

# Application Layer Protocol: DNS

# DNS: Domain Name System

*People have many identifiers:* name, SSN, passport #

*Internet hosts and routers have:*

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

Q: how to map between IP address and name, and vice versa ?

*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)
  - note: core Internet function, implemented as application-layer protocol
  - complexity at network's “edge”

# DNS: Services, Structure

## *DNS services*

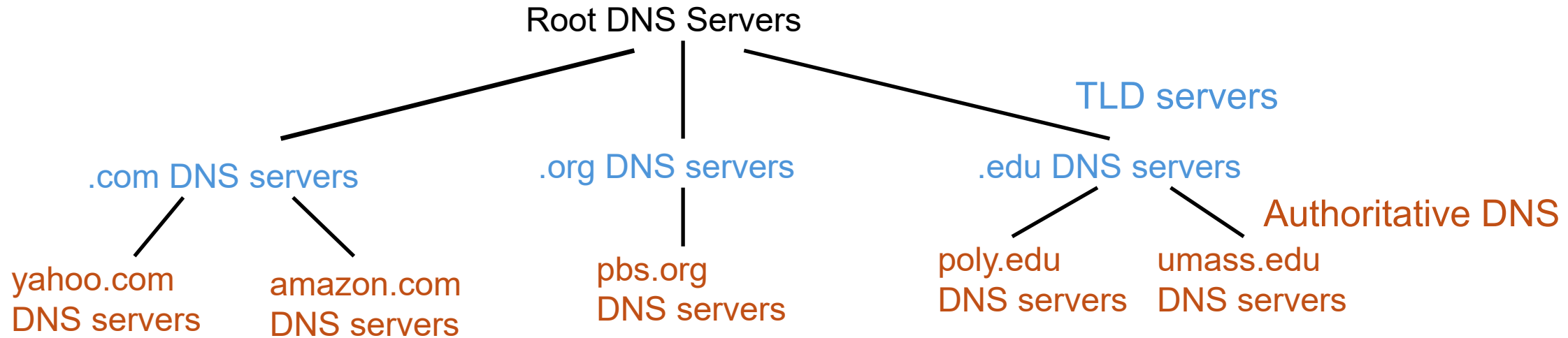
- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

