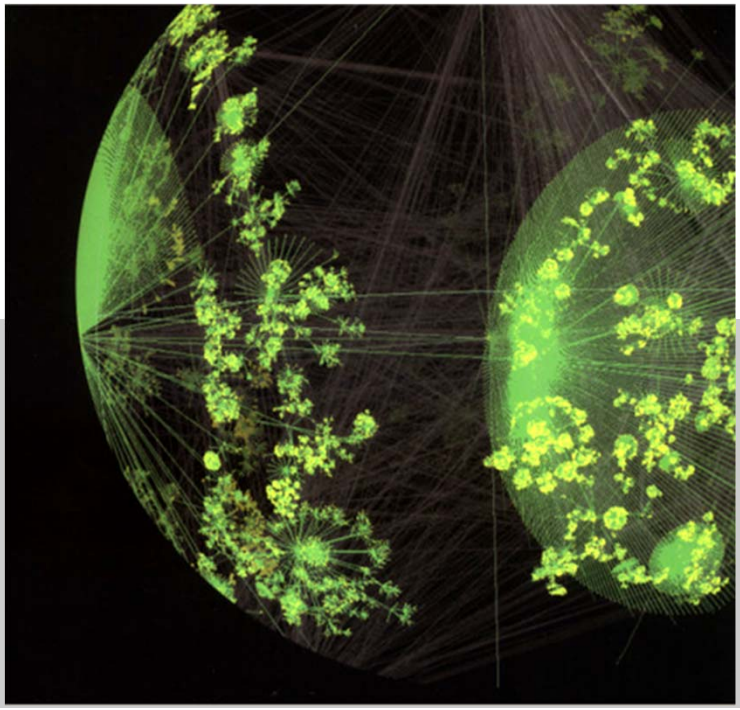
Chapter 1
Linux and TCP/IP Networking



TCP/IP Essentials

A Lab-Based Approach

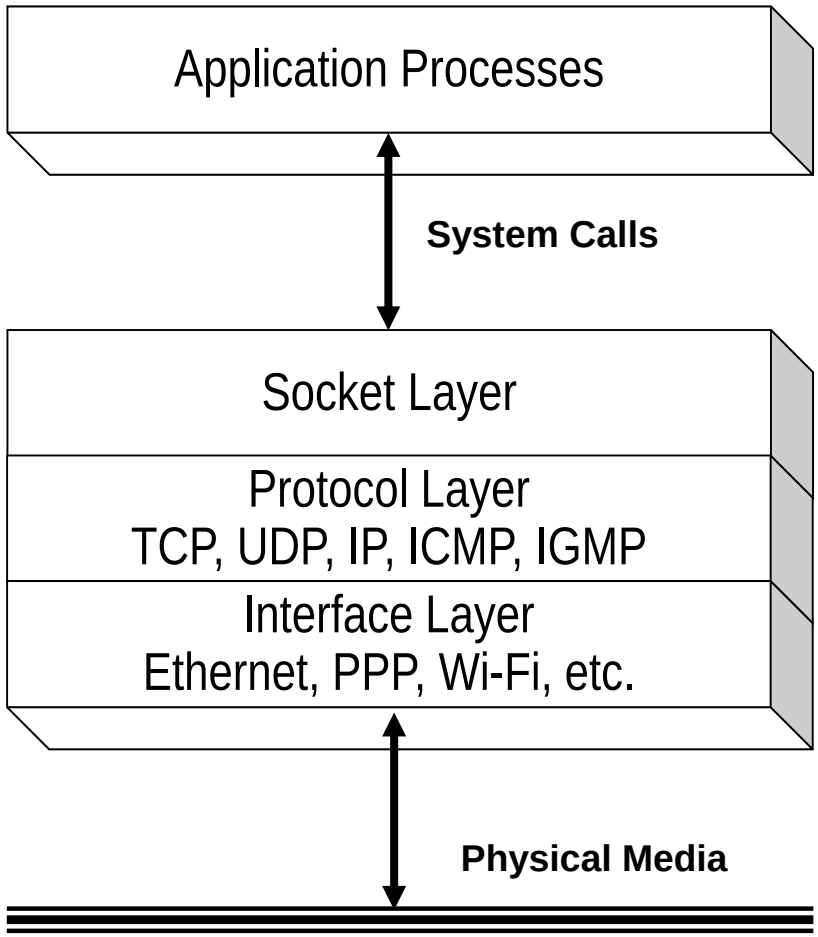
Spring 2017

Linux and TCP/IP Implementations

- Unix: a family of multitasking computer operating systems
  - Developed in 1970s by Bell Labs Research
  - Unix systems included TCP/IP utilities in Berkeley Software Distribution (BSD) release 4.1c
- The first widely available Unix release with TCP/IP is 4.2 BSD
- Unix TCP/IP implementations
  - Solaris
  - FreeBSD
  - Linux

Networking Code Organization

- Most applications are implemented as *User Space* processes.
- Protocols are implemented in the system kernel
  - Socket layer
  - Protocol layer
  - Interface layer



Network Daemons and Services

- Daemon: a process running in the background of the system. popular network daemons are managed by
  - inetd (most TCP/IP applications, xinetd in Red Hat Linux 9)
  - httpd (web service)
  - named (DNS service)
- Port numbers
  - Well-known port numbers, used by servers
  - Dynamic/private port numbers (per RFC 6335), used by clients (never assigned)
  - The Protocol Type, the IP address and port number pairs of the server and client preserve the uniqueness of a communication session → IP Five–Tuple information

