Lecture 03-2: Application Layer – DNS

CS 326 Elements of Networking

Mikyung Han

mhan@cs.utexas.edu

Internet Reference Model

Example Protocols		Responsible for
FTP, HTTP, SMTP	Application	application specific needs
TCP, UDP	Transport	process to process data transfer
IP	Network	host to host data transfer across different network
Ethernet, WiFi	Link	data transfer between physically adjacent nodes
802.3 PHY	Physical	bit-by-bit or symbol-by-symbol delivery

2

Hands-on 1 : DNS Dig is out

- Due next Wednesday
- Worth 40 pts

By what internet hosts and routers are identified?

`142.250.115.101`

But people are bad with remembering numbers

`google.com`

Need convert “names” to IP and vice versa

DNS provides such translation service!

What is DNS: Domain Name System

- distributed database implemented in hierarchy of many name servers
- application-layer protocol: hosts communicates with DNS servers to resolve names (address/name translation)
  - Follows dumb network and smart endpoints principle

Any words that needs explanation/clarification here?

DNS also provides other services

Besides hostname-to-IP address translation

- host aliasing
  - You have seen this already. linux.cs.utexas.edu (it is actually key.cs.utexas.edu)
- mail server aliasing
  - Two different mailing addresses (user@ex.com and user@ex.org) delivered to same mail server)
- load distribution
  - How?

Why don't we have just one powerful DNS server?

Q: Why not centralized DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance

A: doesn't scale!

- Comcast DNS servers alone: 600B DNS queries/day
- Akamai DNS servers alone: 2.2T DNS queries/day

Engineering questions about the DNS

humongous distributed database:

- ~ billion records, each simple

handles many trillions of queries/day:

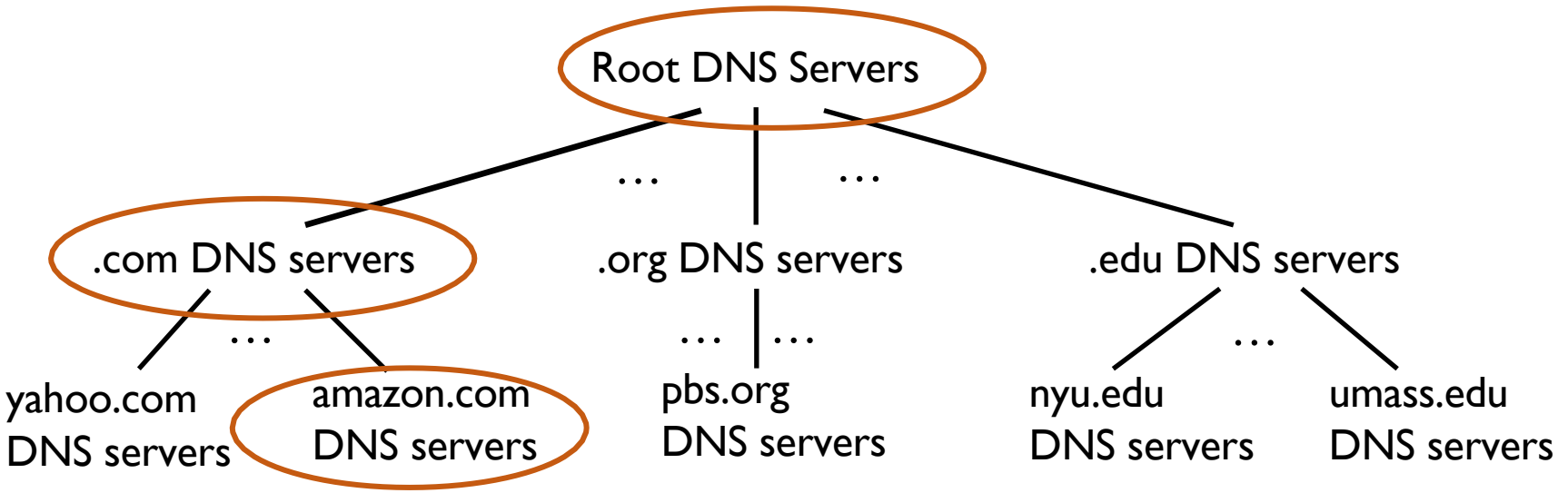
- many more reads than writes
- performance matters: almost every Internet transaction interacts with DNS - msecs count!

organizationally, physically decentralized:

- millions of different organizations responsible for their records

"bulletproof": reliability, security

DNS: a distributed, hierarchical "database"