## *why not centralize DNS?*

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

*Ans: doesn't scale!*

# DNS: a distributed, hierarchical database

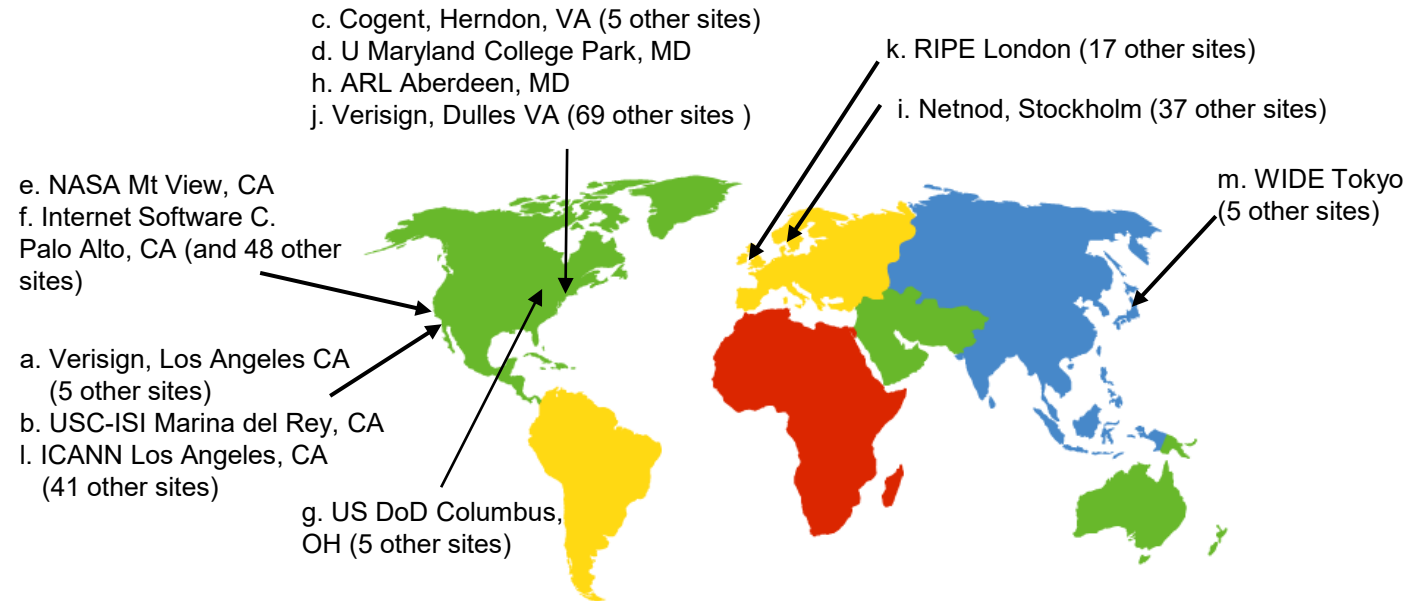


*client wants IP for [www.amazon.com](http://www.amazon.com); 1<sup>st</sup> approx:*

- client queries root server to find .com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for [www.amazon.com](http://www.amazon.com)

# DNS: root name servers

- 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



*13 root name “servers” worldwide*

# TLD, authoritative 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:*

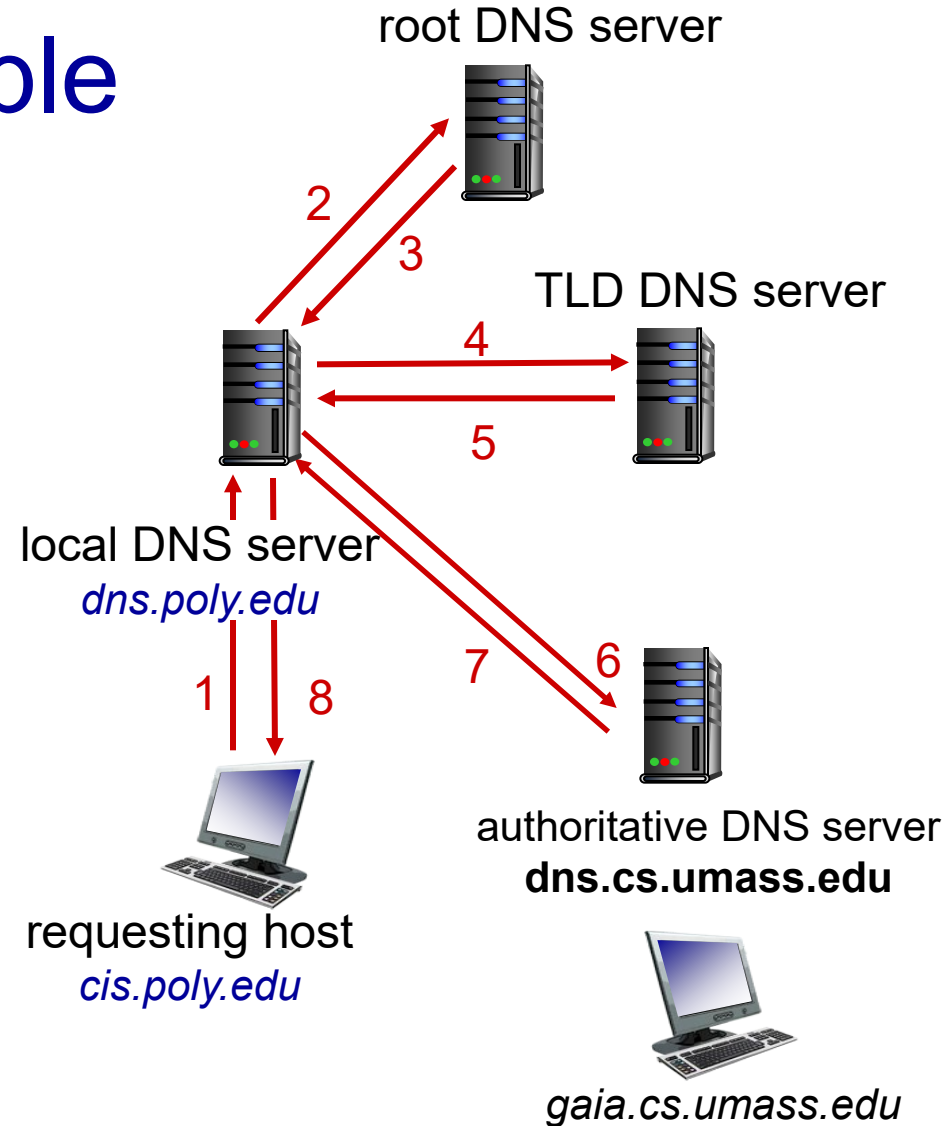
- Organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

