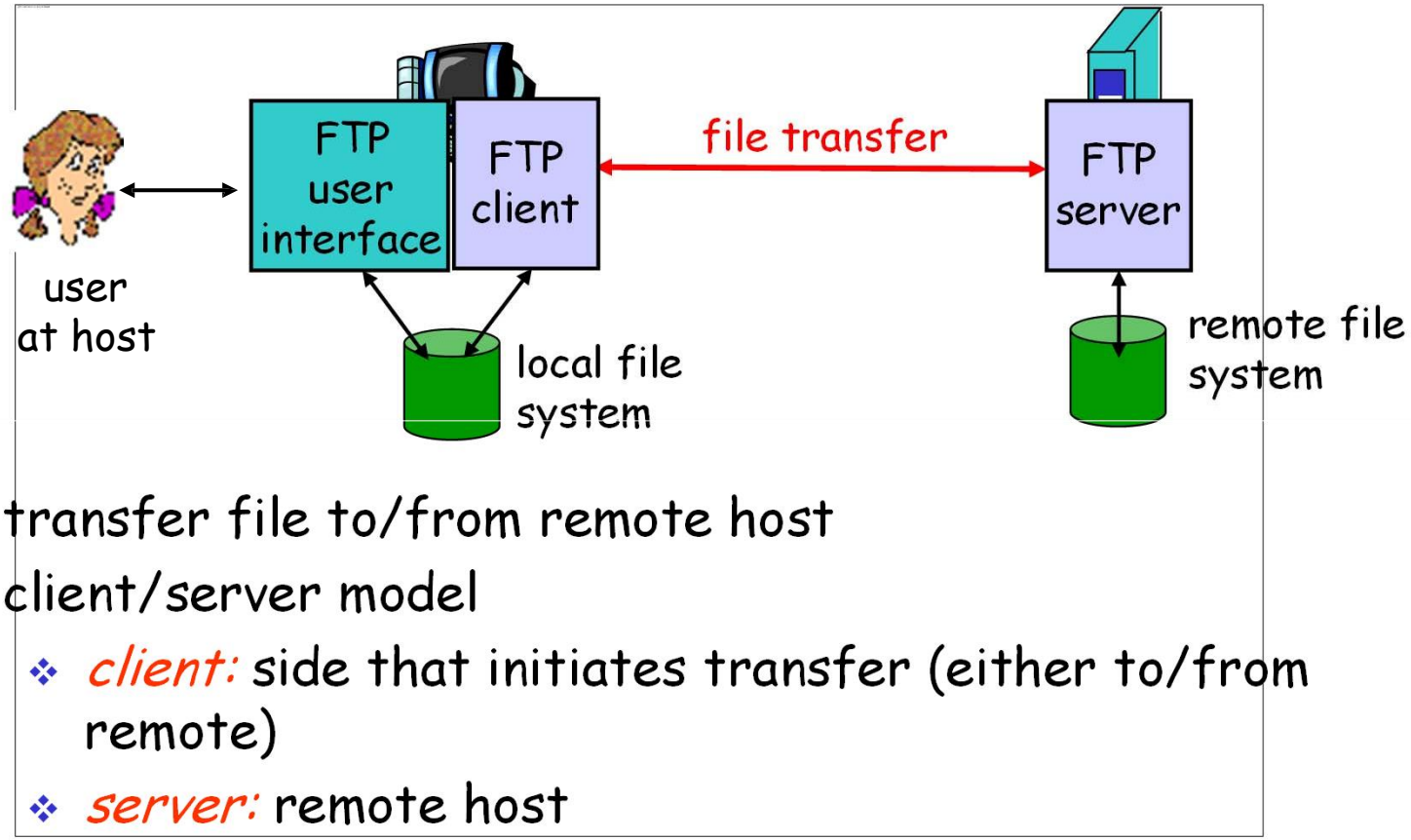
Network Applications and Transport Services

Outline

- Context/overview
- Network application design principles
- Applications and protocols (Application Layer)
  - Web and HTTP
  - FTP
  - Electronic Mail
  - DNS
- Transporting application messages (Transport Layer)

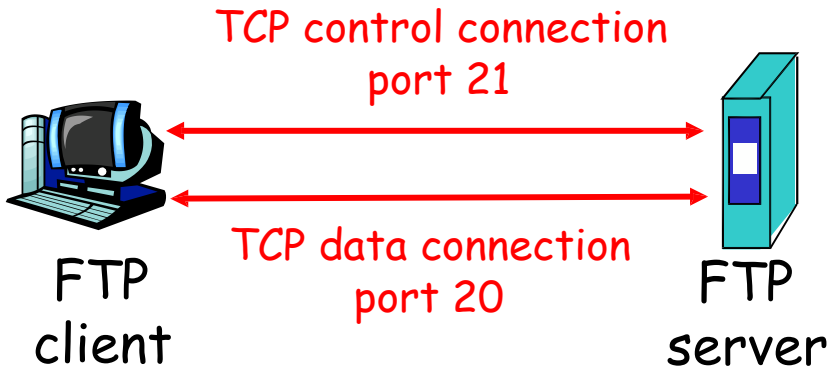
FTP: the file transfer protocol



- transfer file to/from remote host
- client/server model
  - *client:* side that initiates transfer (either to/from remote)
  - *server:* remote host
- ftp: RFC 959
- ftp server: port 21

FTP: separate control, data connections

- FTP client contacts FTP server at port 21, TCP is transport protocol
- client authorized over control connection
- client browses remote directory by sending commands over control connection.
- when server receives file transfer command, server opens 2nd TCP connection (for file) to client
- after transferring one file, server closes data connection.



- server opens another TCP data connection to transfer another file.
- control connection: "out of band"
- FTP server maintains "state": current directory, earlier authentication

FTP commands, responses

Sample commands:

- sent as ASCII text over control channel
- **USER *username***
- **PASS *password***
- **LIST** return list of file in current directory
- **RETR filename** retrieves (gets) file
- **STOR filename** stores (puts) file onto remote host

Sample return codes

- status code and phrase (as in HTTP)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

Network Applications and Transport Services
Outline

- Context/overview
- Network application design principles
- Applications and protocols (Application Layer)
  - Web and HTTP
  - FTP
  - Electronic Mail
  - DNS
- Transporting application messages (Transport Layer)

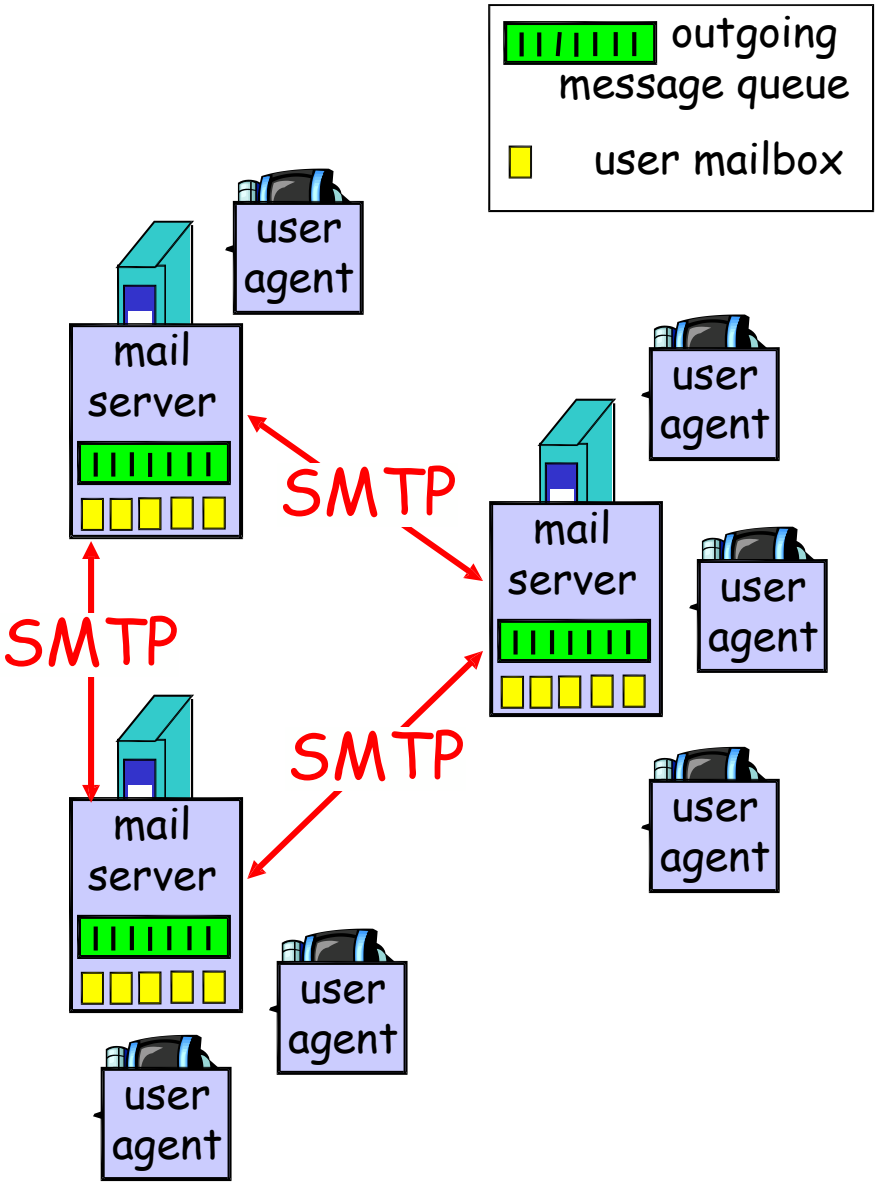
Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent

