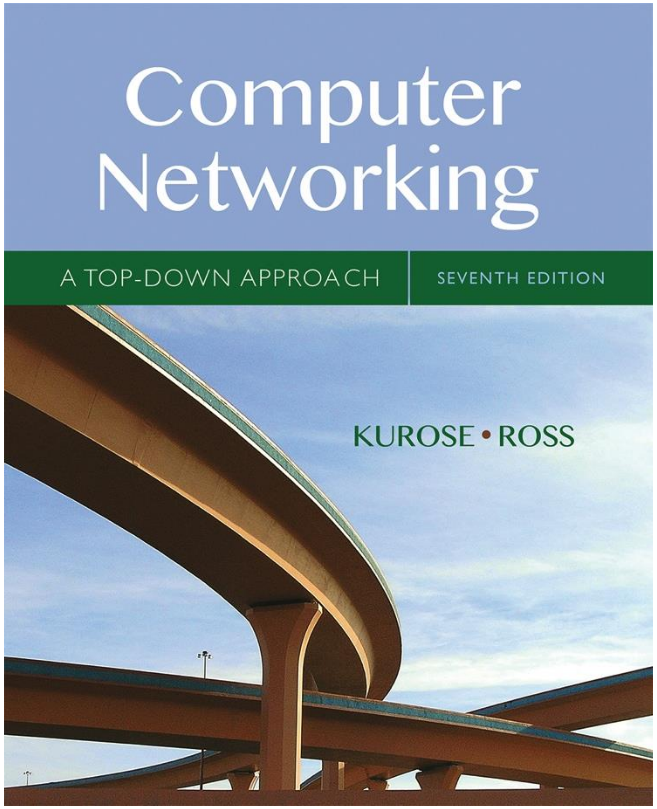
Chapter 2
Application Layer



A note on the use of these Powerpoint slides:

We' re making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we' d like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Computer Networking: A Top Down Approach