Root

Top Level Domain

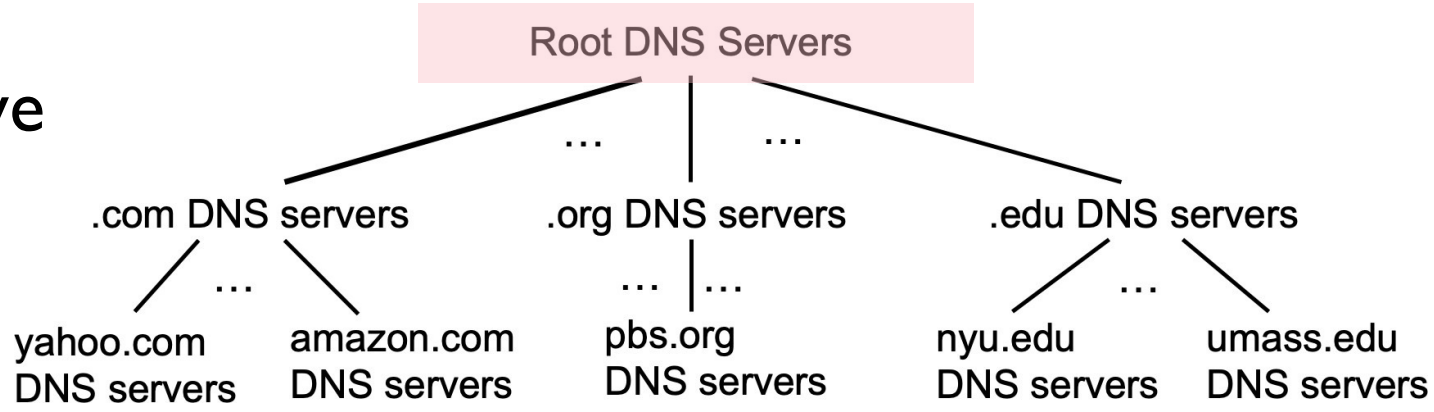
Authoritative

Client wants IP address for www.amazon.com thus asks its local DNS resolver

- First root server is queried
- Next .com TLD DNS server is queried
- Last amazon.com DNS server is queried

DNS: root name servers

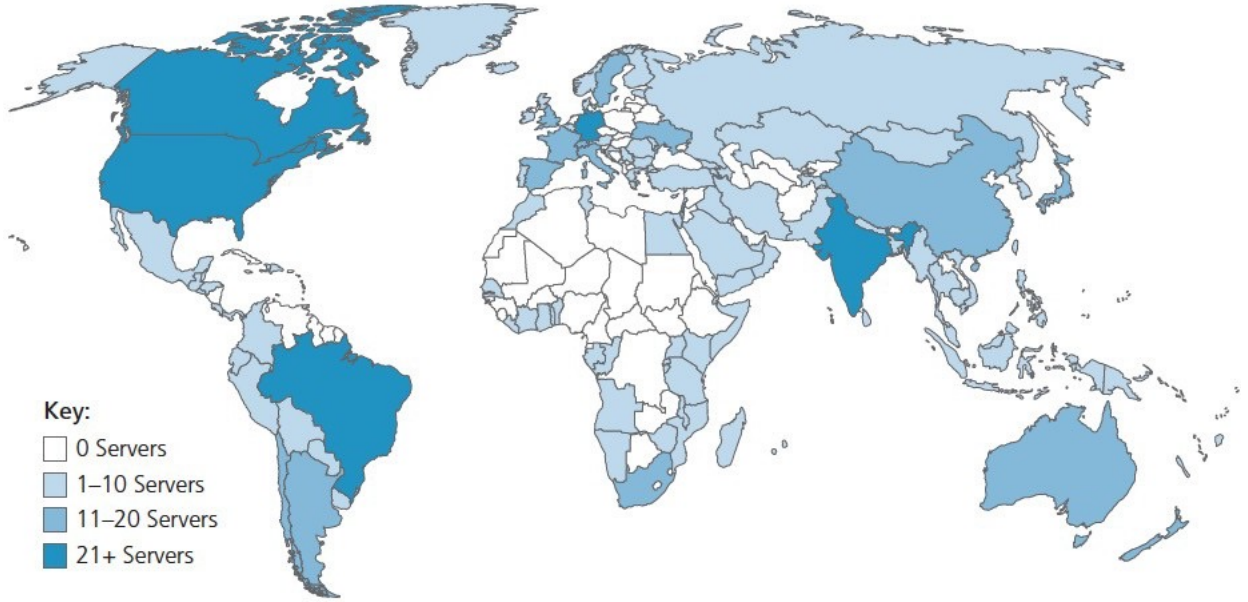
- Official contact by name servers that can not resolve name



DNS: root name servers

- official, contact-of-last-resort by name servers that can not resolve name
- incredibly important Internet function
  - Internet couldn't function without it!
  - DNSSEC – provides security (authentication, message integrity)
- ICANN (Internet Corporation for Assigned Names and Numbers) manages root DNS domain

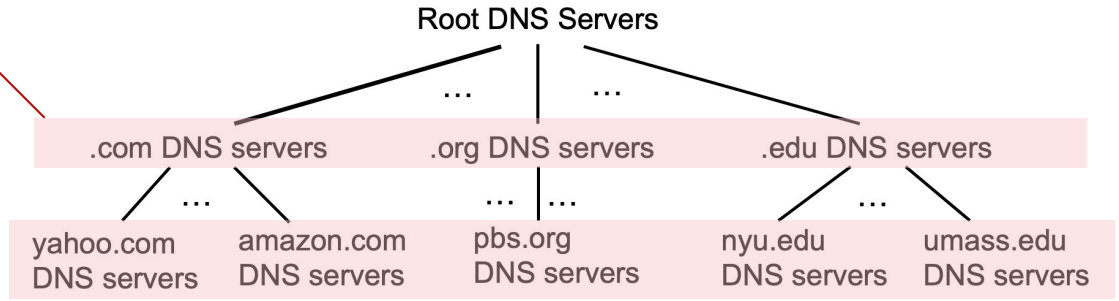
13 logical root name "servers" worldwide each "server" replicated many times (~200 servers in US)