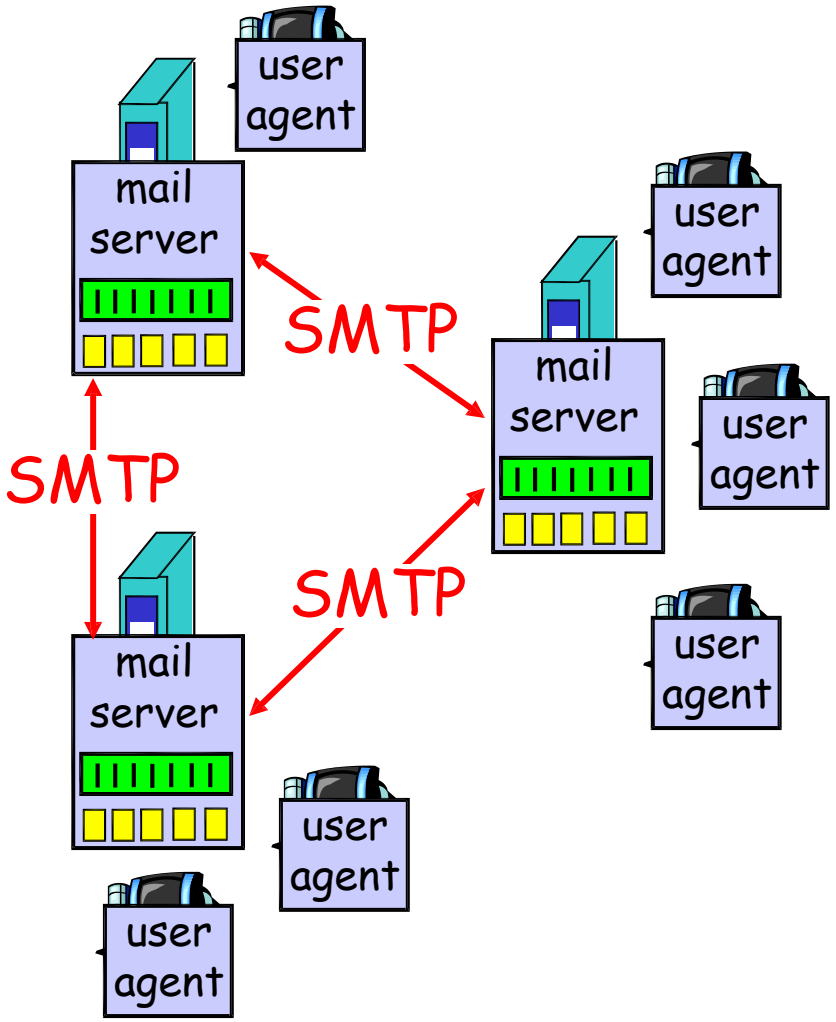
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Outlook, Thunderbird, iPhone mail client
- outgoing, incoming messages stored on server



Electronic Mail: mail servers

Mail Servers

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server

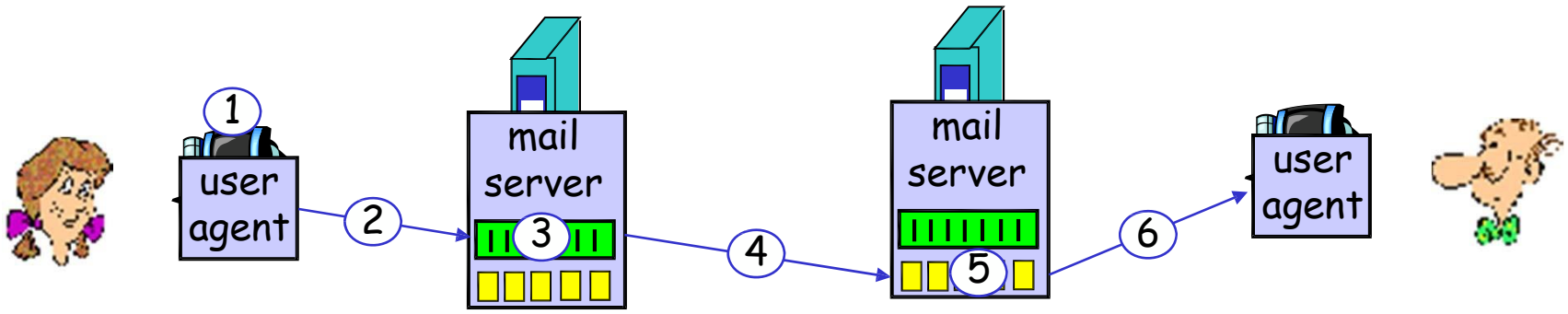


Electronic Mail: SMTP [RFC 2821]

- uses TCP to reliably transfer email message from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
- messages must be in 7-bit ASCII

Scenario: Alice sends message to Bob

1) Alice uses UA to compose message and "to" bob@someschool.edu
2) Alice's UA sends message to her mail server; message placed in message queue
3) Client side of SMTP opens TCP connection with Bob's mail server
4) SMTP client sends Alice's message over the TCP connection
5) Bob's mail server places the message in Bob's mailbox
6) Bob invokes his user agent to read message



Sample SMTP interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Try SMTP interaction for yourself:

- **`telnet servername 25`**
- see 220 reply from server
- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands

above lets you send email without using email client (reader)

