - Name = hostname
  - Value = canonical hostname
  - Useful for aliasing
  - CDNs use this

Query
Name: foo.mysite.com
Type: CNAME

Resp.
Name: foo.mysite.com
Value: bar.mysite.com

- Type = MX
  - Name = domain in email address
  - Value = canonical name of mail server

Query
Name: cs.umass.edu
Type: MX

Resp.
Name: cs.umass.edu
Value: barramail.cs.umass.edu.

DNS Directory Design: which would you choose?

A. Flood the query to all end-hosts, the end-host with the name responds.
B. Centralized server: all data and queries handled by one machine.
C. Push data to all end-hosts (/etc/hosts): each end-host stores the full listing.
D. Something else

Domain Name System (DNS)

- Distributed administrative control
  - Hierarchical name space divided into zones
  - Distributed over a collection of DNS servers

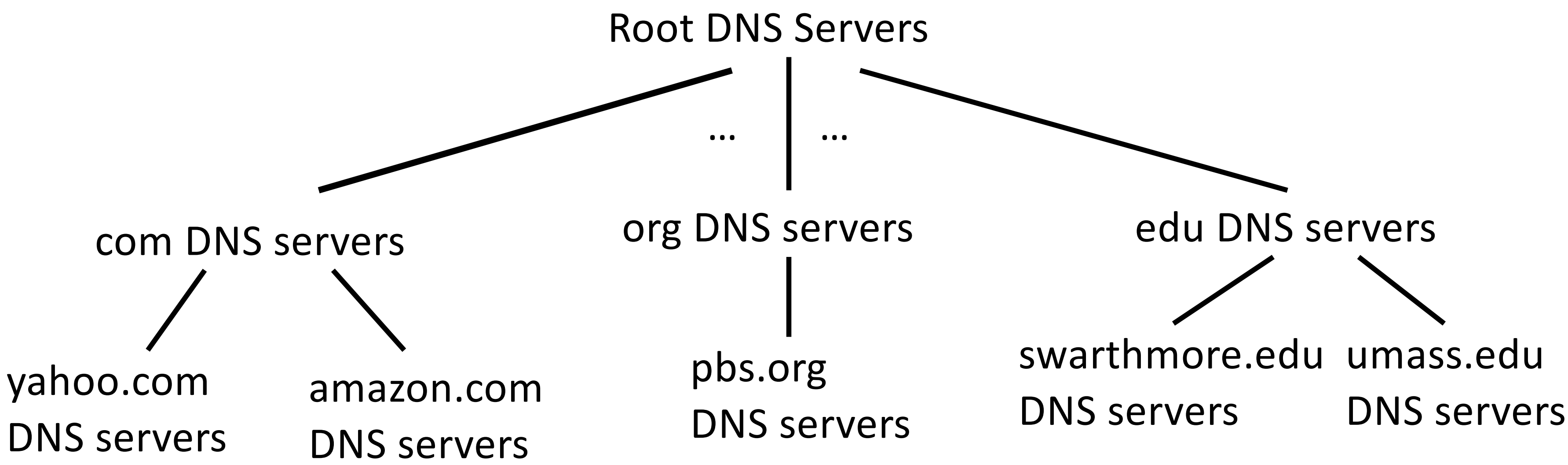
Domain Name System (DNS)

- Hierarchy of DNS servers
  - Root servers
  - Top-level domain (TLD) servers
  - Authoritative DNS servers

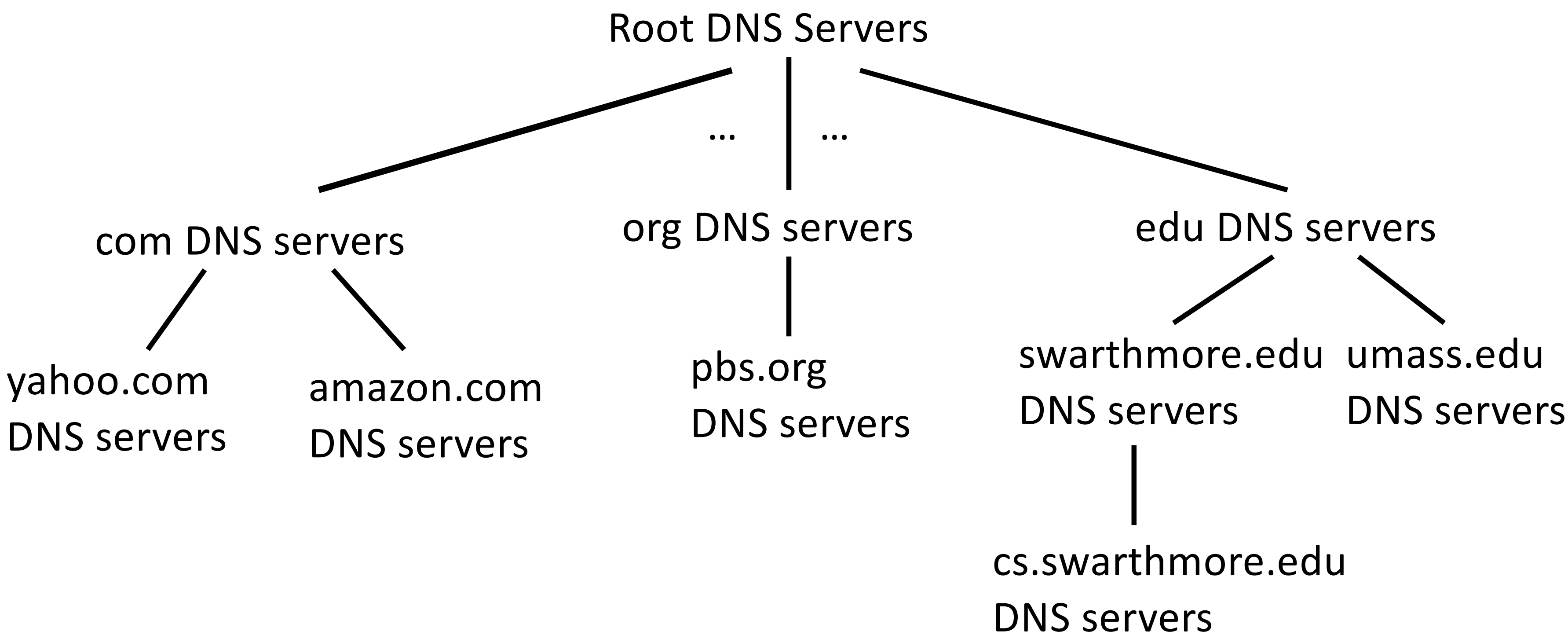
Domain Name System (DNS)