Top-Level Domain, and authoritative servers

Top-Level Domain (TLD) servers:

- responsible for .com, .org, .net, .edu, .aero, .jobs, .museums, and all top-level country domains, e.g.: .cn, .uk, .fr, .ca, .jp



authoritative DNS servers:

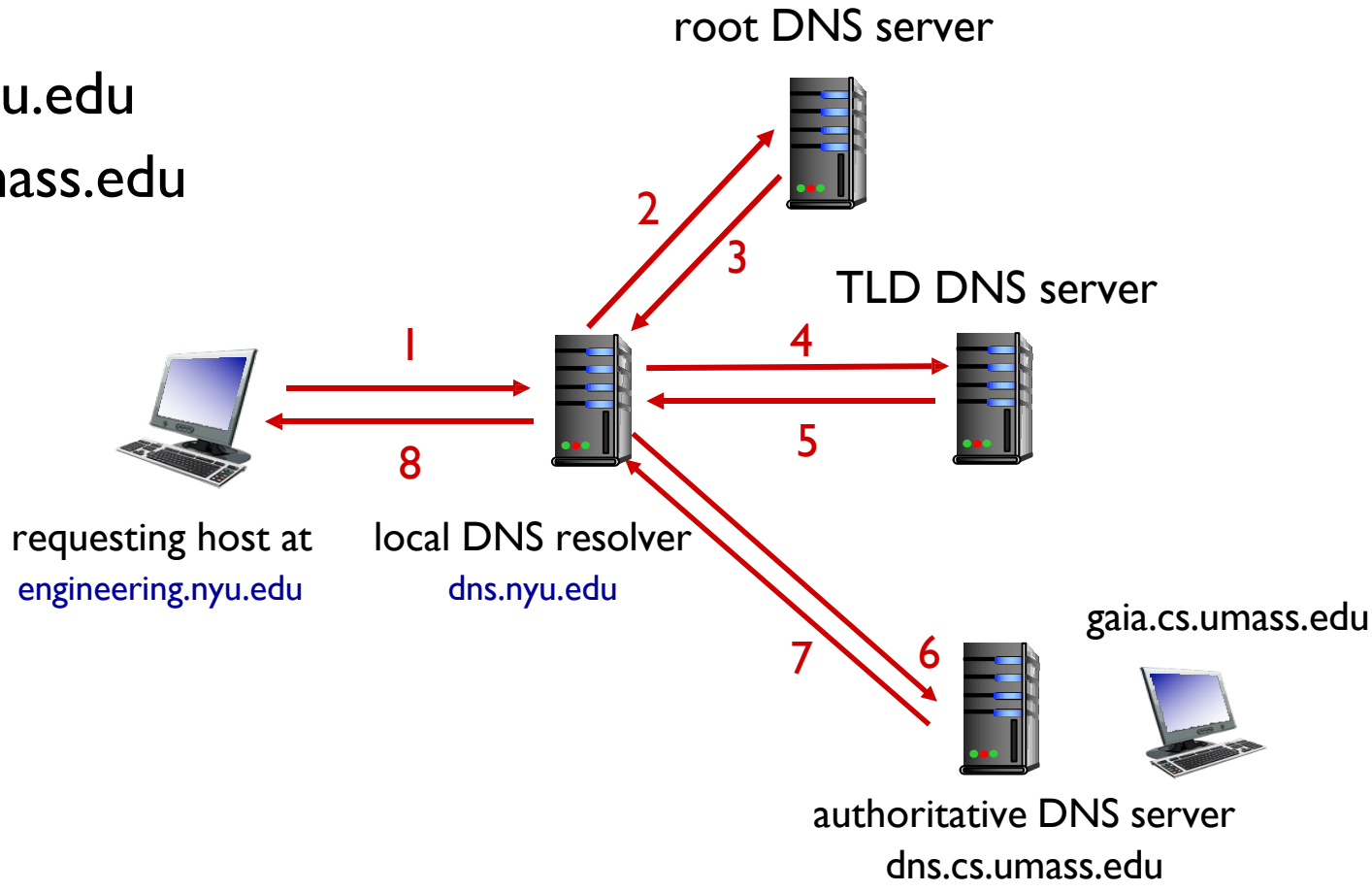
- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

DNS name resolution: iterated query

Example: host at engineering.nyu.edu wants IP address for gaia.cs.umass.edu

Iterated query:

- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



That is a lot of querying. How to improve performance?

Caching in DNS significantly improves response time

- once name server gives out the mapping, DNS resolver caches the mapping
- Next time upon asked the same query it can return the answer immediately from its cache
- Also, caching can shorten response time for similar query

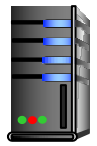
DNS name resolution with cache (example 1)

Example: host at engineering.nyu.edu queries again for gaia.cs.umass.edu

Local Cache lookup!