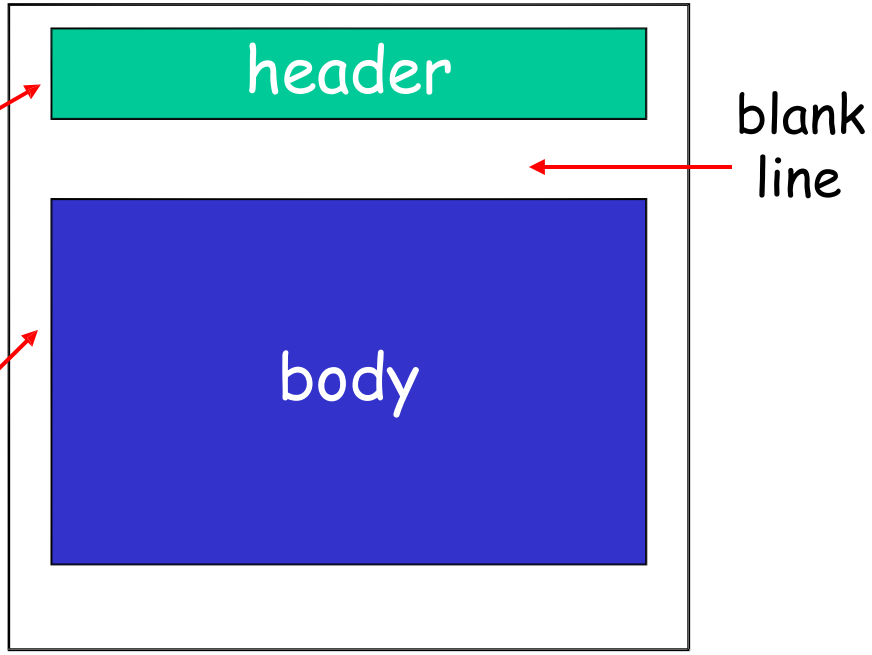
Mail message format

SMTP: protocol for exchanging email msgs

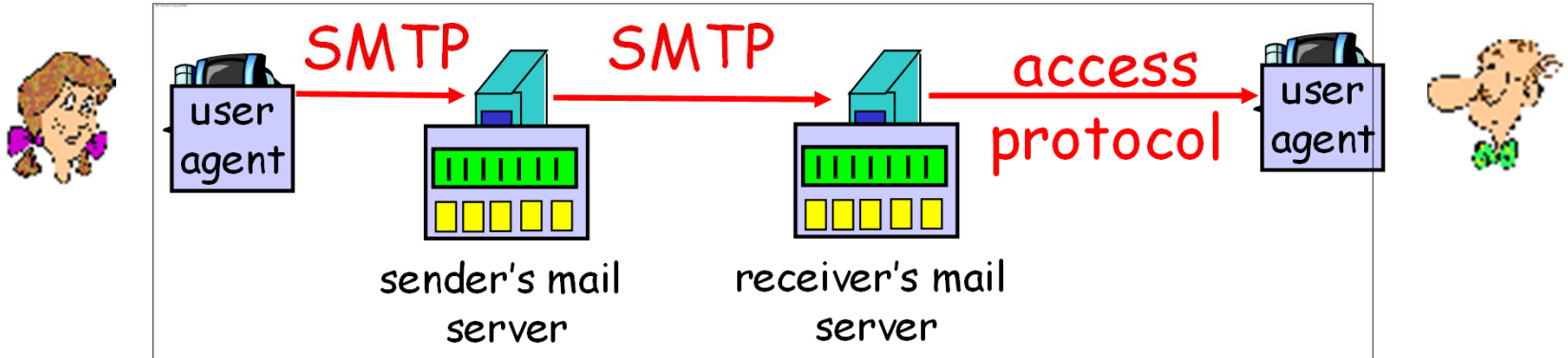
RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject:

  different from SMTP commands!
- body
  - the "message", ASCII characters only



Mail access protocols



- SMTP: delivery/storage to receiver's server
- Mail access protocol: retrieval from server
  - POP: Post Office Protocol [RFC 1939]
    - authorization (agent <-->server) and download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - more features (more complex)
    - manipulation of stored msgs on server
  - HTTP: gmail, Hotmail, Yahoo! Mail, etc.

POP3 protocol

authorization phase

- client commands:
  - `user:` declare username
  - `pass:` password
- server responses
  - `+OK`
  - `-ERR`

transaction phase, client:

- `list:` list message numbers
- `retr:` retrieve message by number
- `dele:` delete
- `quit`

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 (more) and IMAP

More about POP3

- Previous example uses "download and delete" mode.
- Bob cannot re-read e-mail if he changes client
- "Download-and-keep": copies of messages on different clients
- POP3 is stateless across sessions

IMAP

- Keep all messages in one place: the server
- Allows user to organize messages in folders
- IMAP keeps user state across sessions:
  - names of folders and mappings between message IDs and folder name

Network Applications and Transport Services
Outline

- Context/overview
- Network application design principles
- Applications and protocols (Application Layer)
  - Web and HTTP
  - FTP
  - Electronic Mail
  - DNS
- Transporting application messages (Transport Layer)

DNS: Domain Name System

People: many identifiers:

- name, ID#, passport #

To identify Internet 'hosts':
(hosts/end systems- run/host applications)

- IP address
  - used for addressing datagrams.
  - Routers prefer IP addresses (fixed-length, hierarchically structured)
- "name" or "hostname",
  e.g.,
  cnn.com
  ww.yahoo.com
  (used by humans)

map between name and IP address?

Domain Name System:

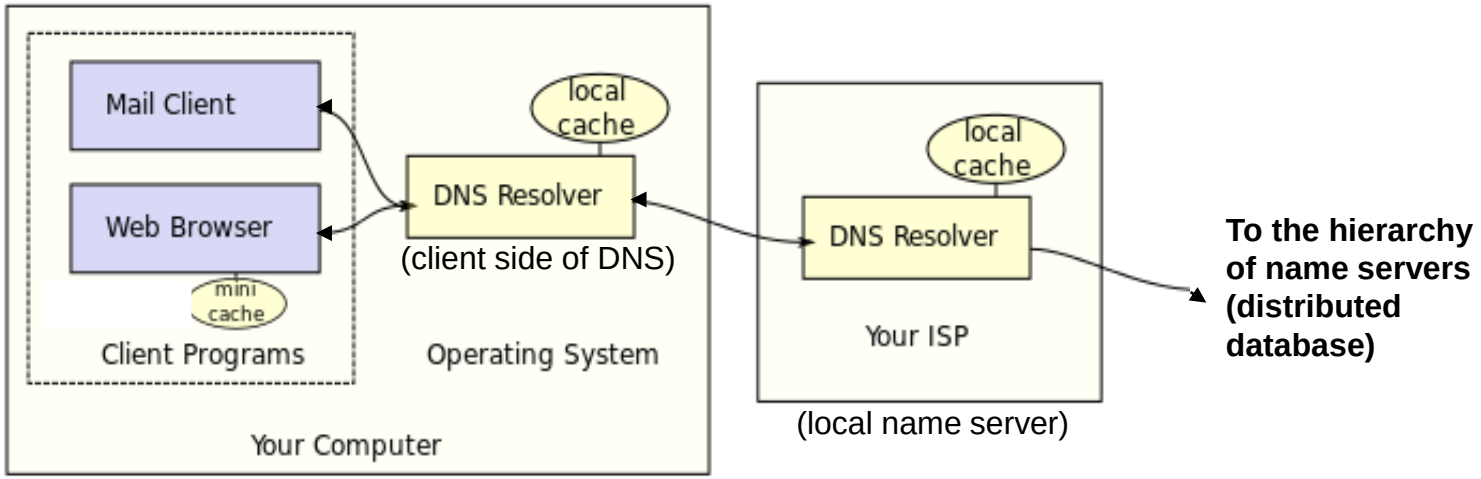
- distributed database implemented in hierarchy of many name servers
- application-layer protocol to resolve names (address/name translation)
  - note: core Internet function, implemented as application-layer protocol
  - complexity at network's "edge"
  - The DNS protocol runs over UDP and uses port 53.

  why UDP?

How do you (your computer) use DNS?

- DNS is commonly used by other application-layer protocols—including HTTP, SMTP, and FTP—to translate user-supplied hostnames to IP addresses.

- E.g., a browser in your machine requests URL 'www.someschool.edu/index.html'
  - Your machine runs the client side of the DNS application.
  - The browser extracts the hostname, www.someschool.edu, from the URL and passes the hostname to the client side of the DNS application.
  - The DNS client sends a query containing the hostname to a DNS server (typically, to the 'local DNS server/local name server' in your institution).
  - The DNS client eventually receives a reply, which includes the IP address for the hostname.
  - Once the browser receives the IP address from DNS, it can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.