- Performing the translations
  - Local DNS servers
  - Resolver software

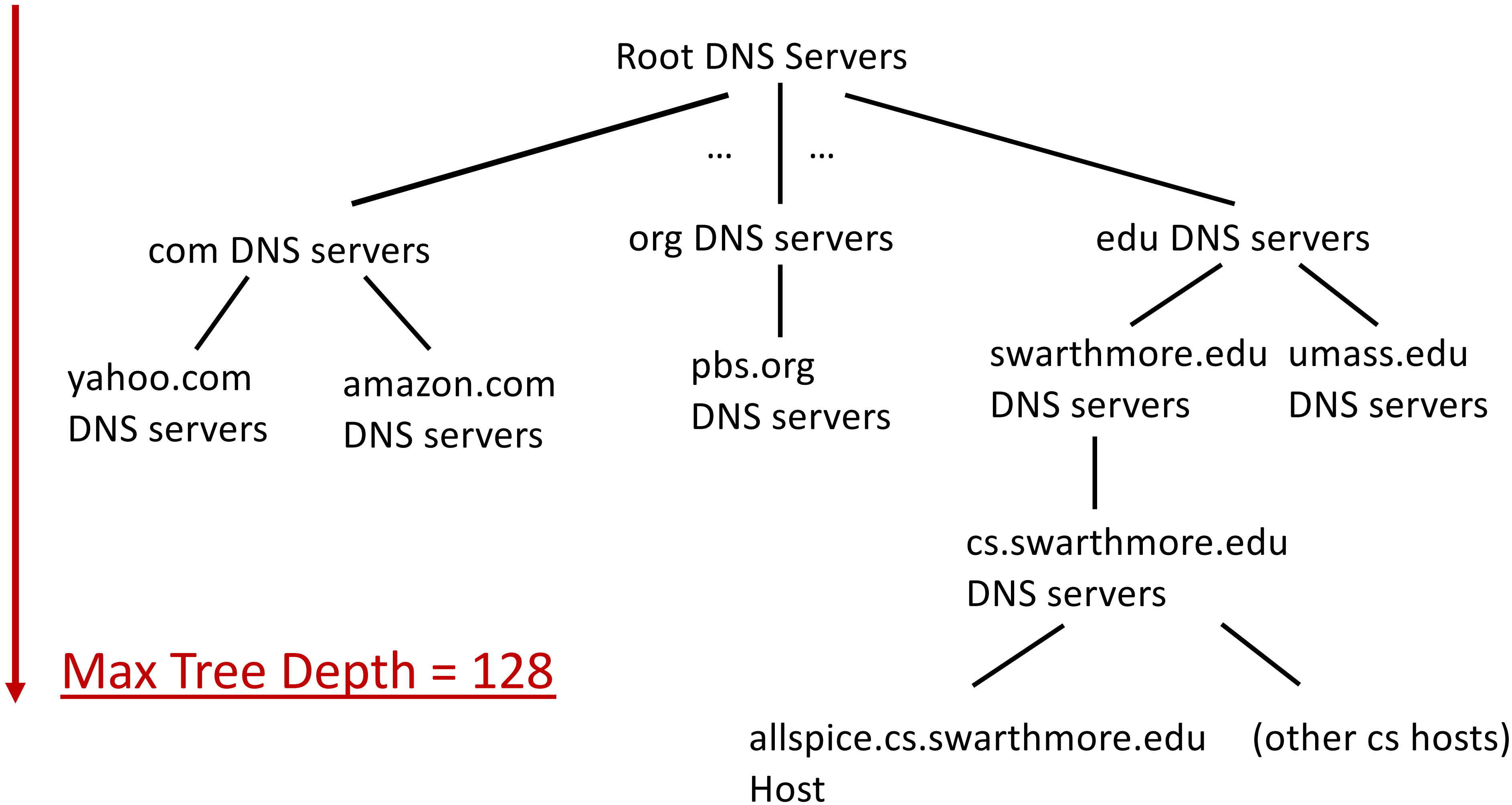
DNS: a distributed, hierarchical database



DNS: a distributed, hierarchical database

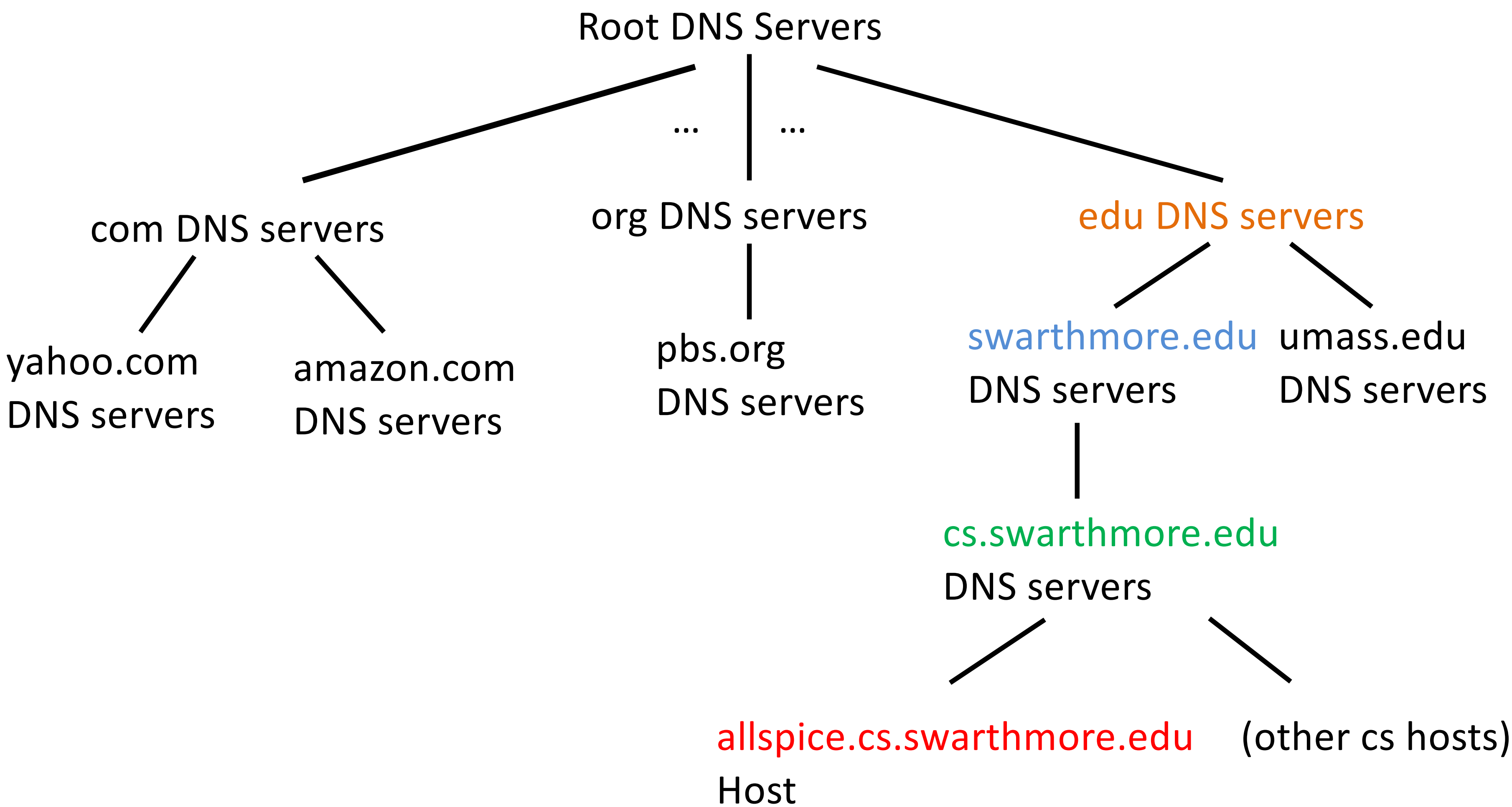


DNS: a distributed, hierarchical database



allspice.cs.swarthmore.edu.

DNS: a distributed, hierarchical database



- allspice.cs.swarthmore.edu.

Nameless root,
Usually implied.

Why do we structure DNS like this? Which of these helps the most? Drawbacks?