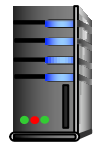
- Host itself has cache
- Very fast
- No need to go to DNS resolver

requesting host at engineering.nyu.edu

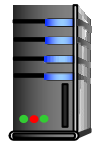


local DNS resolver dns.nyu.edu

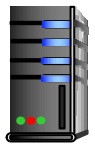
root DNS server



TLD DNS server



gaia.cs.umass.edu

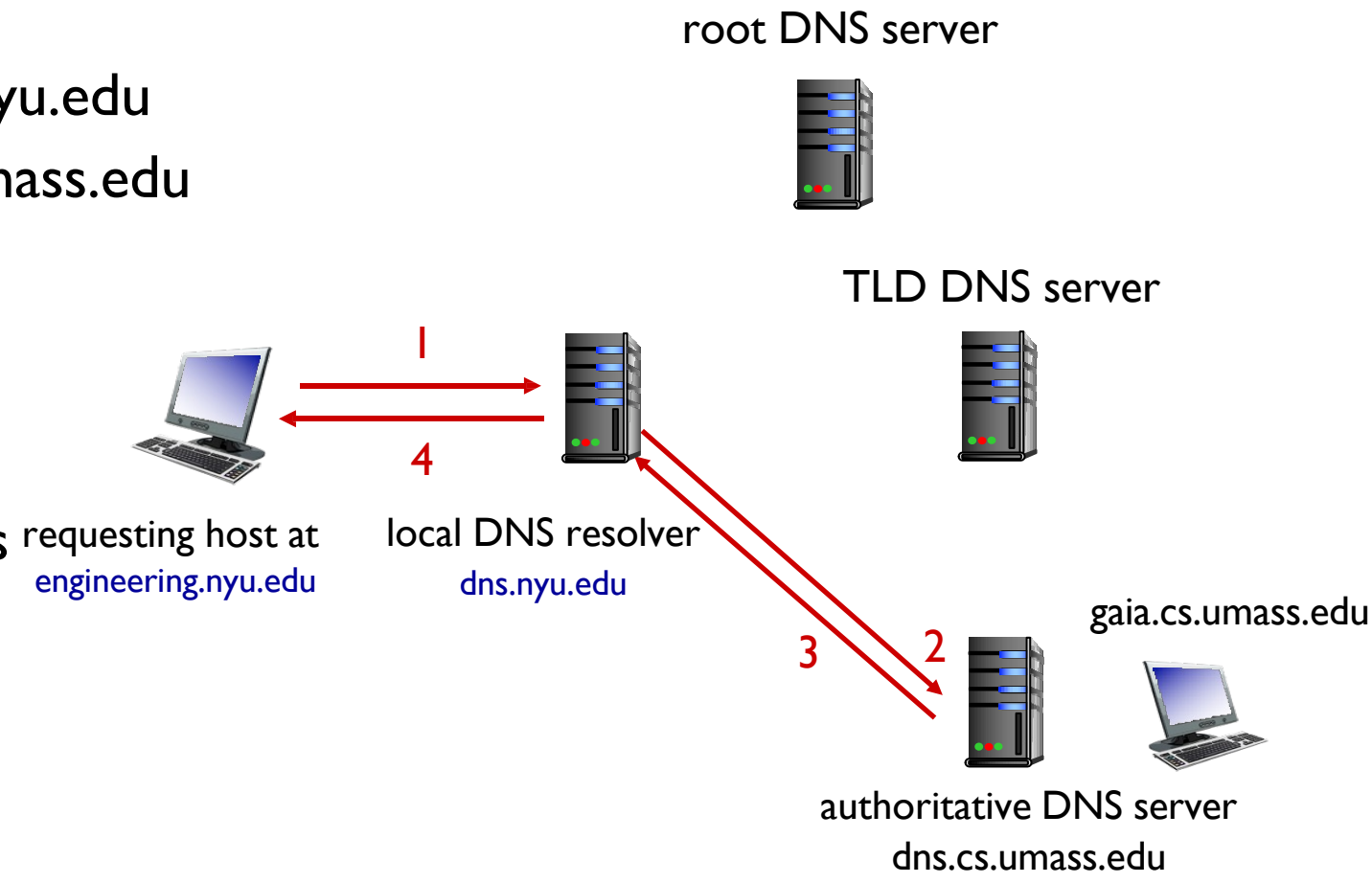


authoritative DNS server dns.cs.umass.edu

DNS name resolution with cache (example 2)