DNS

DNS services

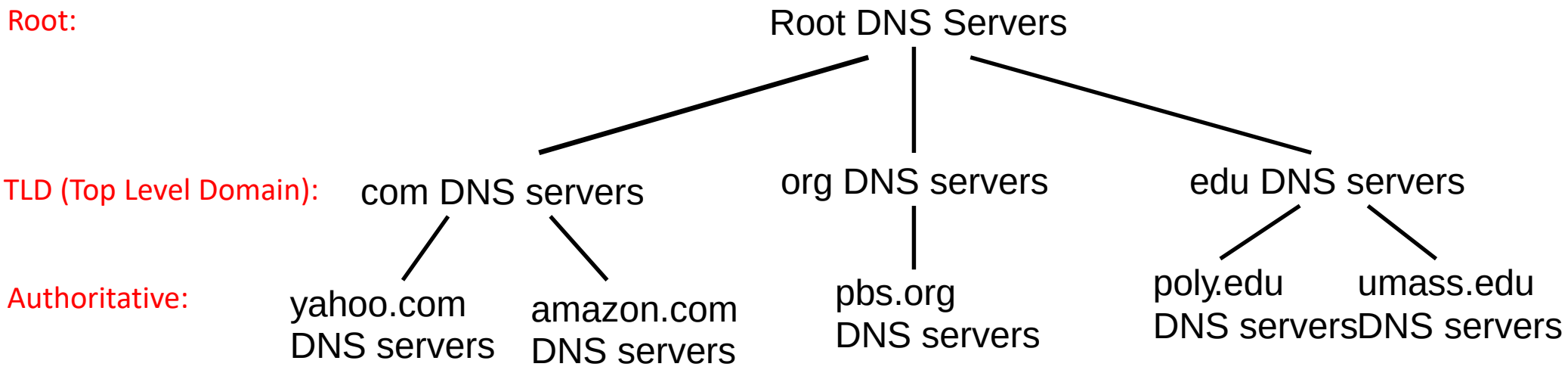
- hostname to IP address translation
- host aliasing
  - Canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: set of IP addresses for one canonical name

Why not centralize DNS?

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

doesn't scale!

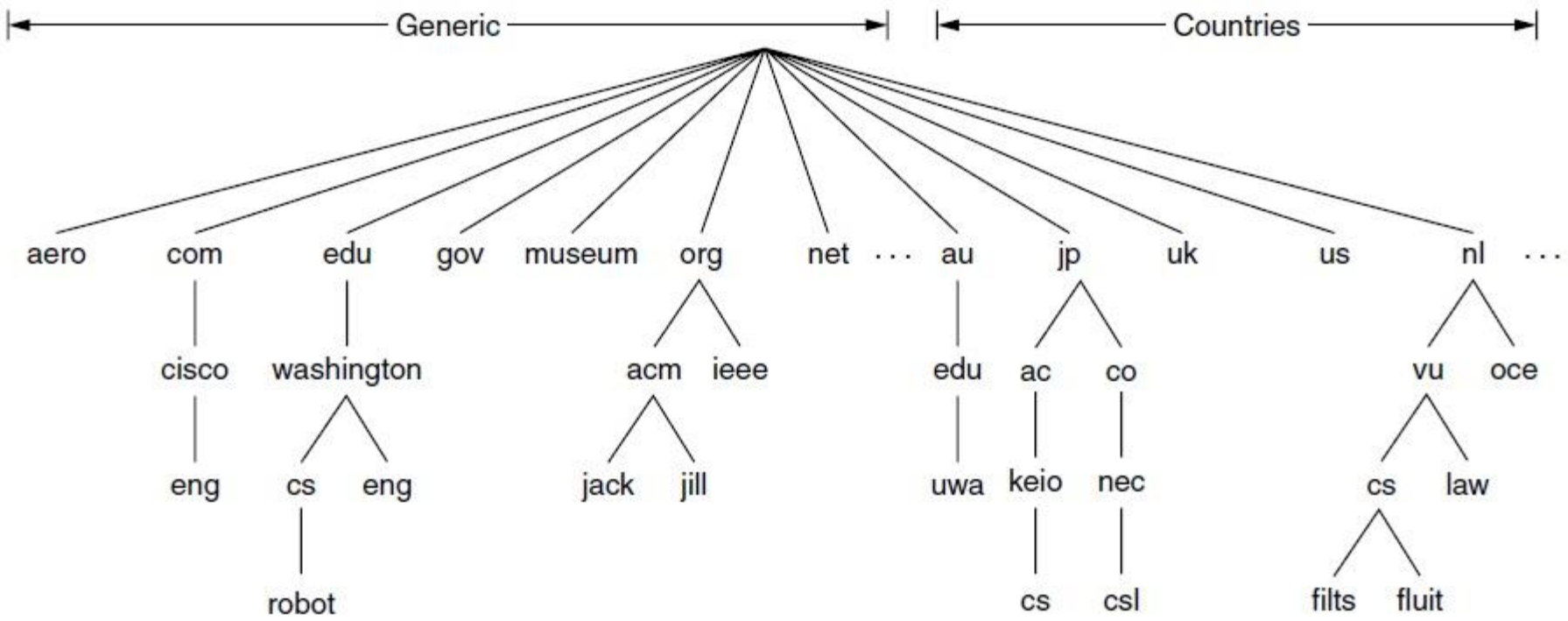
Distributed, Hierarchical Database



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

- client queries a 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

* Local name server: typically, DNS query is sent into the hierarchy via 'local name server'

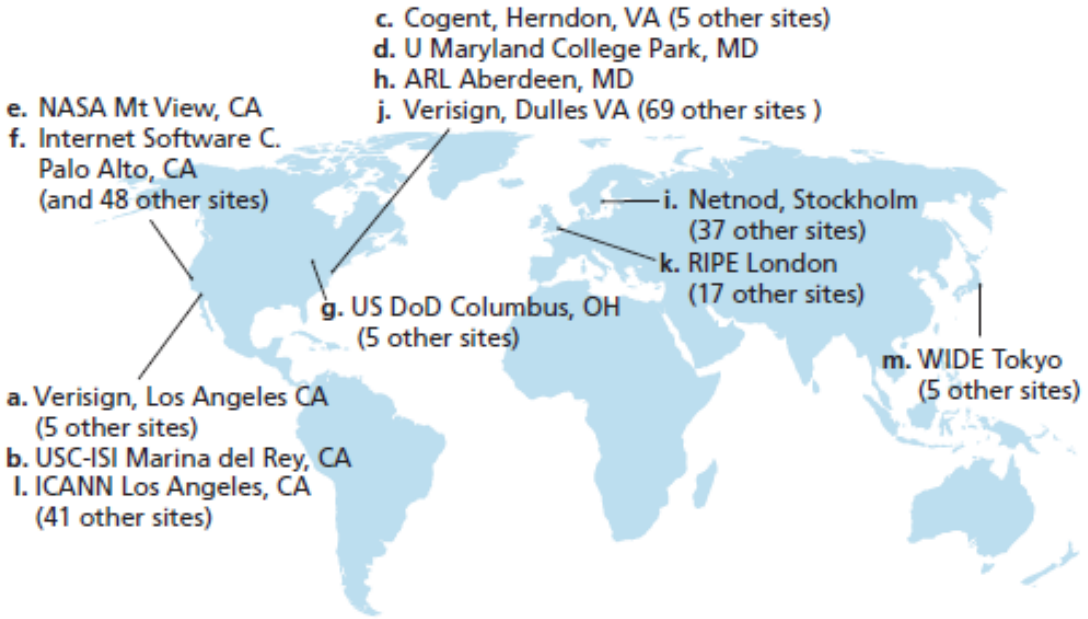


- For the Internet, the top of the naming hierarchy is managed the **ICANN (Internet Corporation for Assigned Names and Numbers)**.

As of February 2017, the root domain contains 1528 top-level domains.
(e.g., .com .lk .art .android .apple .apartments .baby)

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 logical root name "servers" worldwide- operated by different organizations (each "server" replicated many times)

The use of 'anycast' addressing permits the actual number of root server instances to be much larger: 632 as of 25 October 2016.