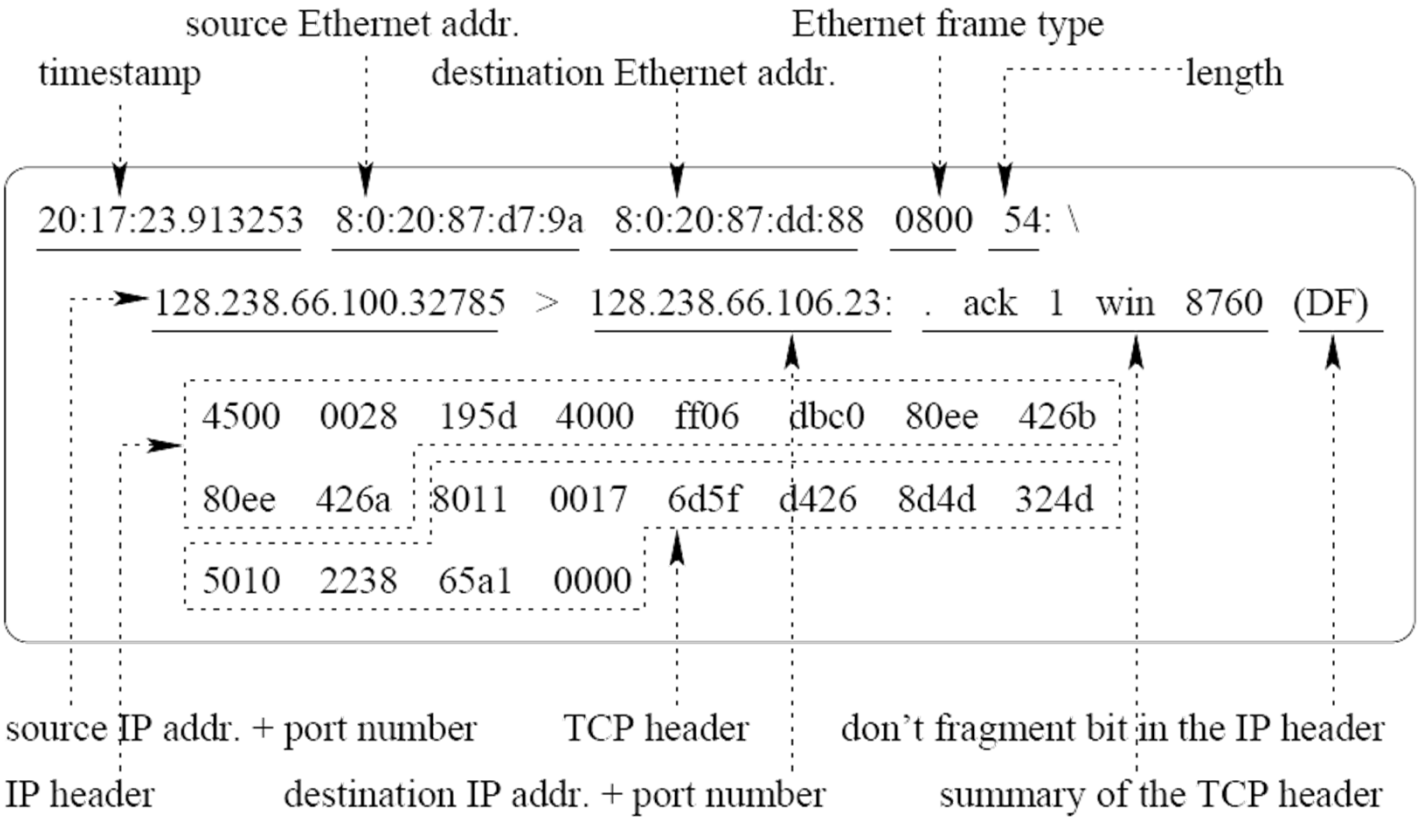
Network Configurations Files

- When a host is configured to boot locally, TCP/IP configuration parameters are stored in files.
  - /etc/services (well-known port numbers, a.k.a. System Port numbers)
  - /etc/inetd.conf (inetd daemon parameters)
  - /etc/sysconfig/network (host name, default gateway IP address)
  - /etc/resolv.conf (IP addresses of DNS servers)
- When the system boots up, parameters are read from the files and used to configure the daemons and the network interface.
- A parameter may be changed by editing the corresponding configuration file.
- *Most user hosts today are configured remotely by DHCP server (dynamic host configuration protocol) with distributed network configuration parameters*

Linux Commands and Tools

- Basic Linux commands: man, passwd, ls, … many more
- Text editor
  - vi
  - Other text editors: Emacs, gedit, OpenOffice.org
- Window Dump using PrintScreen key
- Using USB memory stick, …to collect lab data for reports

Diagnostic Tools



Tcpdump – a network traffic sniffer

Ethereal – a network protocol analyzer

More about Domain Name System

DNS: domain name system

People: many identifiers:

- SSN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams
- "name", e.g., www.nyu.edu - 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"