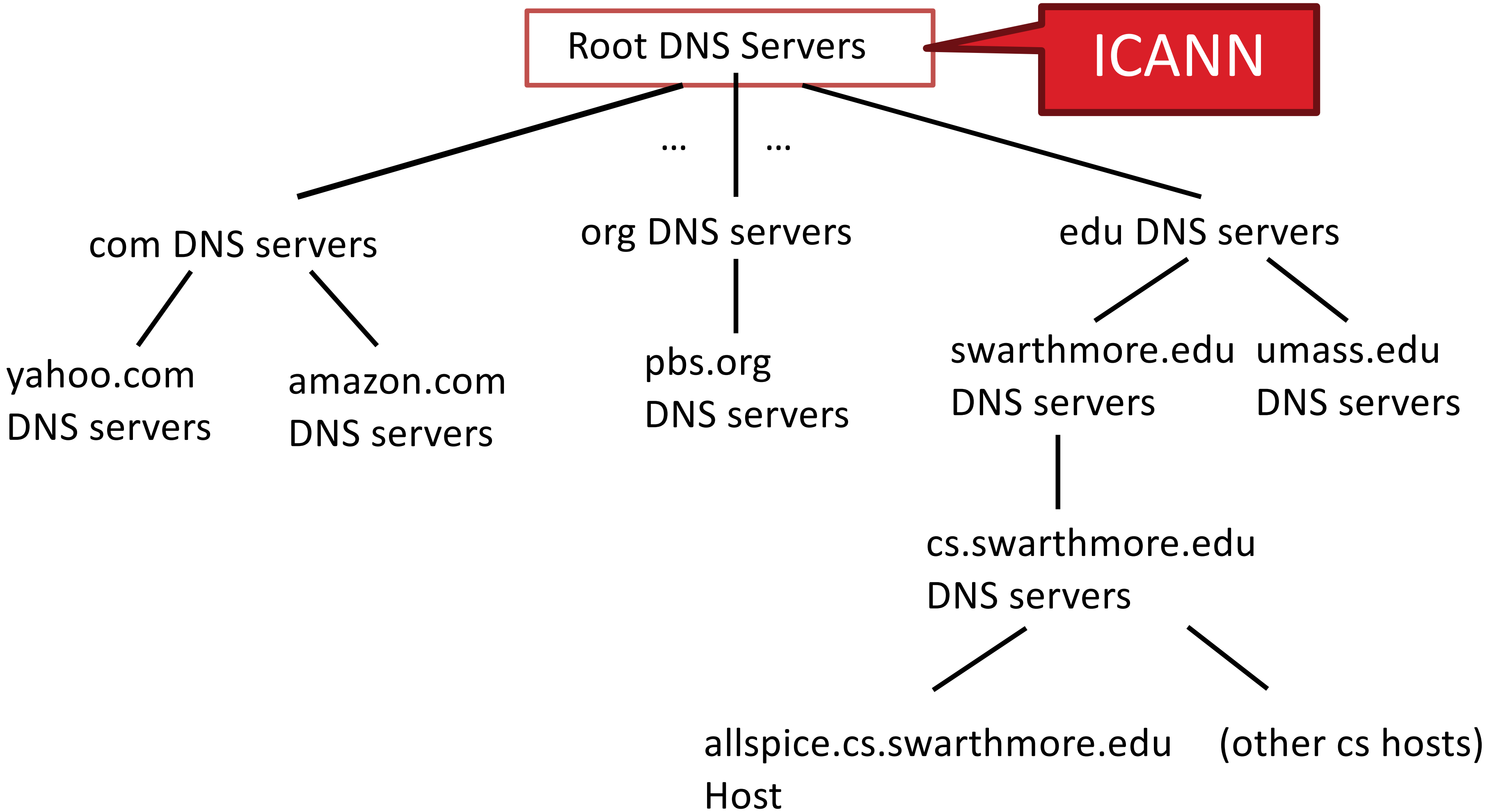
A. It divides up responsibility among parties.

B. It improves performance of the system.

C. It reduces the size of the state that a server needs to store.

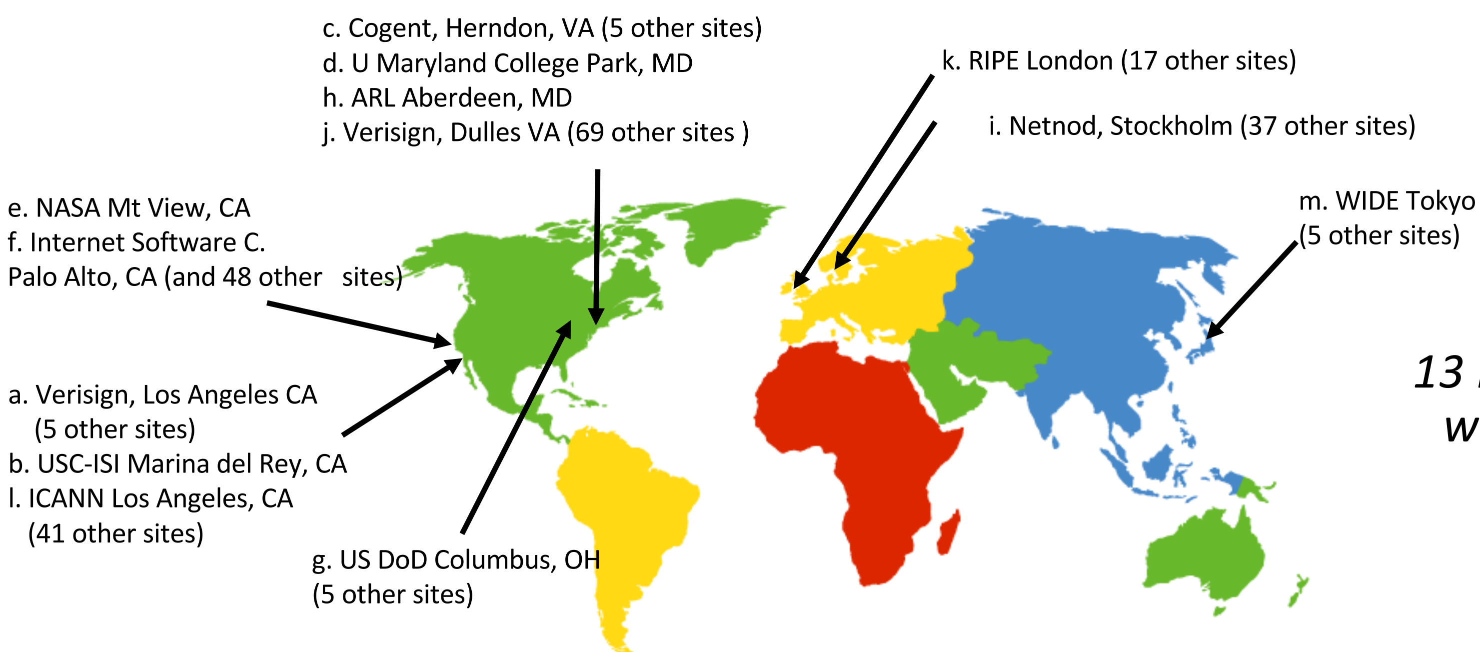
D. Some other reason.