TLD 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.: uk, fr, ca, jp
  - Verisign maintains servers for .com TLD. Educause for .edu TLD
- Authoritative DNS servers:
  - organization's DNS servers, providing authoritative hostname to IP mappings for organization's servers (e.g., Web, mail).
  - can be maintained by organization or service provider

Local Name Server

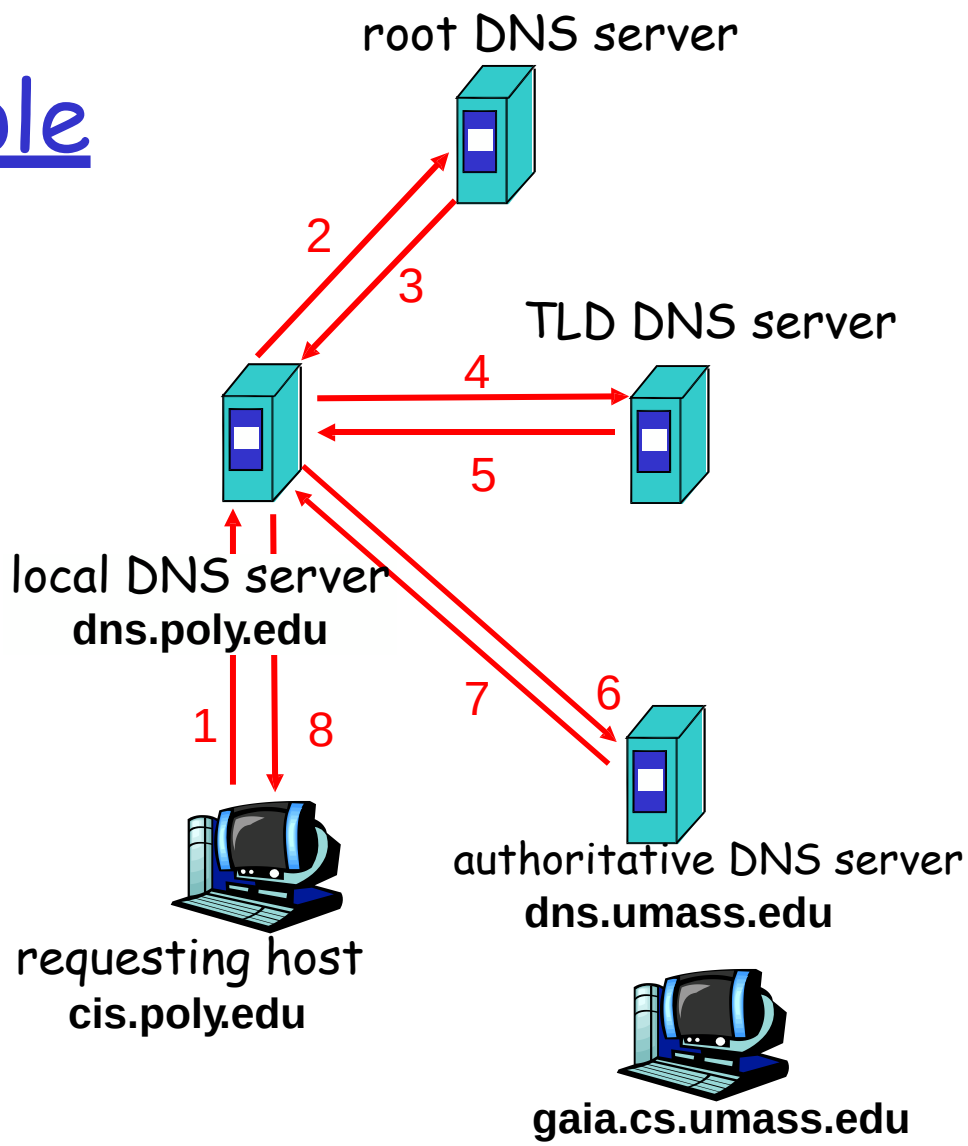
- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one.
  - also called "default name server"
- when host makes DNS query, query is sent to its local DNS server
  - acts as proxy, forwards query into hierarchy

DNS name resolution example

- Host at cis.poly.edu wants IP address for gaia.cs.umass.edu

iterated query:

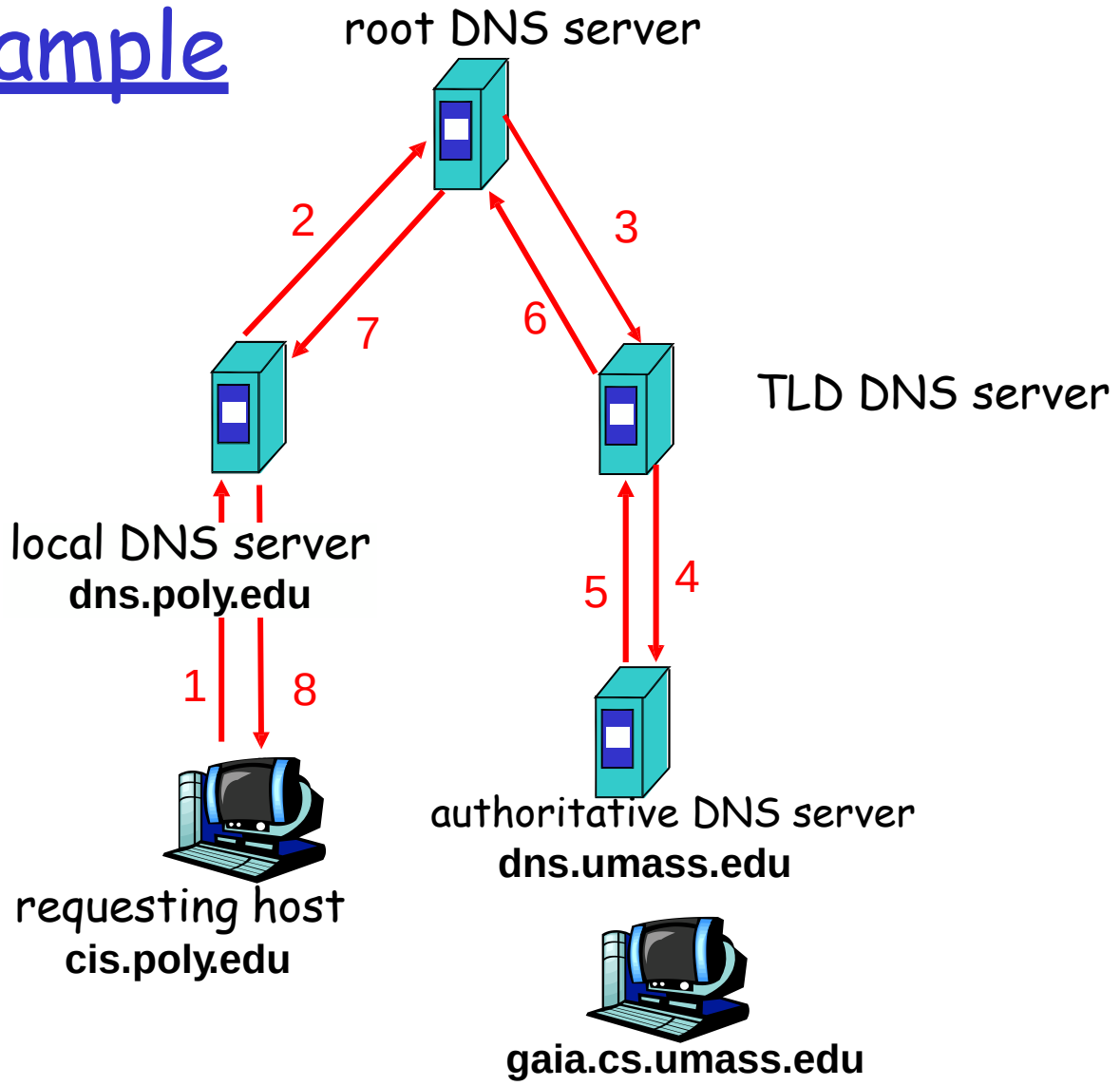
- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



DNS name resolution example

recursive query:

- puts burden of name resolution on contacted name server
- heavy load?



DNS: caching and updating records

- once (any) name server learns mapping, it caches mapping
  - cache entries timeout (disappear) after some time
  - TLD servers typically cached in local name servers
    - Thus root name servers not often visited
- update/notify mechanisms under design by IETF
  - RFC 2136
  - http://www.ietf.org/html.charters/dnsind-charter.html

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**

Type	Meaning	Value
SOA	Start of authority	Parameters for this zone
A	IPv4 address of a host	32-Bit integer
AAAA	IPv6 address of a host	128-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
SPF	Sender policy framework	Text encoding of mail sending policy
SRV	Service	Host that provides it
TXT	Text	Descriptive ASCII text

SRV: allows a host to be identified for a given service in a domain. This record generalizes the *MX* record that performs the same task for mail servers.