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

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

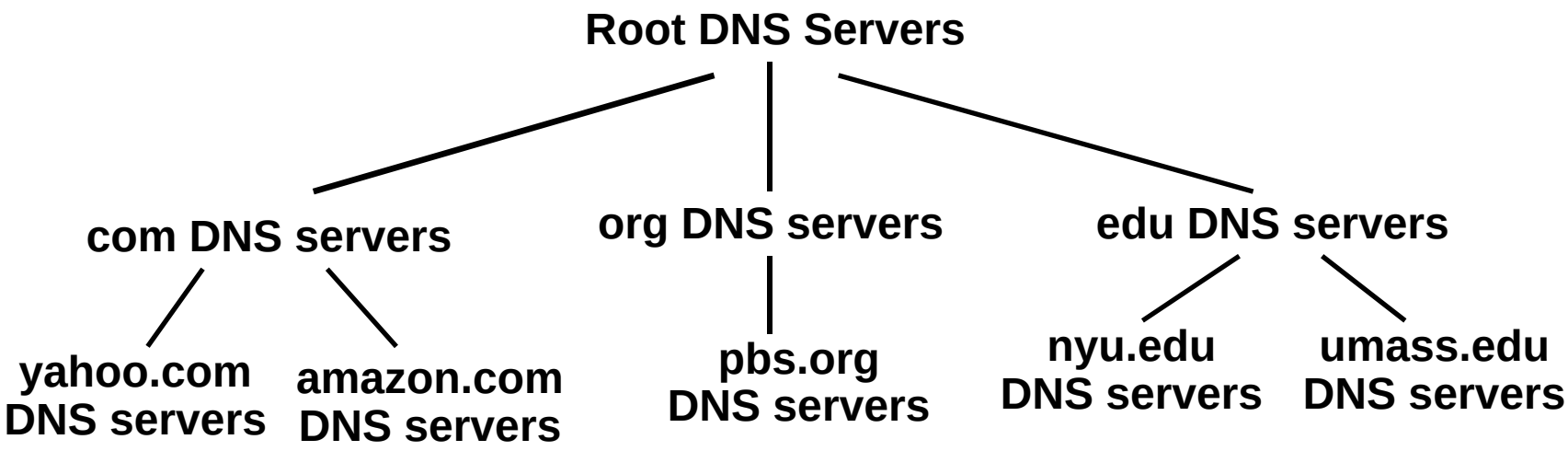
distant centralized database

maintenance

A: ***It doesn't scale!***

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

DNS: a distributed, hierarchical database



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

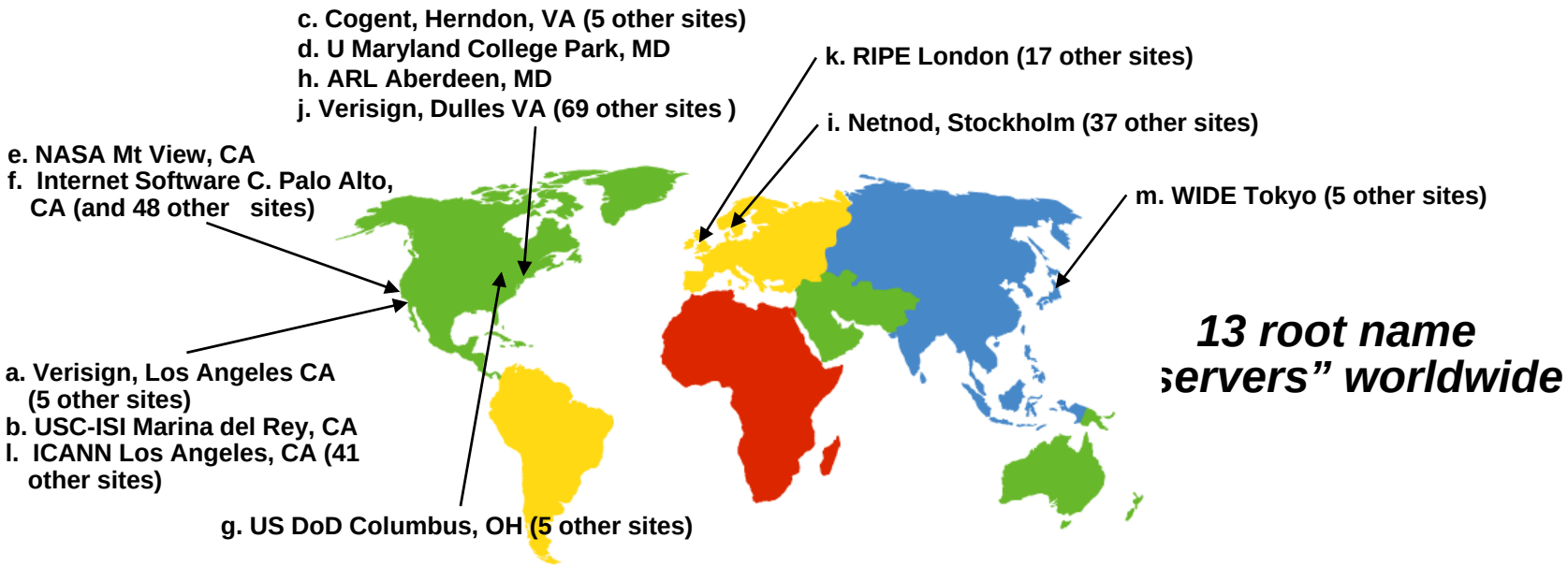
- client queries root server to find .com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