DNS: a distributed, hierarchical database



DNS: Root Name Servers

- **<u>Know how to find top-level domains (.com, .edu, .gov, etc.)</u>**
- approx. 400 total root servers
- Significant amount of traffic is not legitimate



13 root name "servers" worldwide

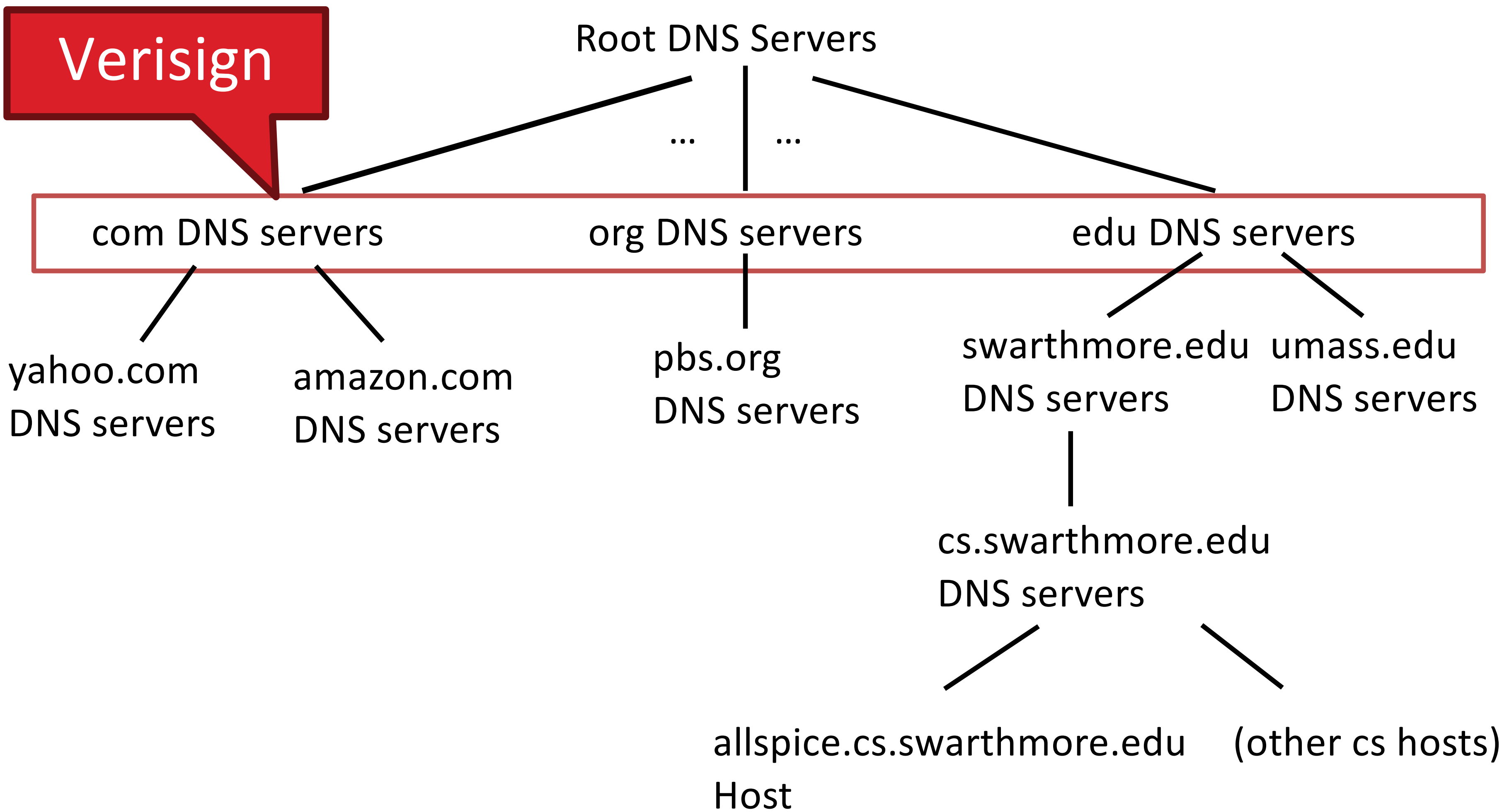
Root Name Servers

- Responsible for the Root Zone File

com.	172800	IN	NS	a.gtld-servers.net.
com.	172800	IN	NS	b.gtld-servers.net.
com.	172800	IN	NS	c.gtld-servers.net.

  - In practice, most systems cache this information
  - Lists the TLDs and who controls them
  - ~272KB in size

Top Level Domain (TLD) servers

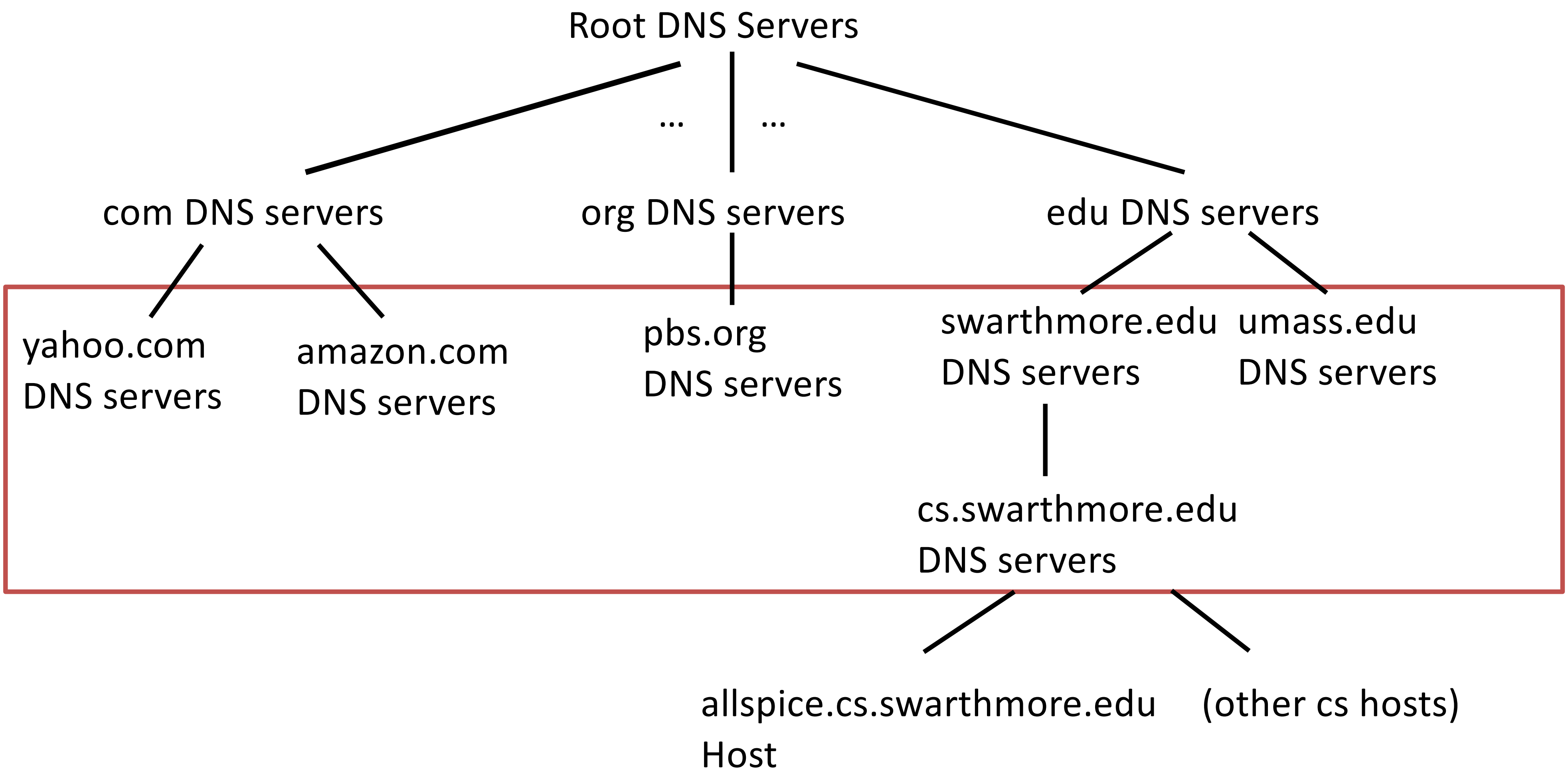


Top Level Domain (TLD) servers

- who maintains the servers?:
  - Verisign: .com, .net
  - Educause: .edu (Verisign backend)
  - local governments or companies