7th edition
Jim Kurose, Keith Ross
Pearson/Addison Wesley
April 2016

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

2.7

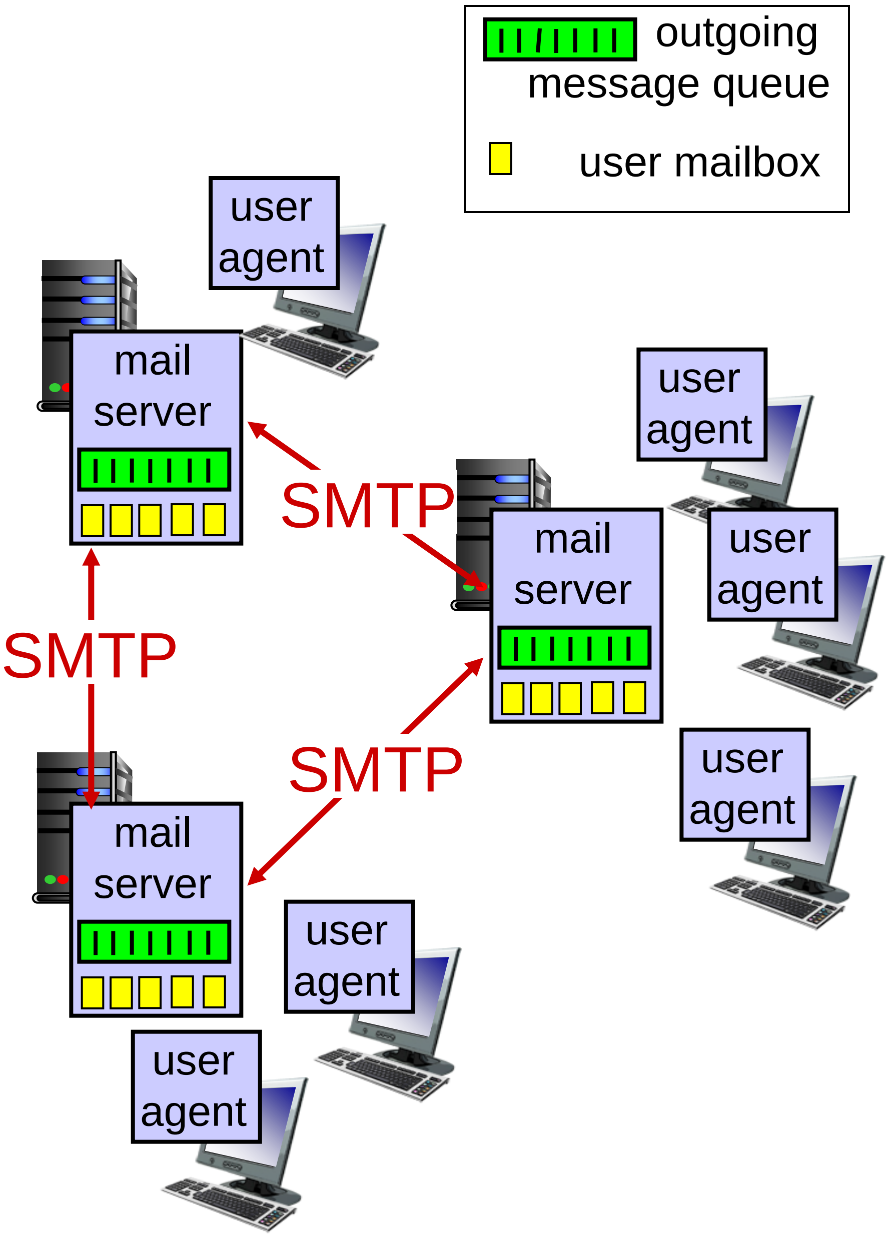
Electronic mail

Three major components:

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

User Agent

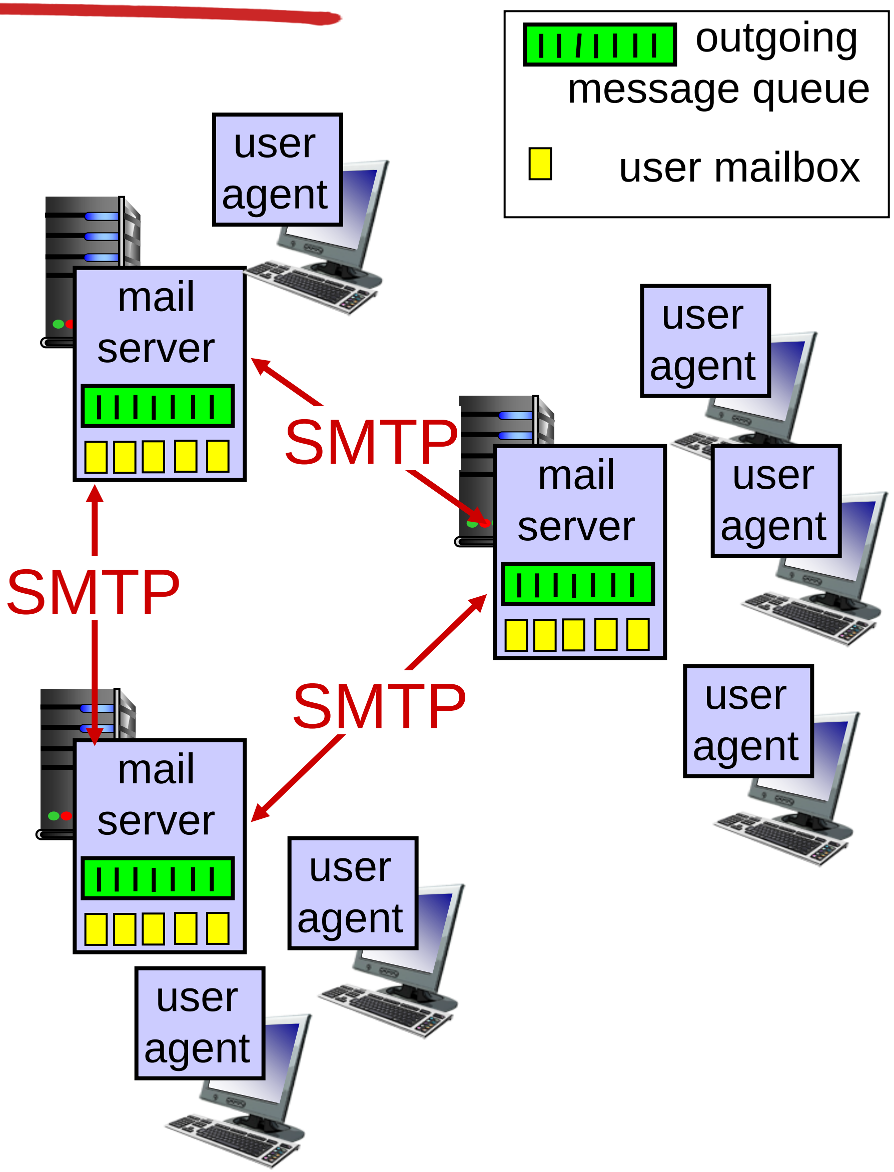
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Outlook, Thunderbird, iPhone mail client
- outgoing, incoming messages stored on server