DNS: root name servers

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

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

TLD servers, 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
- Verisign Global Registry 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

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

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

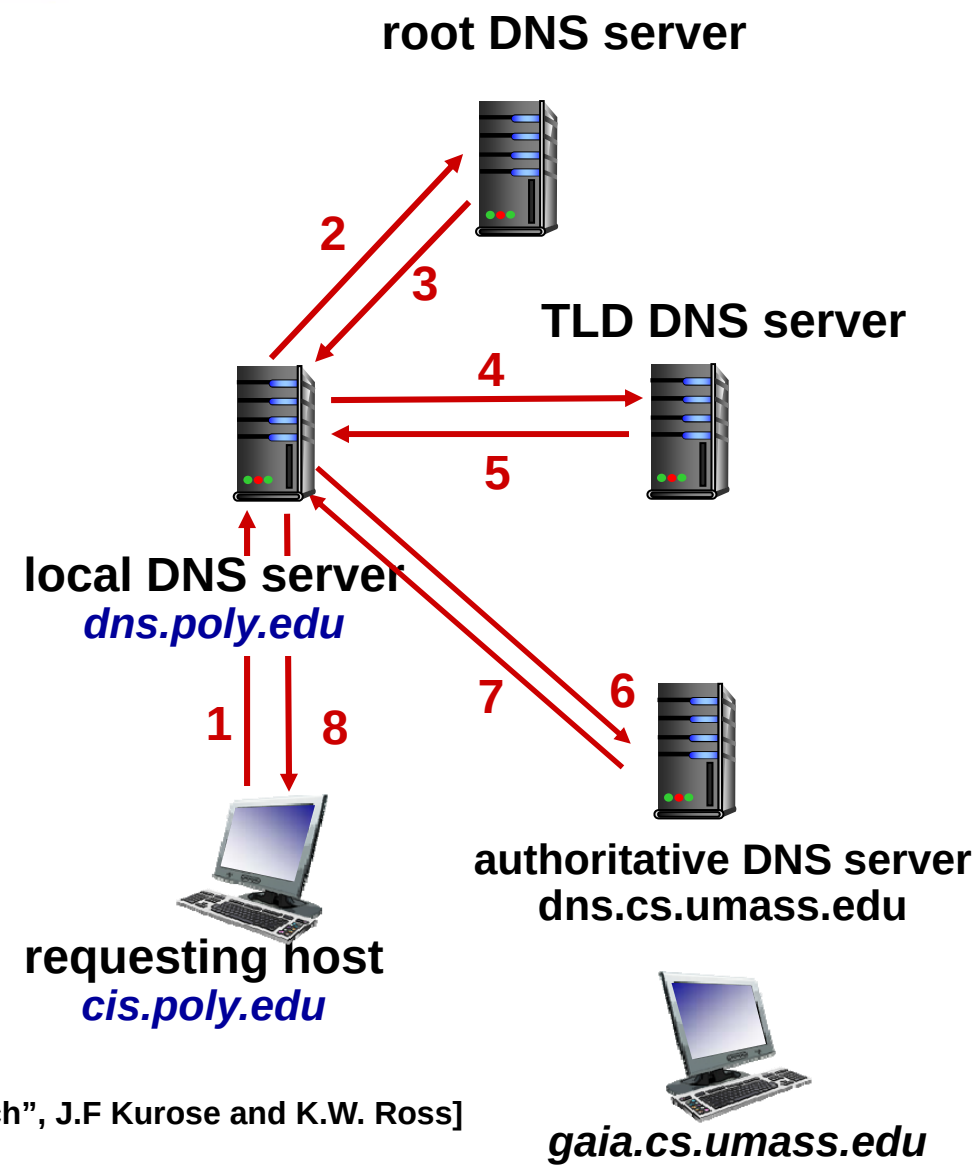
[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