- Responsible for:
  - com, org, net, edu, gov, aero, jobs, museums,
  - all top-level country domains, e.g.: uk, fr, de, ca, jp, etc

Authoritative Servers

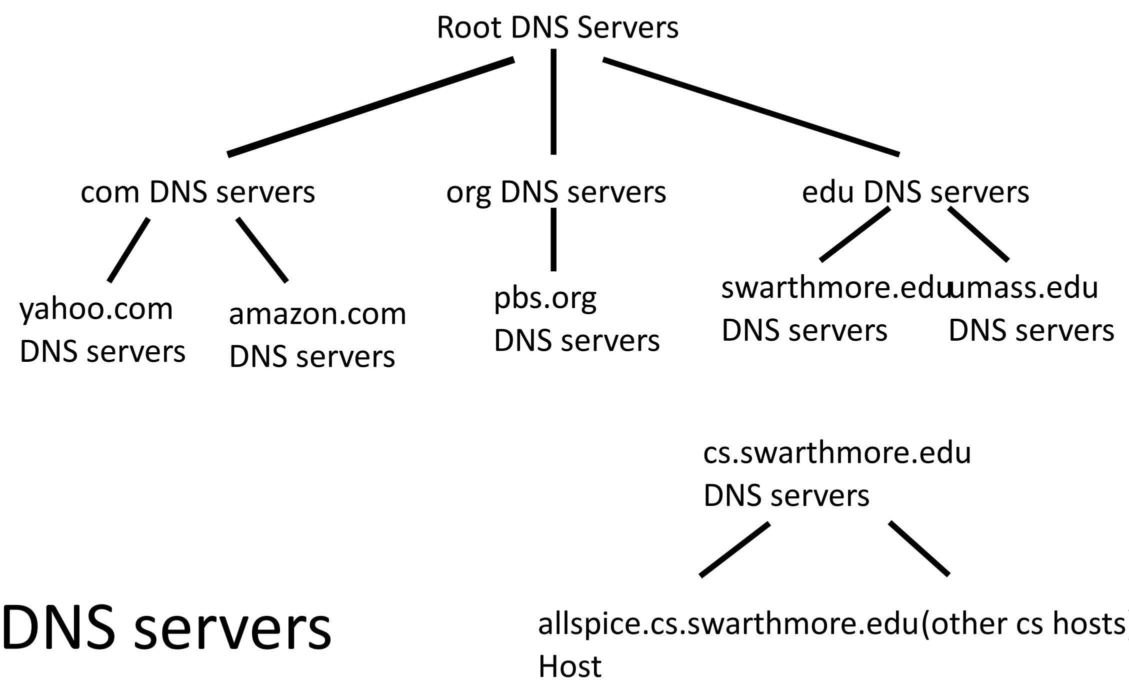


Authoritative Servers

- Organization’s own DNS server(s),
  - for organization’s named hosts
  - authoritative hostname - IP mappings

- maintained by:
  - organization or service provider, easily changing entries
  - Often, but not always, acts as organization’s local name server (for responding to look-ups)

Resolution Process: As an end host if you want to look up a hostname (umass.edu) who do you contact?



A. Contact the swarthmore DNS servers

B. Contact edu DNS servers

C. Contact the Root DNS servers

D. Someone else should do this job.

Local DNS Name Server

- Each ISP
  - (residential ISP, company, university) ...
  - has (at least) one

- also called "default name server"

DNS query host –> local DNS server

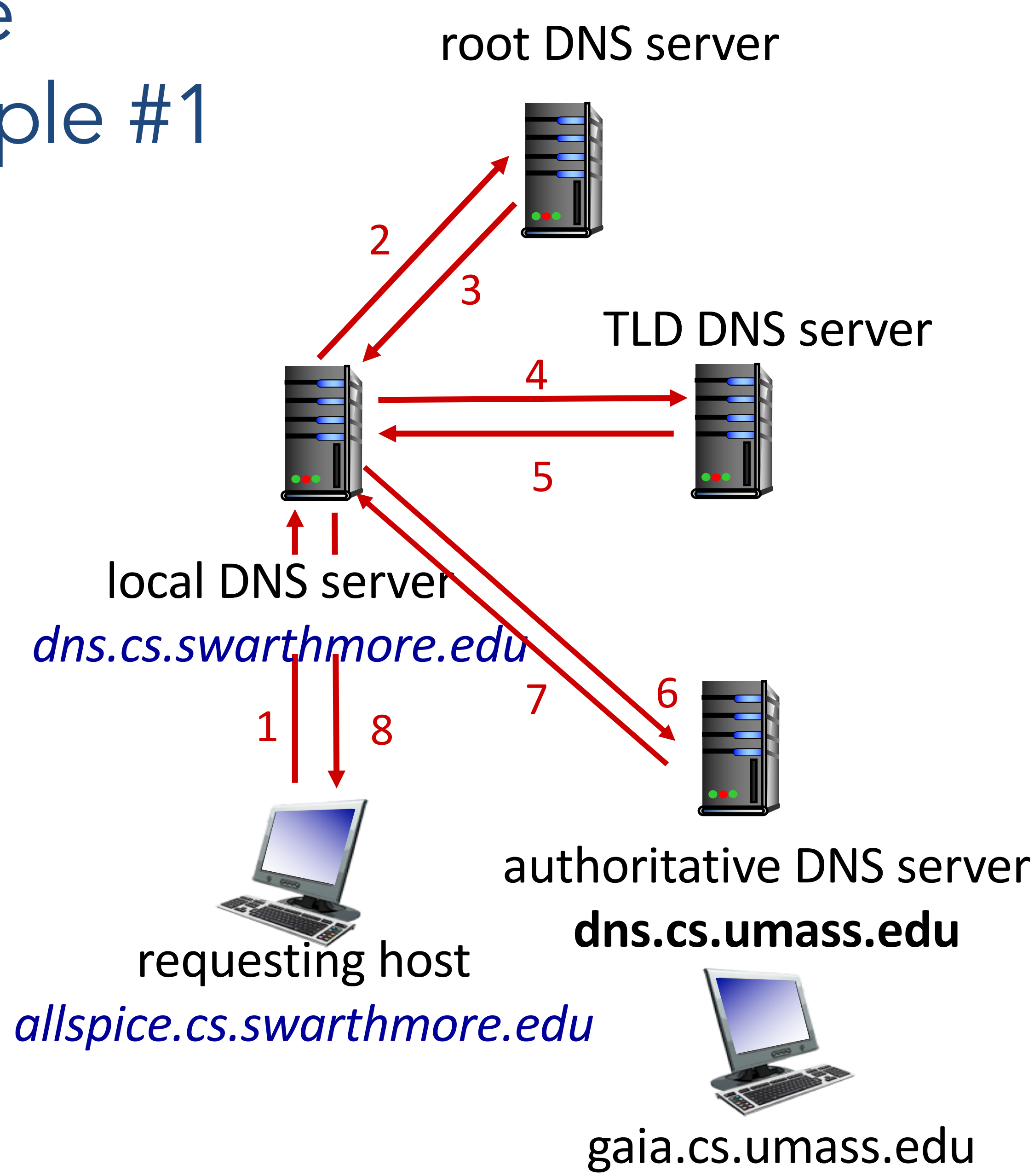
- Local DNS server
  - acts as proxy, forwards query into hierarchy
  - has local cache of recent name-to-address translation pairs (but may be out of date!)