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

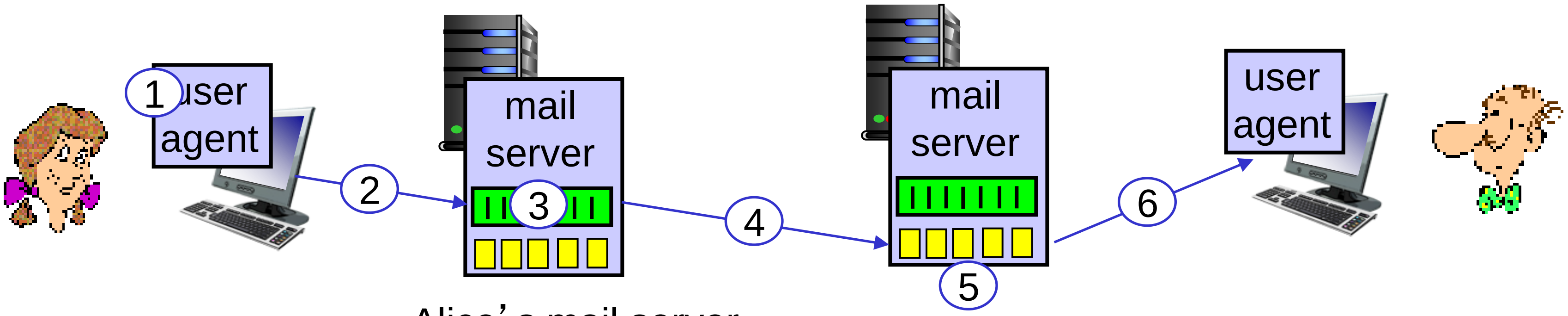


Electronic Mail: SMTP [RFC 2821]

- 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 (like HTTP)
  - response: status code and phrase

Scenario: Alice sends message to Bob

1) Alice uses UA to compose message “to” Bob@yahoomail.com
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



Alice’s mail server

(HotMail!!)

Bob’s mail server

(Yahoo!!)

Sample SMTP interaction

- After the TCP connection is established with the server, it sends a 220

```
S: 220 yahoomail.com
C: HELO Hotmail.com
S: 250  Hello Hotmail.com, pleased to meet you
C: MAIL FROM: <Alice@Hotmail.com>
S: 250 Alice@Hotmail.com... Sender ok
C: RCPT TO: <Bob@yahoomail.com>
S: 250 Bob@yahoomail.com ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Hi, How are you?
C: This is just a SMTP testing.
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 yahoomail.com closing connection
```

SMTP: final words

- SMTP uses persistent connections
  - All the emails are sent over a single TCP connection.

comparison with HTTP:

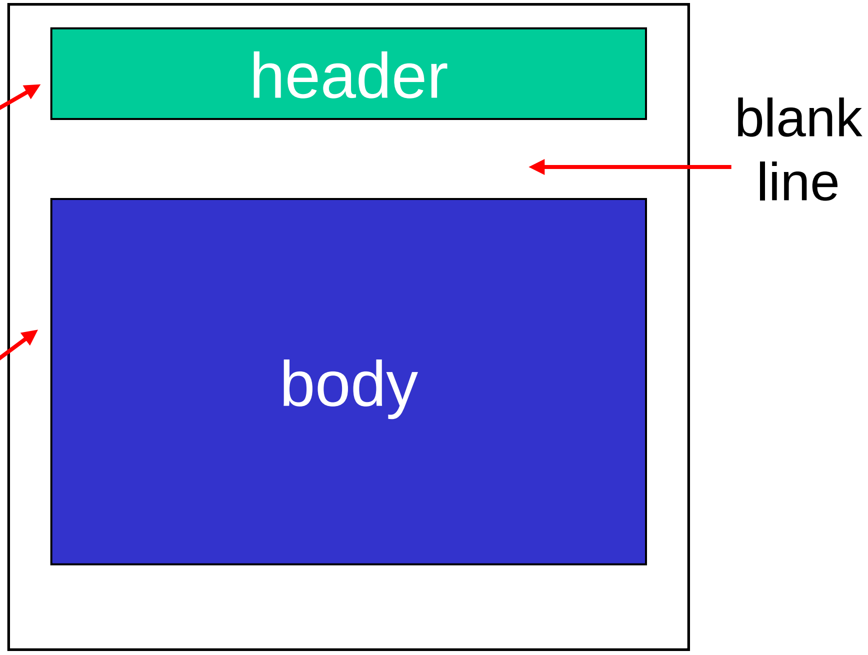
- HTTP: pull
- SMTP: push
  - the TCP connection is initiated by the machine that wants to send the file
- HTTP: each object encapsulated in its own response message
- SMTP: places all of the message's objects into one message.

Mail message format

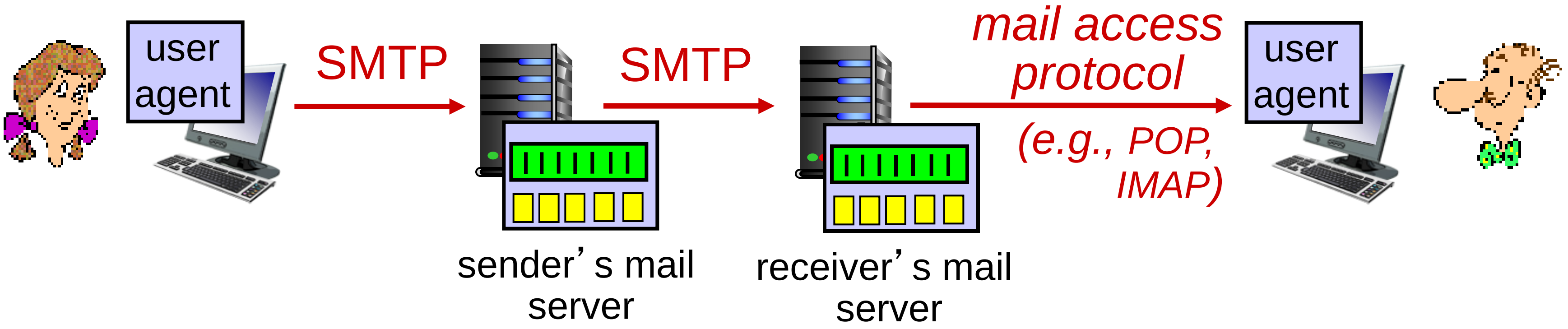
SMTP: protocol for exchanging email messages

RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject:
- 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, download
  - IMAP: Internet Mail Access Protocol [RFC 1730]: more features, including manipulation of stored messages on server

POP3 protocol

authorization phase

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

```
S: +OK POP3 server ready
C: user alice
S: +OK
C: pass student
S: +OK user successfully logged on
```

transaction phase, client:

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

```
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
```

After processing quit command, the POP3 server enters update phase and removes deleted messages from the mailbox.