Example: host at engineering.nyu.edu wants IP address for abc.cs.umass.edu

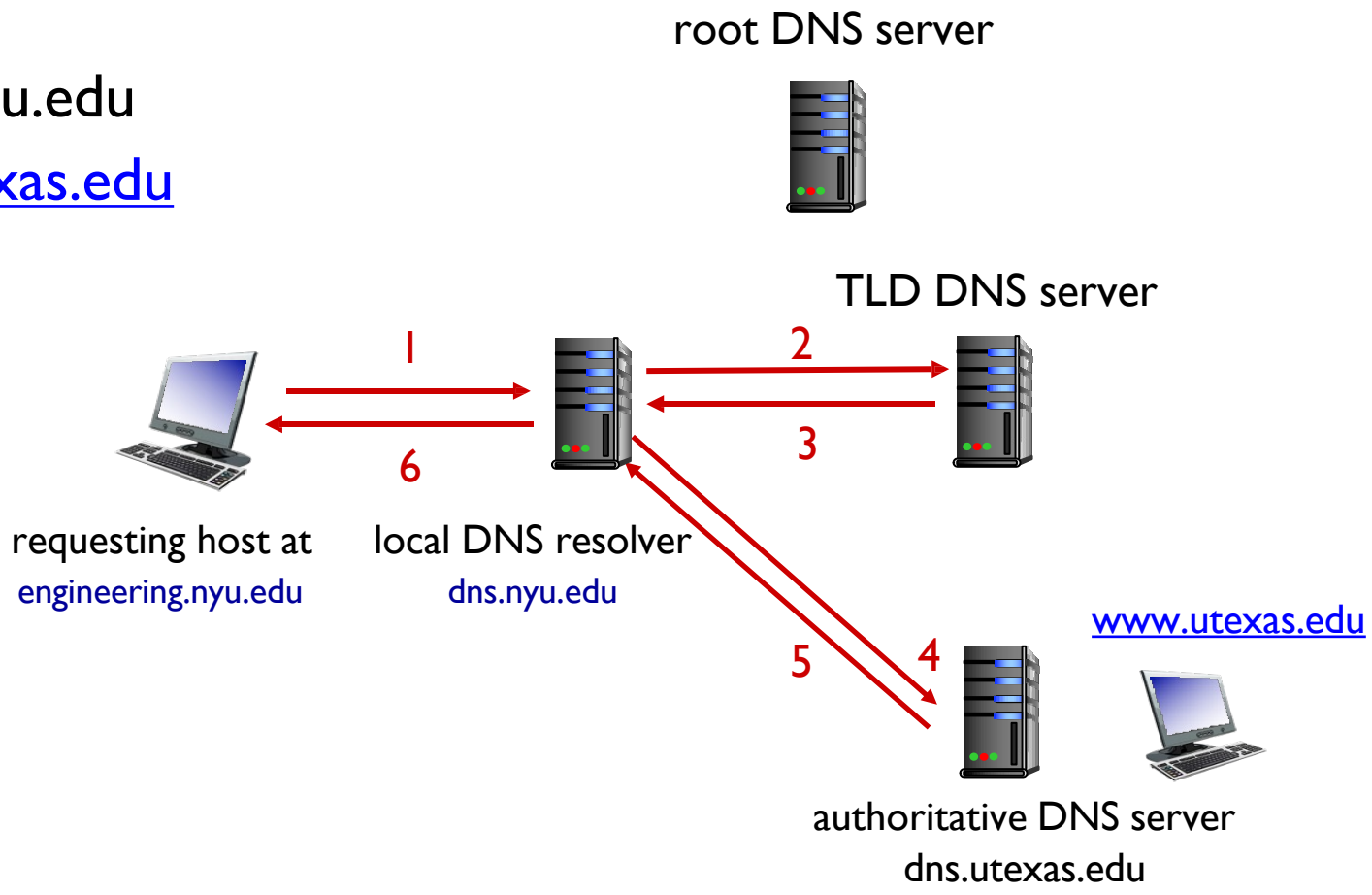
- host cache miss
- Local DNS resolver cached the authoritative DNS server's IP for cs.umass.edu
- Skip asking root DNS or TLD



DNS name resolution with cache (example 3)

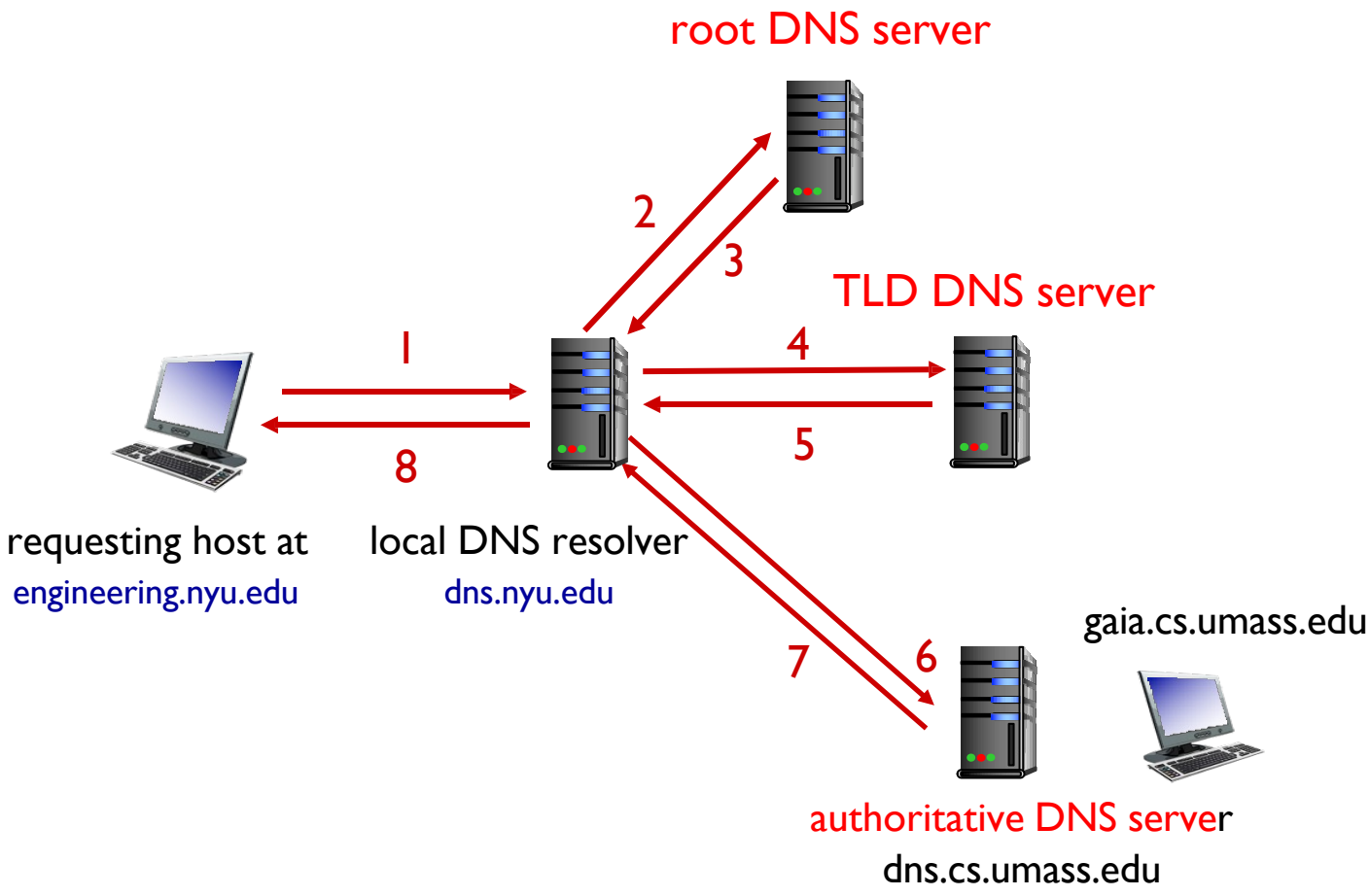
Example: host at engineering.nyu.edu wants IP address for www.utexas.edu