A portion of a sample DNS database:

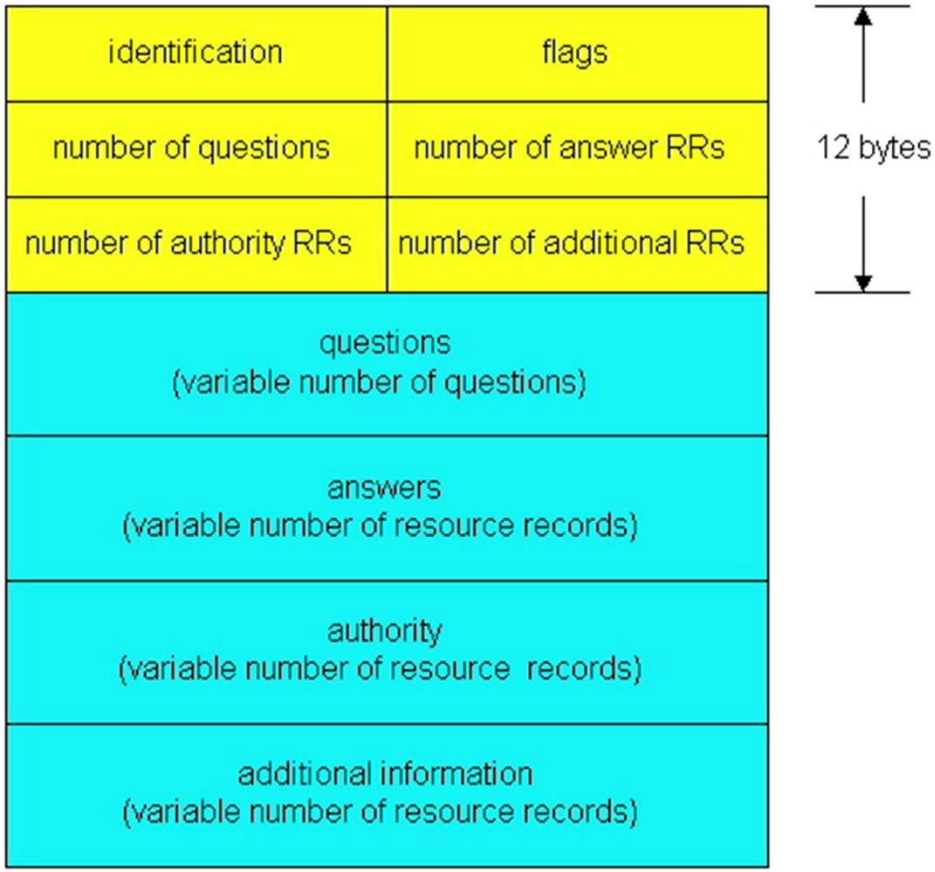
```
$ORIGIN example.com.     ; designates the start of this zone file in the namespace
$TTL 1h                  ; default expiration time of all resource records without their own TTL value
example.com.  IN  SOA   ns.example.com. username.example.com. ( 2007120710 1d 2h 4w 1h )
example.com.  IN  NS    ns                    ; ns.example.com is a nameserver for example.com
example.com.  IN  NS    ns.somewhere.example. ; ns.somewhere.example is a backup nameserver for example.co
example.com.  IN  MX    10 mail.example.com.  ; mail.example.com is the mailserver for example.com
@             IN  MX    20 mail2.example.com. ; equivalent to above line, "@" represents zone origin
@             IN  MX    50 mail3              ; equivalent to above line, but using a relative host name
example.com.  IN  A     192.0.2.1             ; IPv4 address for example.com
              IN  AAAA  2001:db8:10::1        ; IPv6 address for example.com
ns            IN  A     192.0.2.2             ; IPv4 address for ns.example.com
              IN  AAAA  2001:db8:10::2        ; IPv6 address for ns.example.com
www           IN  CNAME example.com.          ; www.example.com is an alias for example.com
wwwtest       IN  CNAME www                   ; wwwtest.example.com is another alias for www.example.com
mail          IN  A     192.0.2.3             ; IPv4 address for mail.example.com
mail2         IN  A     192.0.2.4             ; IPv4 address for mail2.example.com
mail3         IN  A     192.0.2.5             ; IPv4 address for mail3.example.com
```