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"**

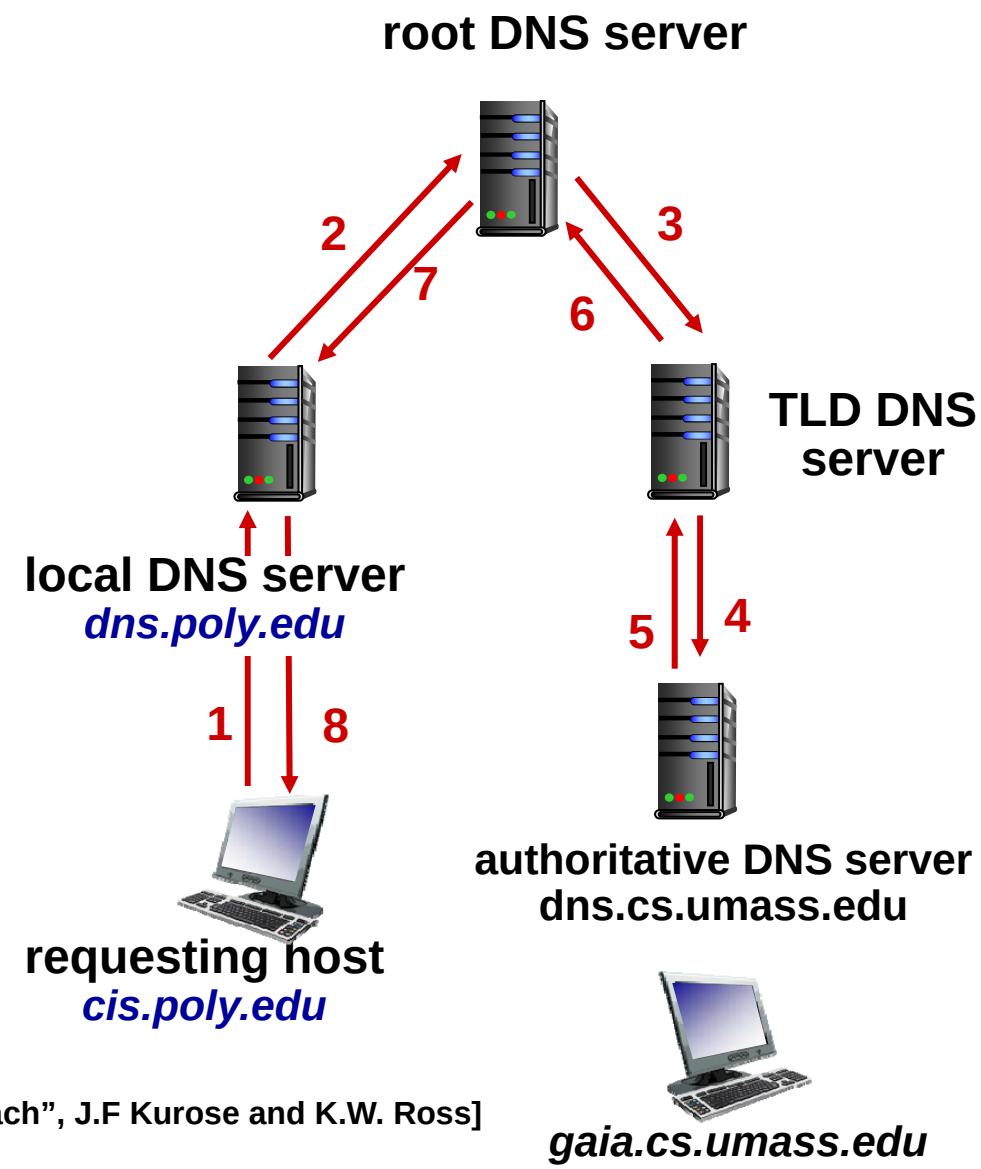


[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

DNS name resolution example

Recursive query:

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



[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]

DNS: caching, updating records

Once (any) name server learns mapping, it *caches* 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 by IETF

- RFC 2136

[Source: "Computing Networking: A Top Down Approach", J.F Kurose and K.W. Ross]