- Host cache miss
- Local DNS resolver knows from cache which TLD to query for .edu
- TLD give authoritative DNS server for www.utexas.edu
- Authoriative DNS server gives the answer
- Skip root DNS server query



Higher-level DNS servers do NOT do caching

- Cache? No thanks.
  - Root name server
  - TLD servers
  - Authoritative DNS servers



Why won't these higher-level servers do caching?

What is more important than performance for the higher-level DNS servers?

- Cache? No thanks.
  - Root name server
  - TLD servers
  - Authoritative DNS servers

Accuracy over performance!

Cache entry can become out-of-date

Stale cache clears out after TTL

- cache entries timeout (disappear) after some time (TTL)
- TLD servers typically cached in local name servers
  - if named host changes IP address, may not be known Internet-wide until all TTLs expire!
  - best-effort name-to-address translation!

Things to think about

What else could go wrong in DNS?

DDoS attacks

- bombard root servers
  - not successful to date
  - traffic filtering
  - local DNS resolver caches IPs of TLD servers, allowing root server bypass
- bombard TLD servers
  - potentially more dangerous

Spoofing attacks

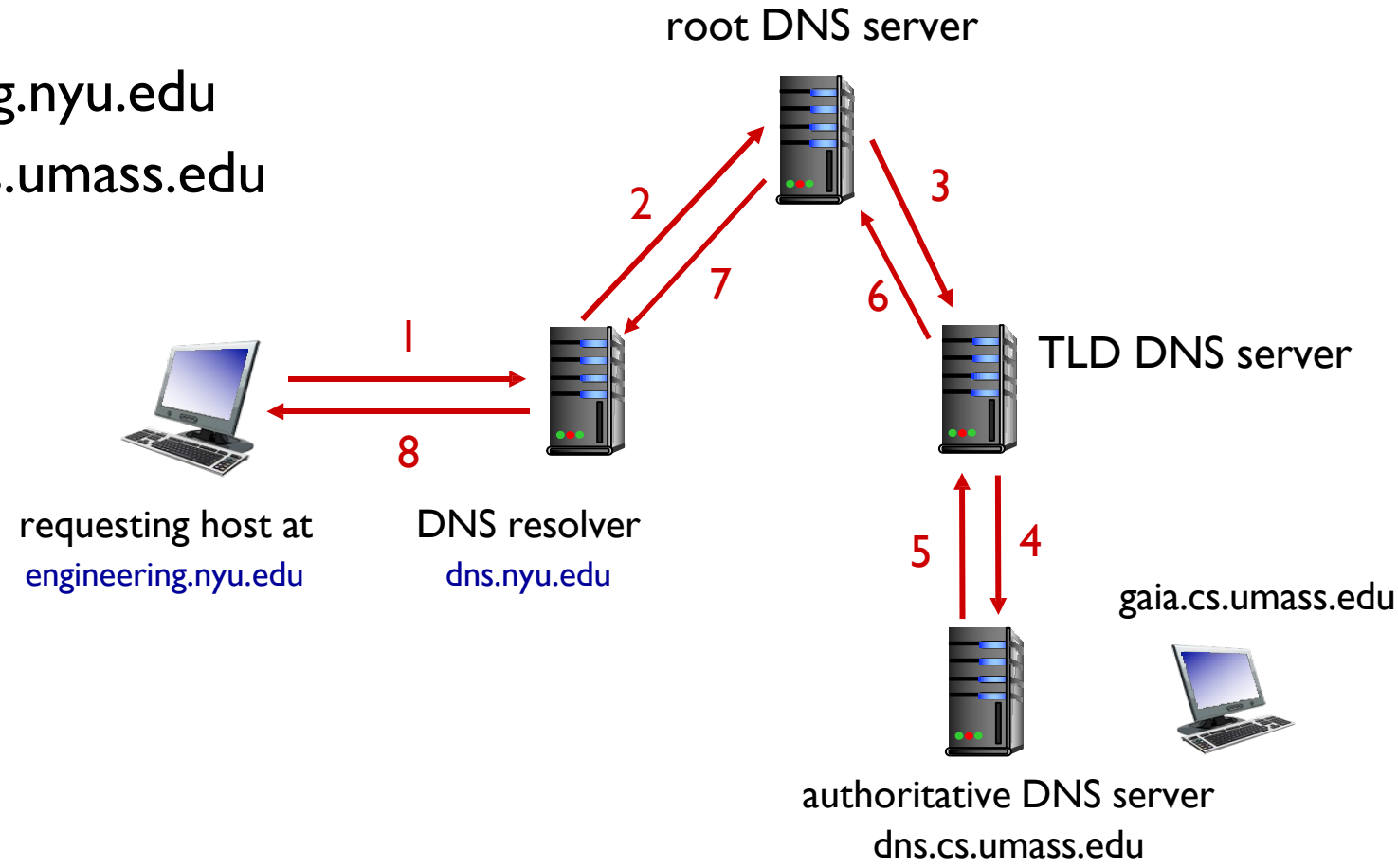
- intercept DNS queries, returning bogus replies
  - DNS cache poisoning
  - RFC 4033: DNSSEC authentication services