DNS protocol, messages

DNS protocol : 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



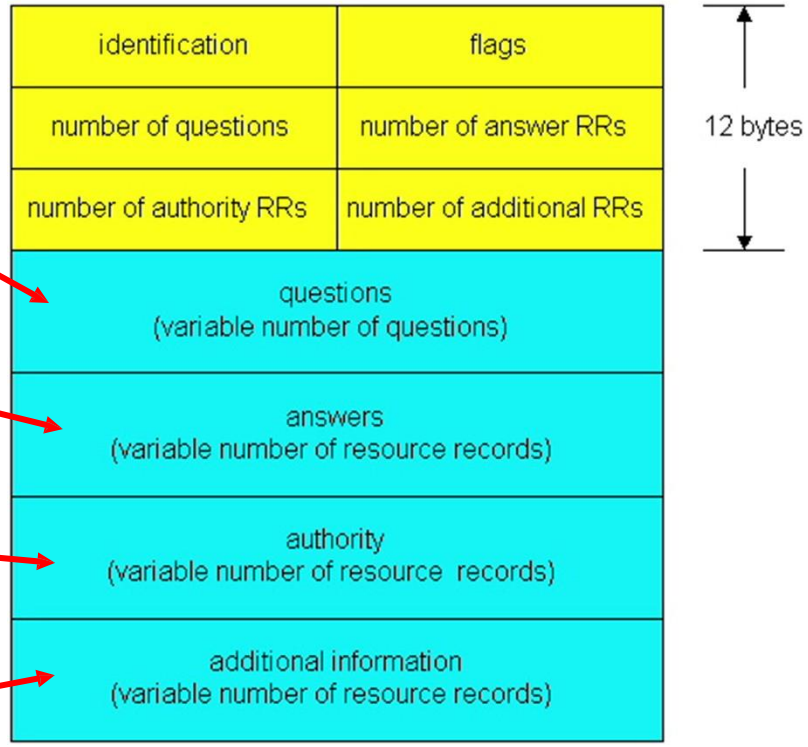
DNS protocol, messages

Name, type fields for a query

RRs in response to query

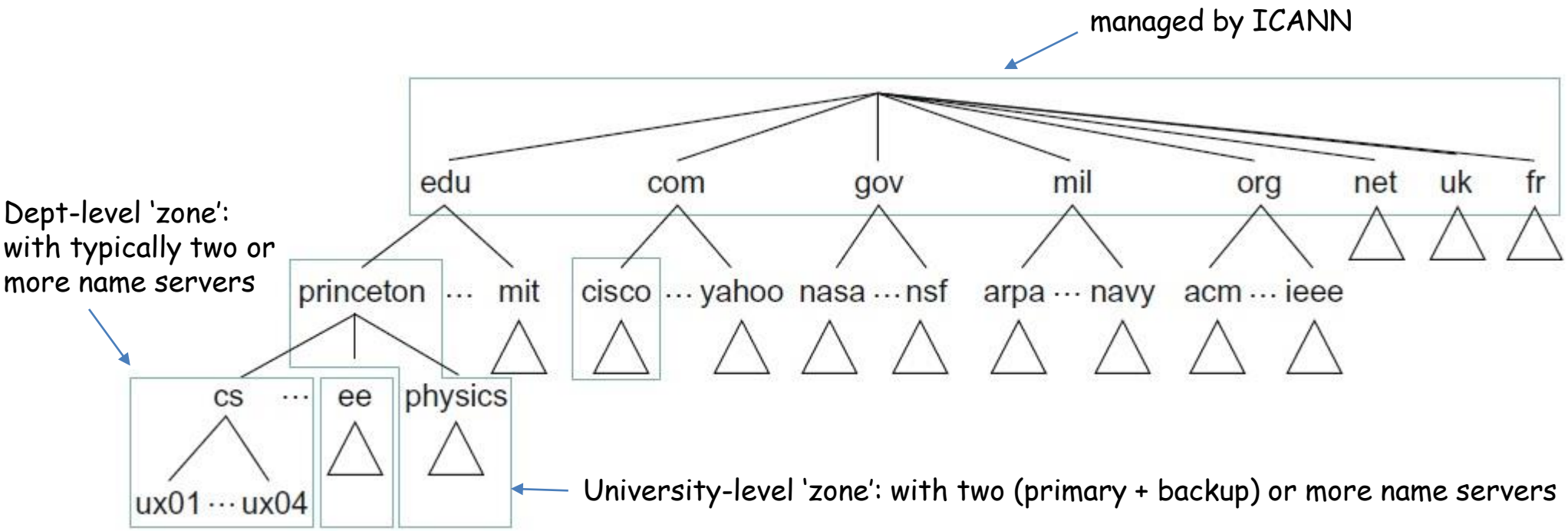
records for authoritative servers

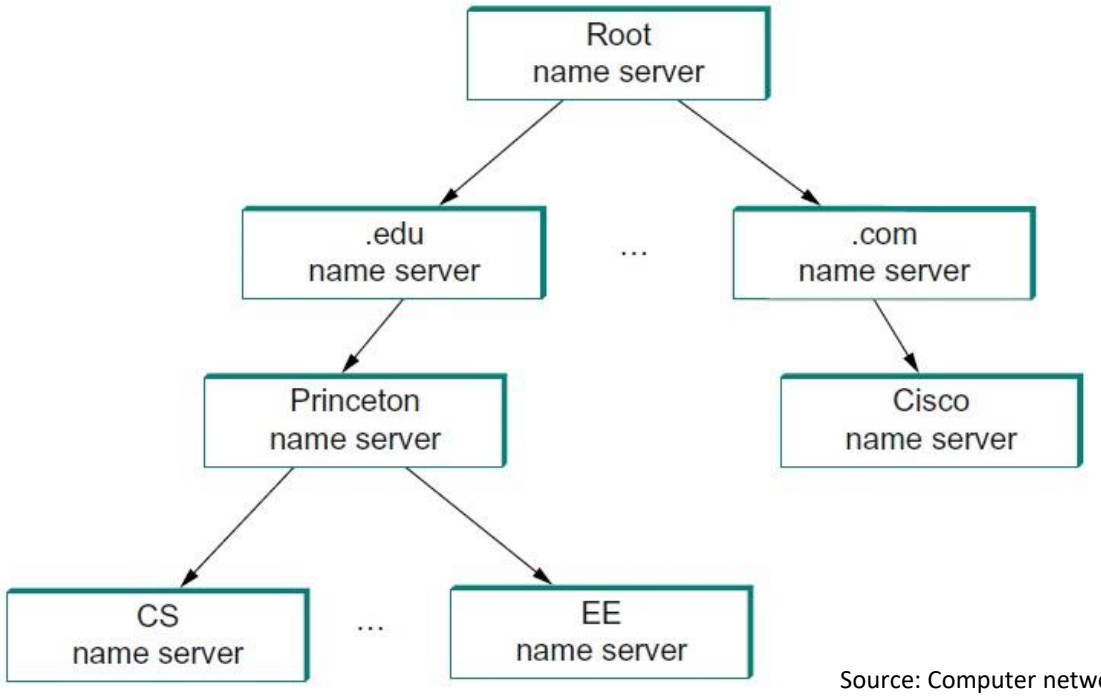
additional "helpful" info that may be used



Example

Domain hierarchy partitioned into 'zones'





Hierarchy of name servers

Source: Computer networks a systems approach- Larry L. Peterson and Bruce S. Davie, 2012