DNS name resolution example #1

allspice wants IP address for gaia.cs.umass.edu

iterative query:

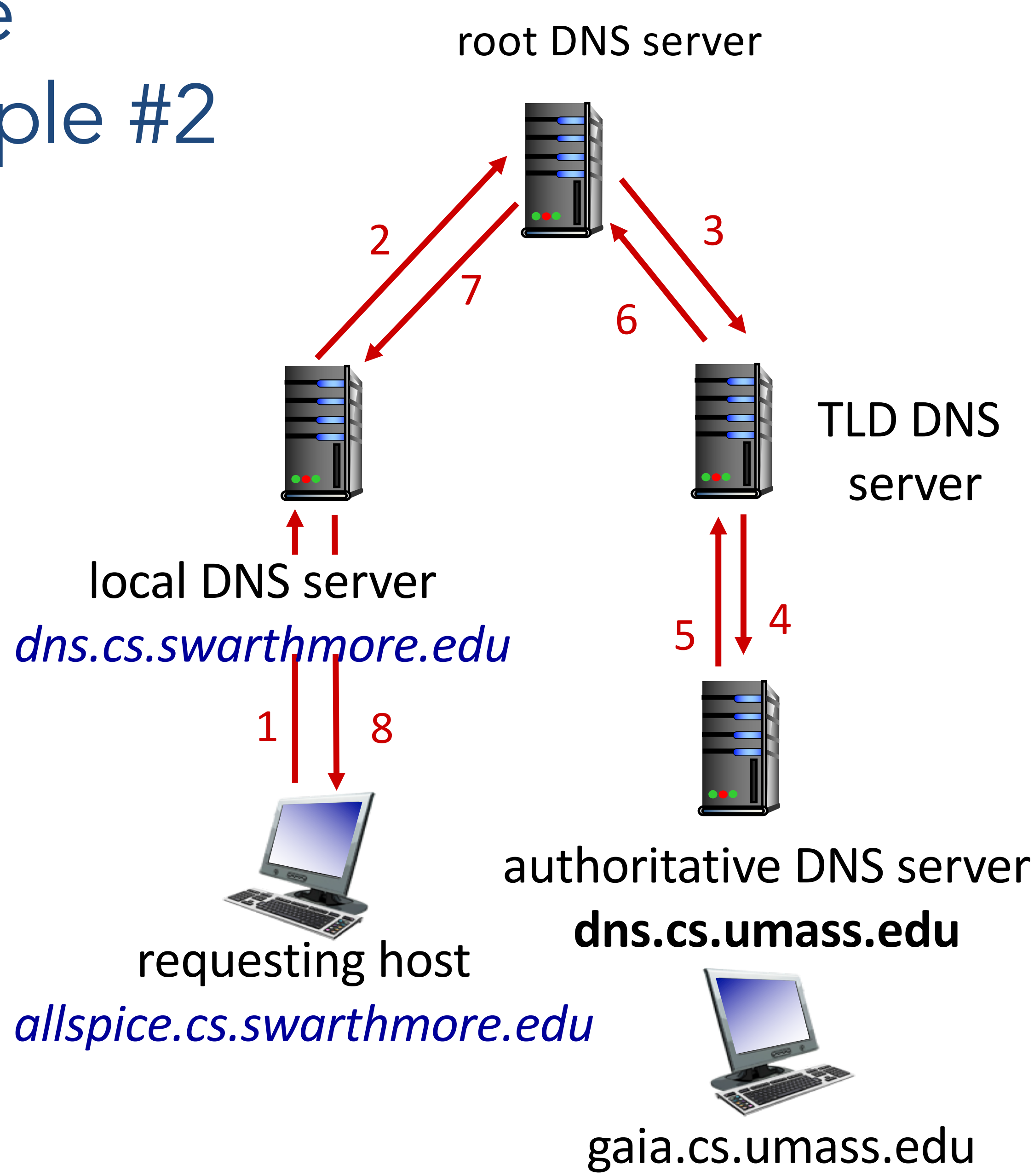
- contacted server replies with name of server to contact
- “I don’t know this name, but ask this server”



DNS name resolution example #2

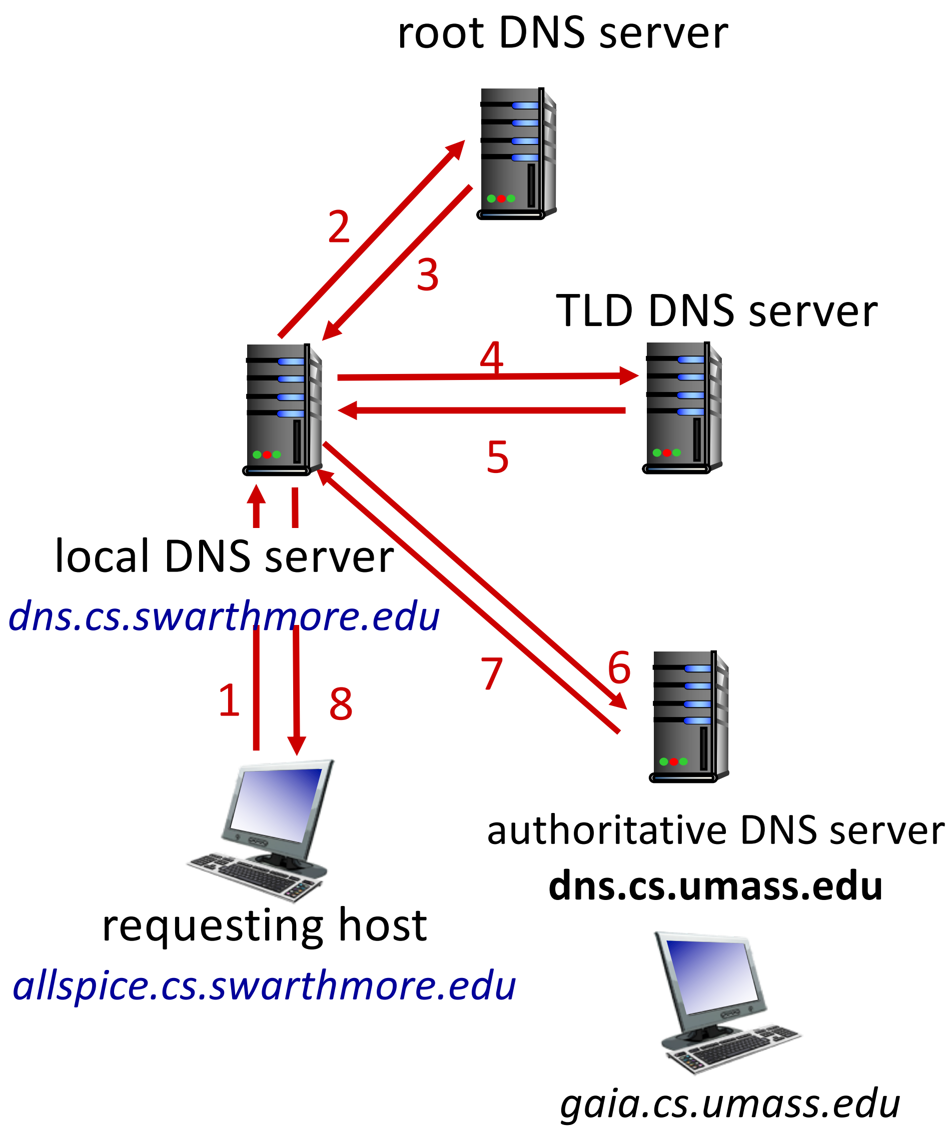
recursive query:

- each server asks the next one, in a chain

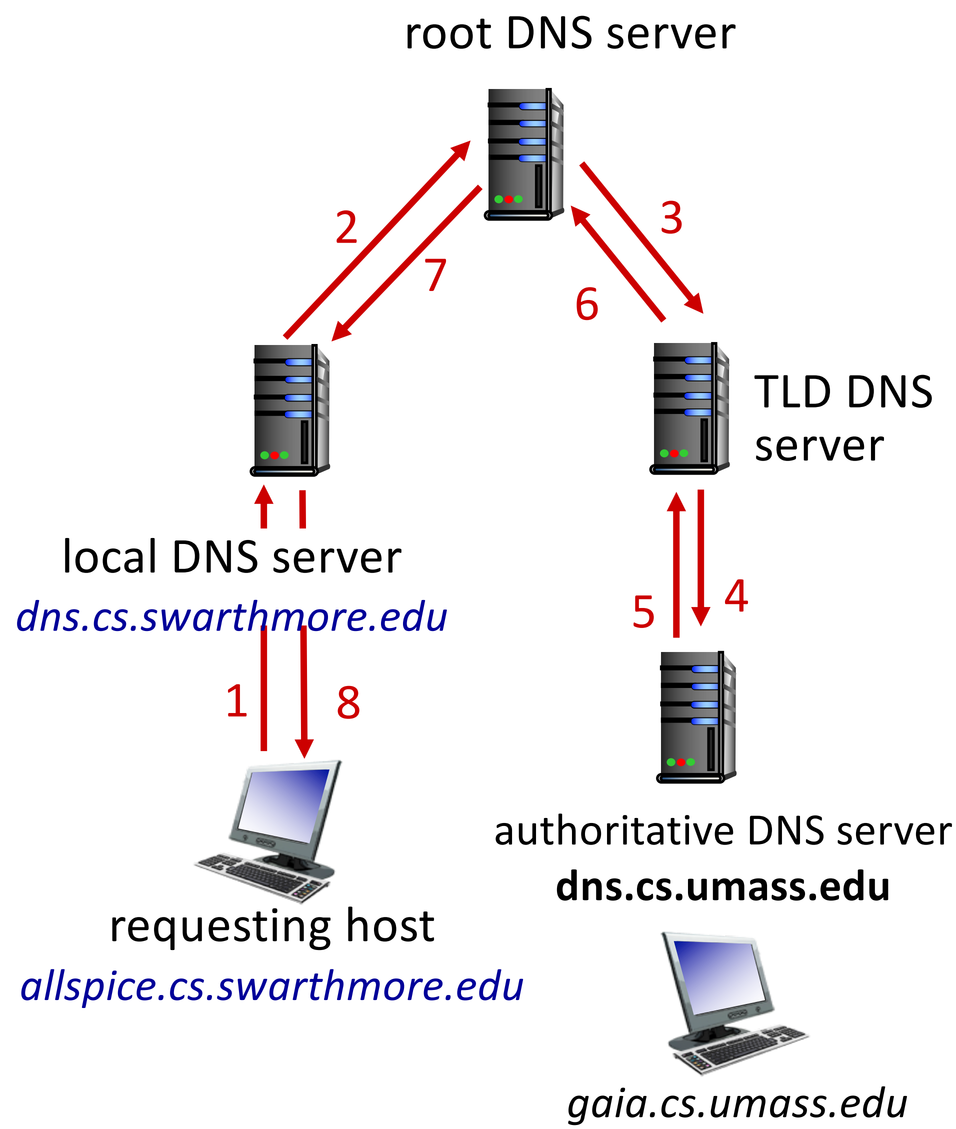


Which would you use? Why?

A. Iterative



B. Recursive

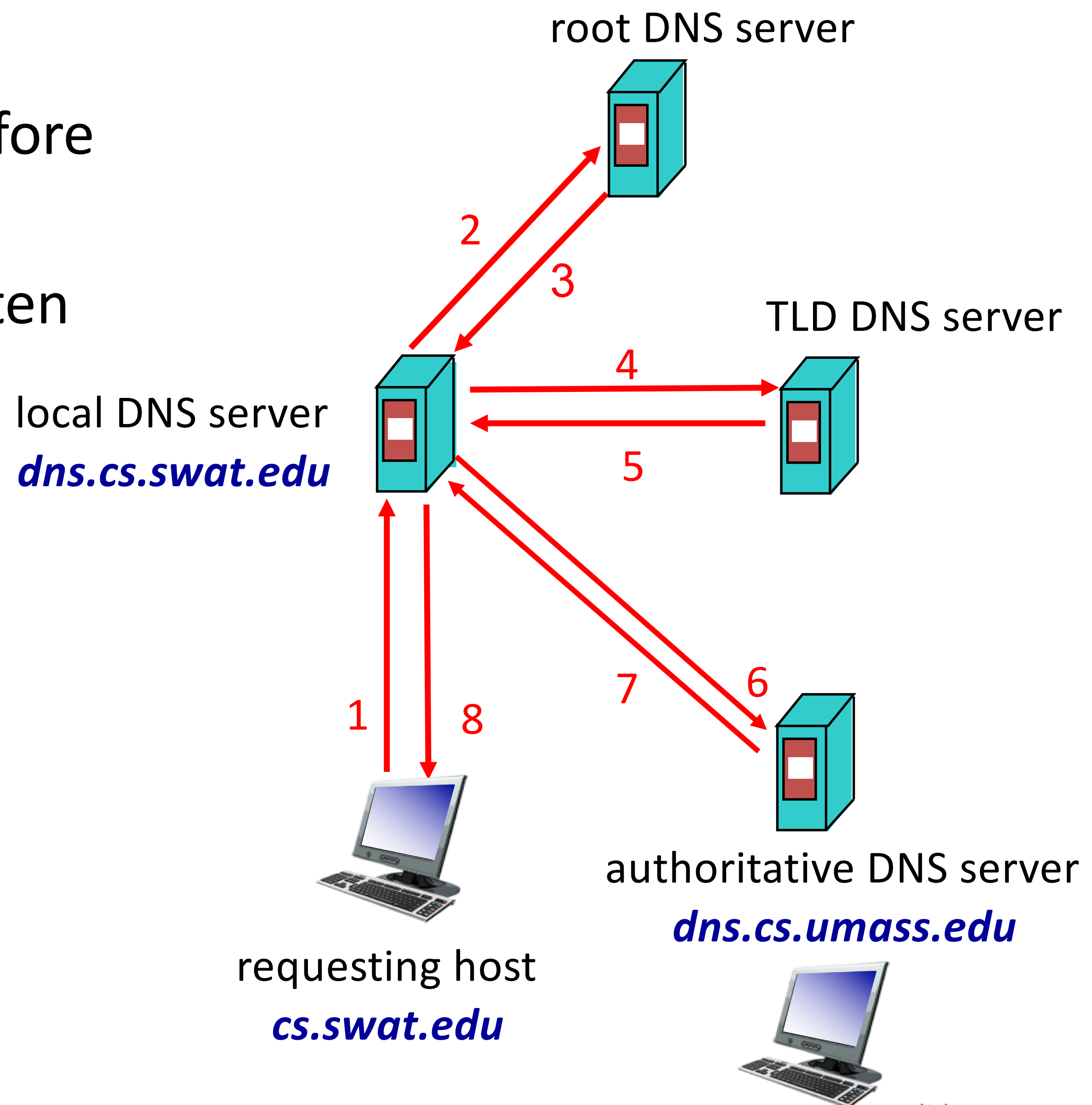


Example: iterative query using dig()

```
dig . ns

dig +norec demo.cs.swarthmore.edu @a.root-servers.net

dig +norec demo.cs.swarthmore.edu @a.edu-servers.net

dig +norec demo.cs.swarthmore.edu @ibext.its.swarthmore.edu

    demo.cs.swarthmore.edu.  259200 IN A  130.58.68.26
```