What could be the other option? recursive query

Example: host at engineering.nyu.edu wants IP address for gaia.cs.umass.edu

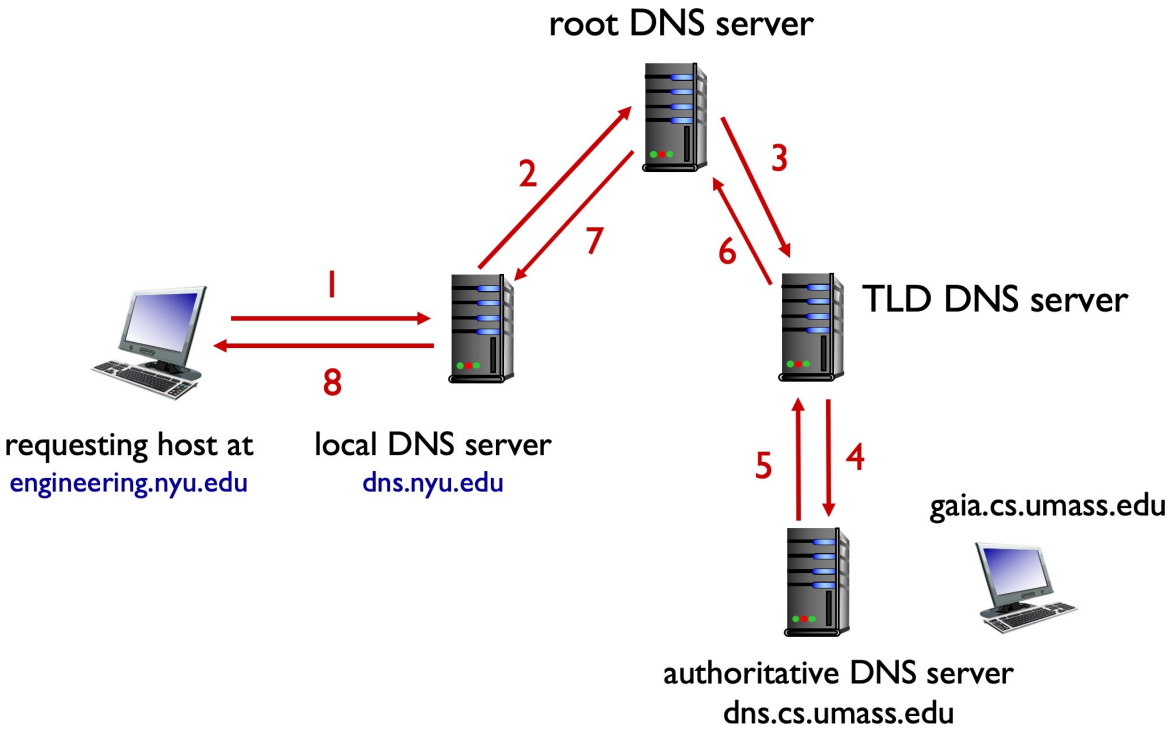
Recursive query:

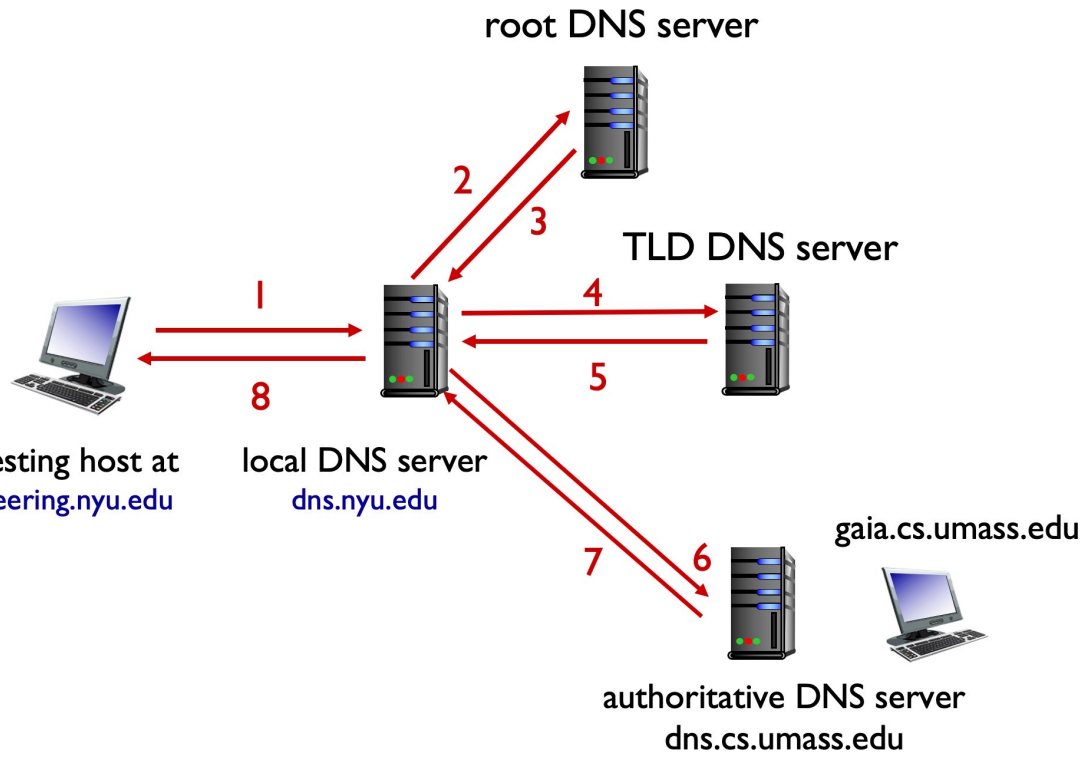
- puts burden of name resolution on contacted name server



Heavy load at upper levels

Assume everybody does caching (not true in reality) Which type of query works better?





Recursive vs Iterated

Who knows the final answer?

DNS Dig terminology (different than slide 24/25 definitions)

Dig's Recursive query

In the context of DNS "dig", a "recursive query" means that when you ask for a domain name's IP address, the DNS server will actively search other DNS servers on your behalf until it finds the answer and returns the complete IP address directly to you; essentially, it does all the work to get the final answer for you.

Dig's Non-recursive query

The DNS server will simply provide a referral to another server if it does not have the answer

T/F? Root Name Server can perform recursive query

How DNS similar/different to/from database service?

- What is similar?
- What is different?

Backup Slides

DNS records

DNS: distributed database 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

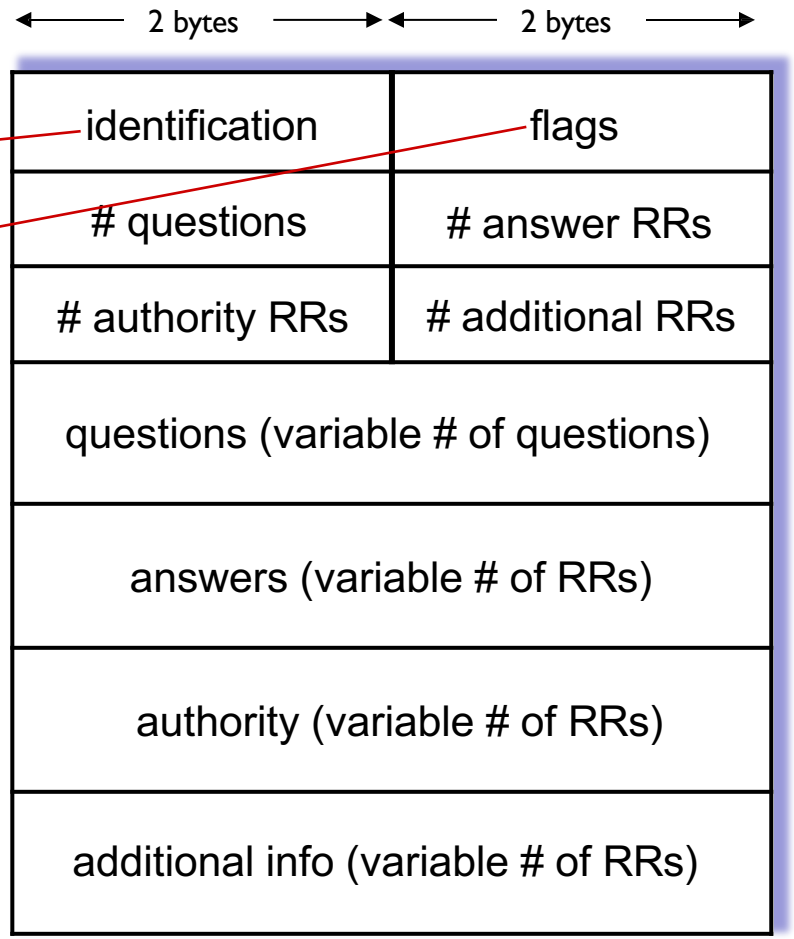
- value is name of SMTP mail server associated with name

DNS protocol messages

DNS query and reply messages, both have same format:

message header:

- identification: 16 bit # for query, reply to query uses same #
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative



Acknowledgements

Slides are adopted from Kurose' Computer Networking Slides