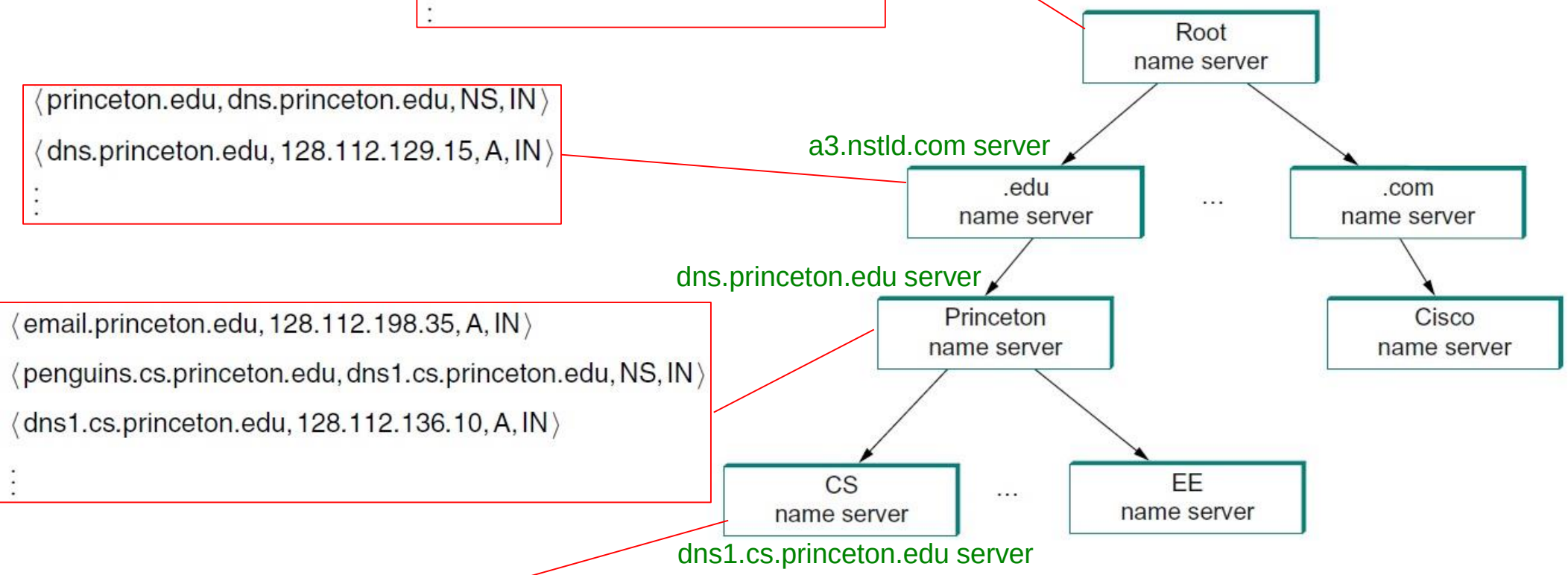
⟨edu, a3.nstld.com, NS, IN⟩

⟨a3.nstld.com, 192.5.6.32, A, IN⟩

⟨com, a.gtld-servers.net, NS, IN⟩

⟨a.gtld-servers.net, 192.5.6.30, A, IN⟩

⋮



⟨princeton.edu, dns.princeton.edu, NS, IN⟩

⟨dns.princeton.edu, 128.112.129.15, A, IN⟩

⋮

⟨email.princeton.edu, 128.112.198.35, A, IN⟩

⟨penguins.cs.princeton.edu, dns1.cs.princeton.edu, NS, IN⟩

⟨dns1.cs.princeton.edu, 128.112.136.10, A, IN⟩

⋮

⟨penguins.cs.princeton.edu, 128.112.155.166, A, IN⟩

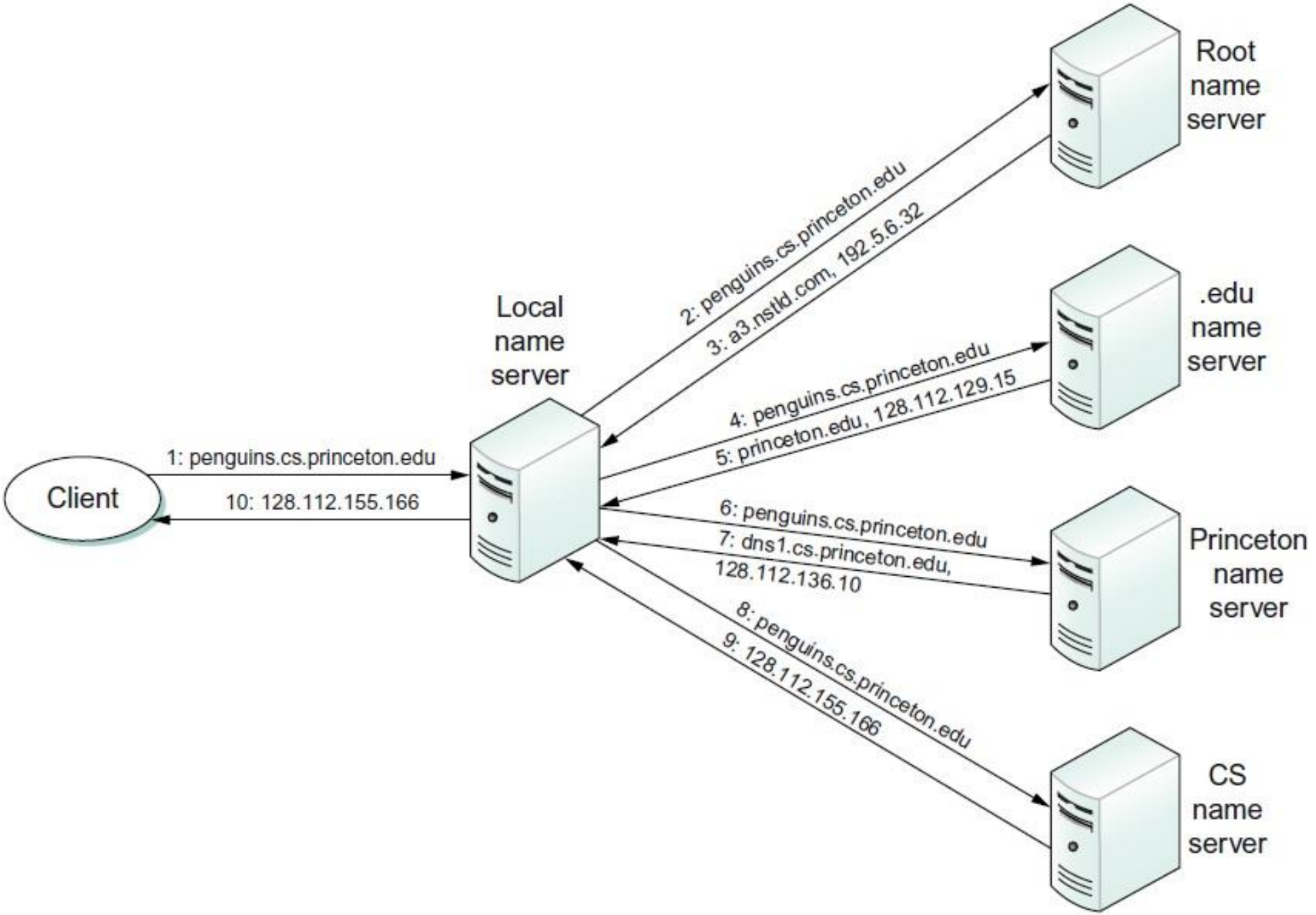
⟨www.cs.princeton.edu, coreweb.cs.princeton.edu, CNAME, IN⟩

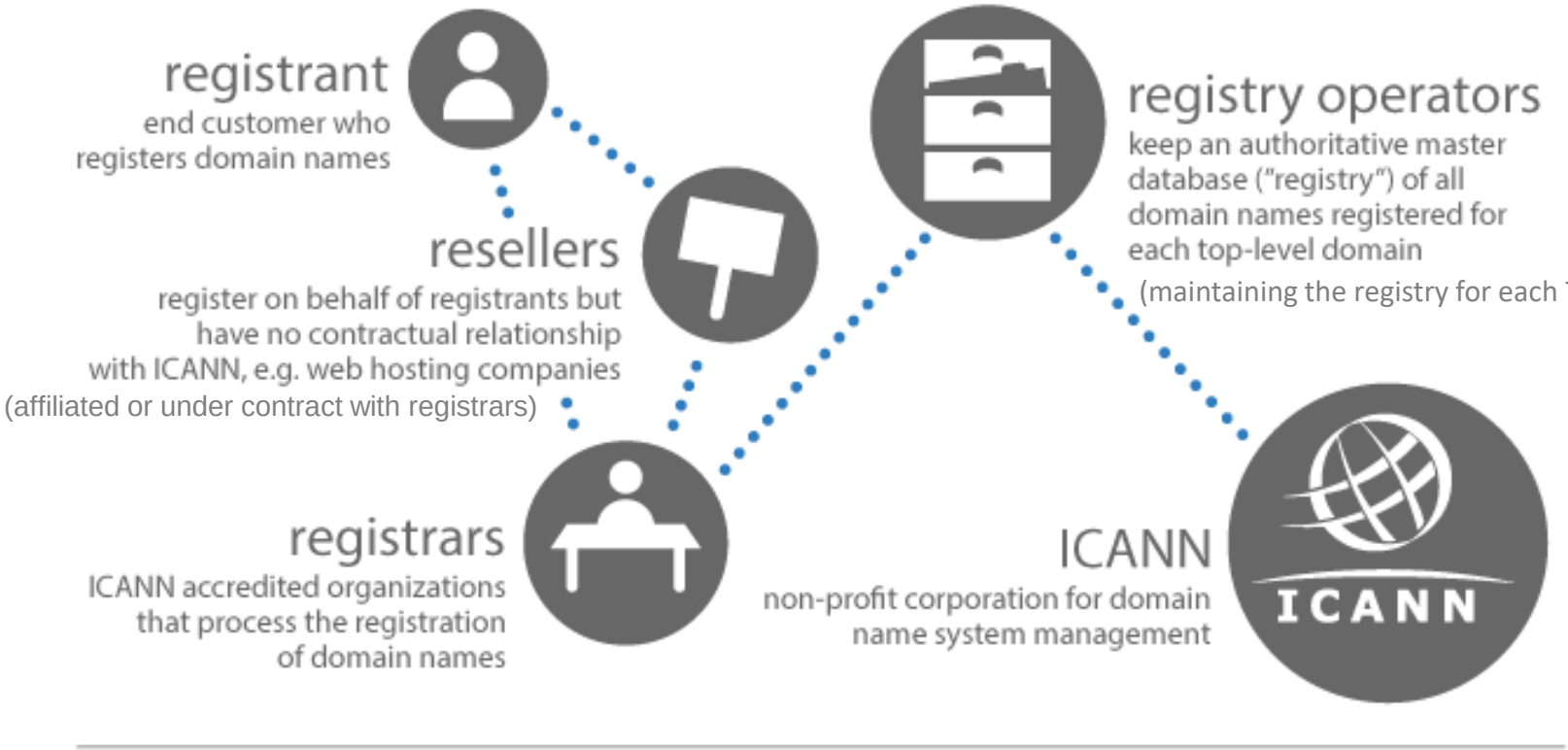
⟨coreweb.cs.princeton.edu, 128.112.136.35, A, IN⟩

⟨cs.princeton.edu, mail.cs.princeton.edu, MX, IN⟩

⟨mail.cs.princeton.edu, 128.112.136.72, A, IN⟩

⋮


Root
name
server
.edu
name
server
Princeton
name
server
CS
name
server
Local
name
server
Client
1: penguins.cs.princeton.edu
10: 128.112.155.166
2: penguins.cs.princeton.edu
3: a3.nstld.com, 192.5.6.32
4: penguins.cs.princeton.edu
5: princeton.edu, 128.112.129.15
6: penguins.cs.princeton.edu
7: dns1.cs.princeton.edu,
128.112.136.10
8: penguins.cs.princeton.edu
9: 128.112.155.166



domain registry process

Source: ICANN