# Local DNS name 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

- host at cis.poly.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”

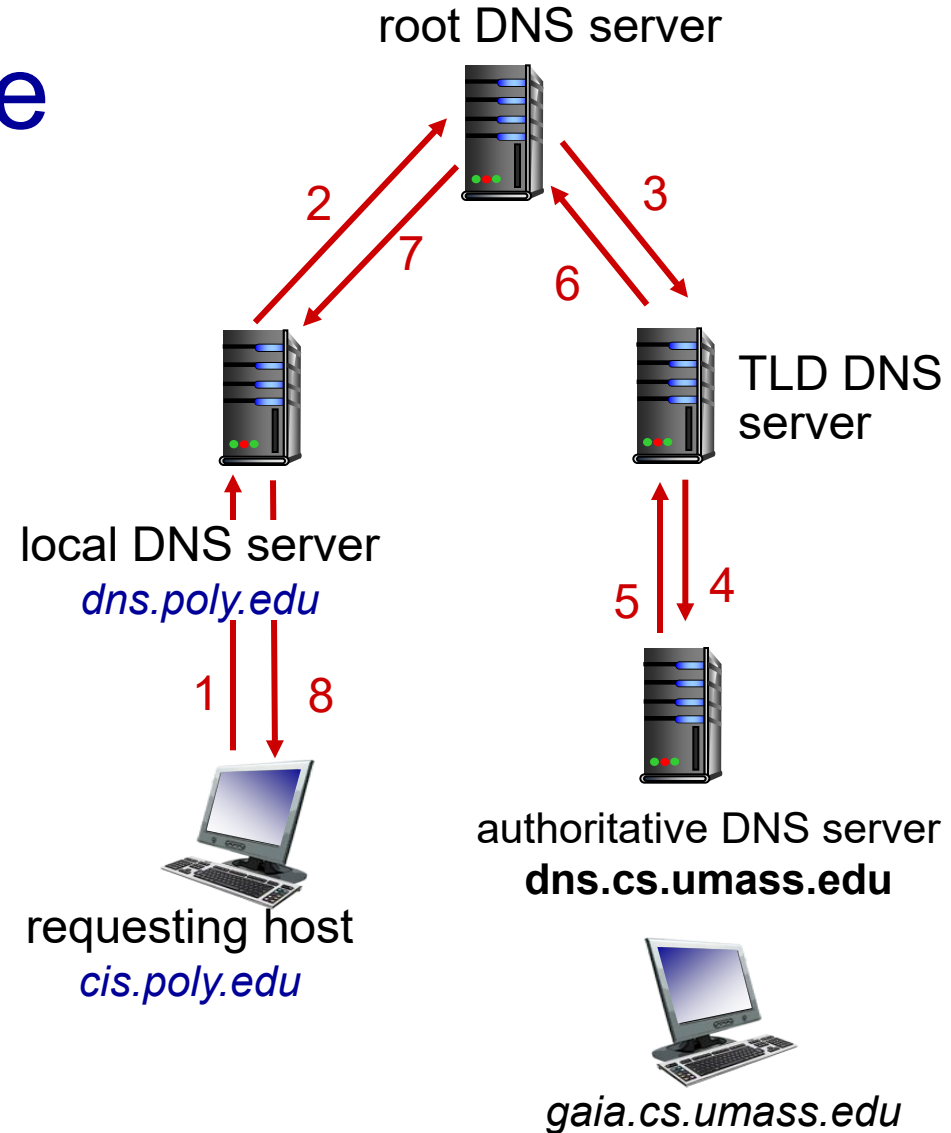




# DNS name resolution example

## *recursive query:*

- ❖ puts burden of name resolution on contacted name server
- ❖ heavy load at upper levels of hierarchy?