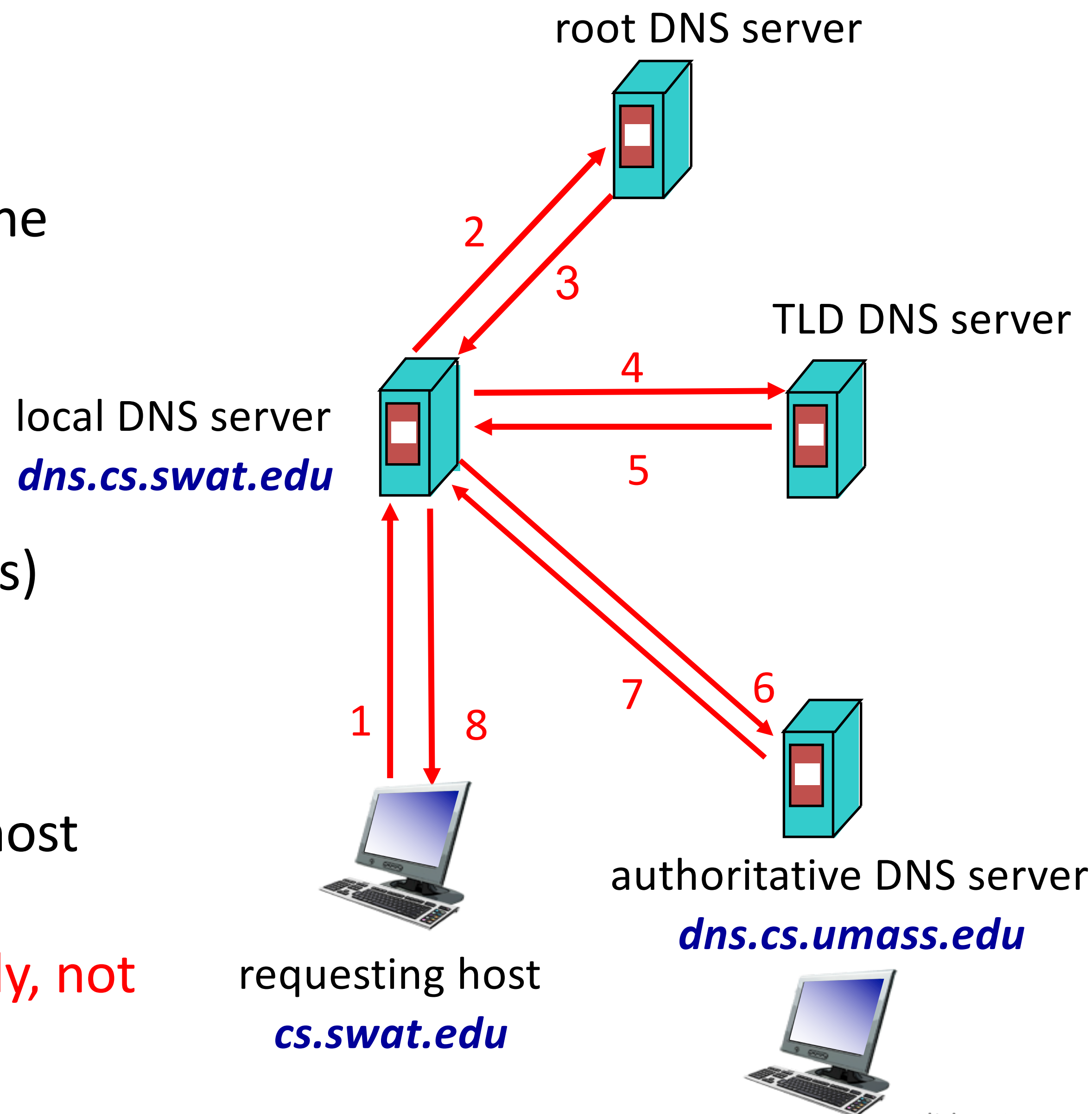
DNS Caching

- Why cache?
  - apprx. 1 sec latency before starting a download
  - Popular sites visited often
- Where to cache?
  - Local DNS server
  - Browser



DNS Caching

- When to cache?
  - learn a mapping? cache!
  - any name server can cache
- For how long?
  - until Time To Live (expires)
- What to cache?
  - TLD servers cached – almost never change
  - Root name servers usually, not visited legitimately



The TTL value should be…

A. Short, to make sure that changes are accurately reflected

B. Long, to avoid re-queries of higher-level DNS servers

C. Something else

Caching

- (+) Subsequent requests need not burden DNS
- (-) Cached entries may be out-of-date (best effort!)
  - If host's name or IP address changes, it may not be known Internet-wide until all TTLs expire

Inserting (or changing) records

Example: new startup “Network Utopia”

Inserting (or changing) records

Example: new startup "Network Utopia"

- Step 1: Register networkuptopia.com at DNS registrar
  - provide names, IP addresses of authoritative name server (primary and secondary)

Inserting (or changing) records

Example: new startup "Network Utopia"

- Step 2: Registrar inserts two RRs into .com TLD server
    - (networkutopia.com, dns1.networkutopia.com, NS)
    - (dns1.networkutopia.com, 212.212.212.1, A)

Inserting (or changing) records

Example: new startup “Network Utopia”

- Step 3: Set up authoritative server at that name/address
  - Create records for the services:

Inserting (or changing) records

Example: new startup “Network Utopia”

- Step 3: Set up authoritative server at that name/address
  - Create records for the services:
    - type A record for www.networkuptopia.com
    - type MX record for @networkutopia.com email

The DNS system can be attacked because

A. can’t tell of reply comes from correct source
B. can’t tell if correct source tells the truth
C. malicious source can insert extra (mis)information
D. malicious bystander can spoof (mis)information
E. All of the above

Attacking DNS

DDoS attacks

- Bombard root servers with traffic
  - Not successful to date
  - Traffic Filtering
  - Local DNS servers cache IPs of TLD servers, bypassing root
- Bombard TLD servers
  - Potentially more dangerous

Redirect attacks

- Man-in-middle
  - Intercept queries
- DNS poisoning
  - Send bogus replies to DNS server that caches

Exploit DNS for DDoS

- Send queries with spoofed source address: target IP
- Requires amplification

Other DNS Uses

- Use of DNS for geo-replicated content
  - Customized responses to queries
  - Inferring the user's location
  - Knowing the user's IP address
- Policy issues
  - Use of DNS to block access to Web sites
  - Collateral damage of DNS injection
  - Redirecting DNS for ads and profit (e.g., Paxfire)

Summary

- DNS maps human readable names to IP addresses

- DNS arranged into a hierarchy
  - Scalability / distributed responsibility
  - Autonomous control of local name servers

- Caching crucial for performance

- DNS has no authentication