# DNS: caching, updating records

- once (any) name server learns mapping, it *cached* mapping
  - cache entries timeout (disappear) after some time (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 records

**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 protocol, messages

- *query* and *reply* messages, both with same *message format*

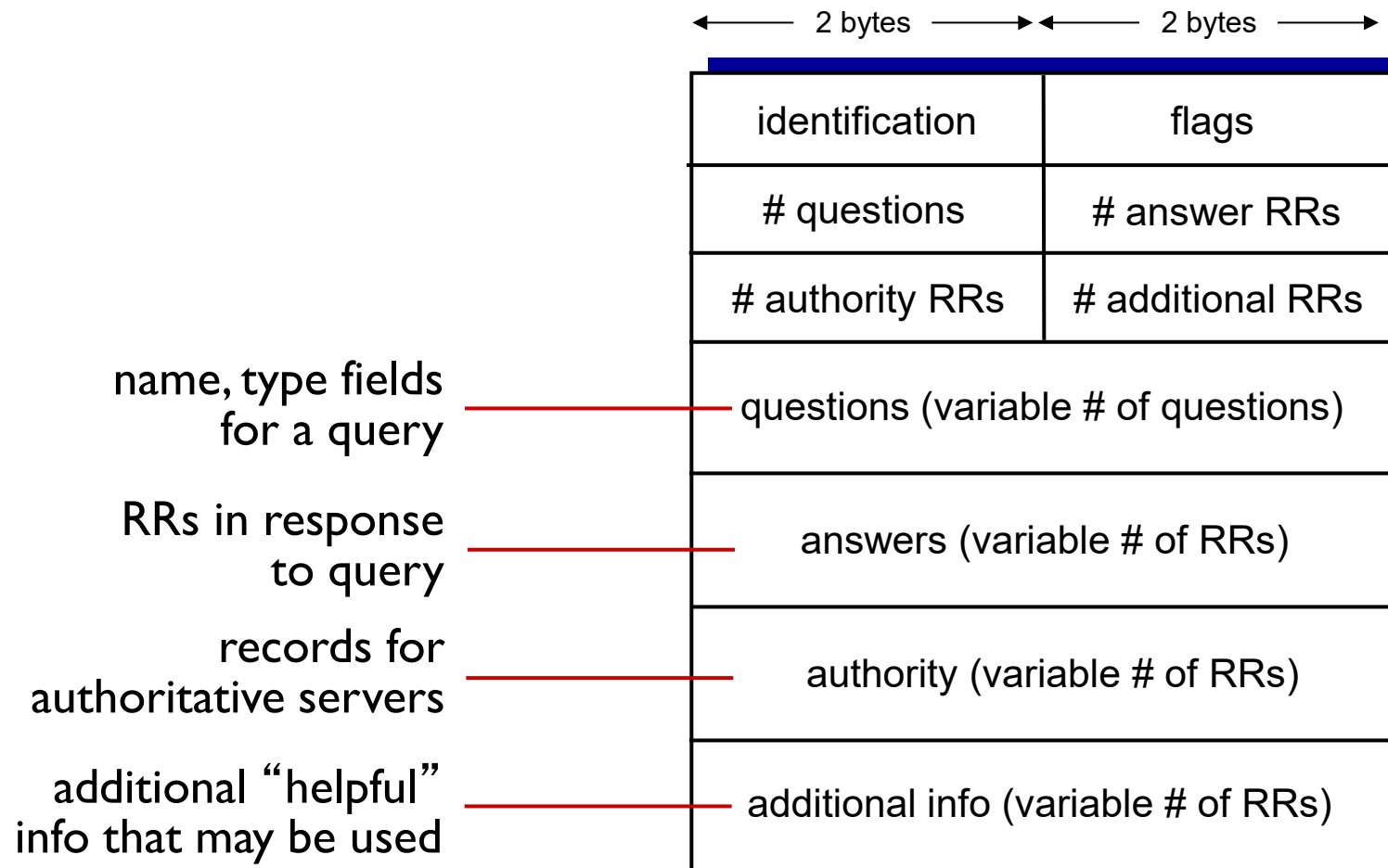
msg header

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

← 2 bytes → ← 2 bytes →

identification	flags
# questions	# answer RRs
# authority RRs	# additional RRs
questions (variable # of questions)	
answers (variable # of RRs)	
authority (variable # of RRs)	
additional info (variable # of RRs)	

# DNS protocol, messages



# Inserting records into DNS

- example: new startup “Network Utopia”
- register name networkutopia.com at *DNS registrar* (e.g., Network Solutions)
  - provide names, 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, 212.212.212.1, A)
- create authoritative server type A record for www.networkutopia.com; type MX record for networkutopia.com

# Attacking DNS

## DDoS attacks

- Bombard root servers with traffic
  - Not successful to date
  - Traffic Filtering
  - 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 bogus replies to DNS server, which caches

## Exploit DNS for DDoS

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