POP3 (more) and IMAP

more about POP3

- POP3 uses port 110
- A user agent using POP3 can often be configured (by the user) to:
  - "download and delete" or
  - to "download and keep".
- In "download and delete" user cannot re-read e-mail if he changes client

IMAP

- IMAP uses port 993
- keeps all messages in one place: at server
- allows user to organize messages in folders
- keeps user state across sessions:
  - names of folders and mappings between message IDs and folder name

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

DNS: domain name system

people: many identifiers:

- SSN, name, passport #

Internet hosts, routers:

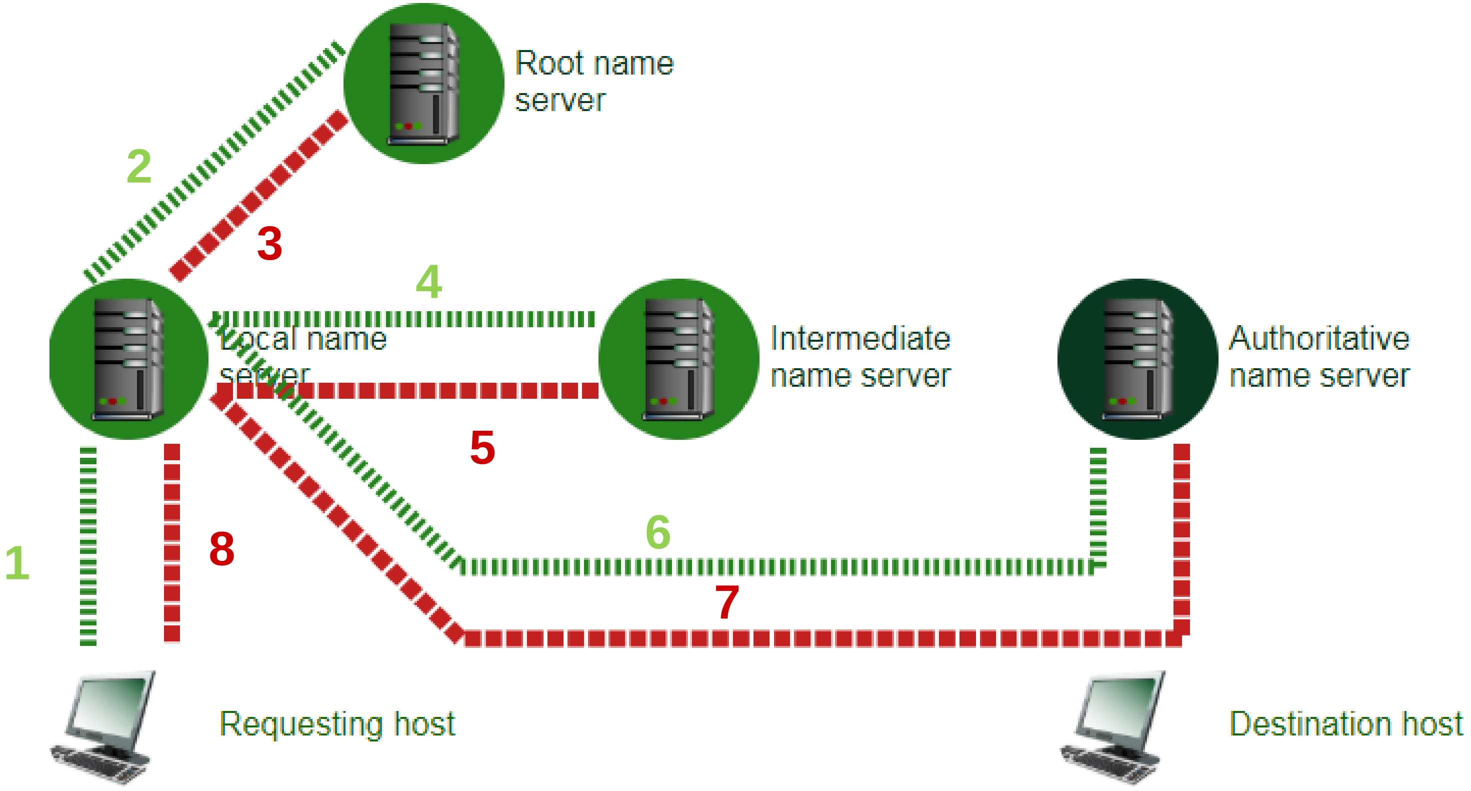
- IP address (32 bit) - used for addressing datagrams
- "name", e.g., www.yahoo.com - used by humans

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"

Bigger Picture of DNS Operation



DNS: services, structure

DNS services

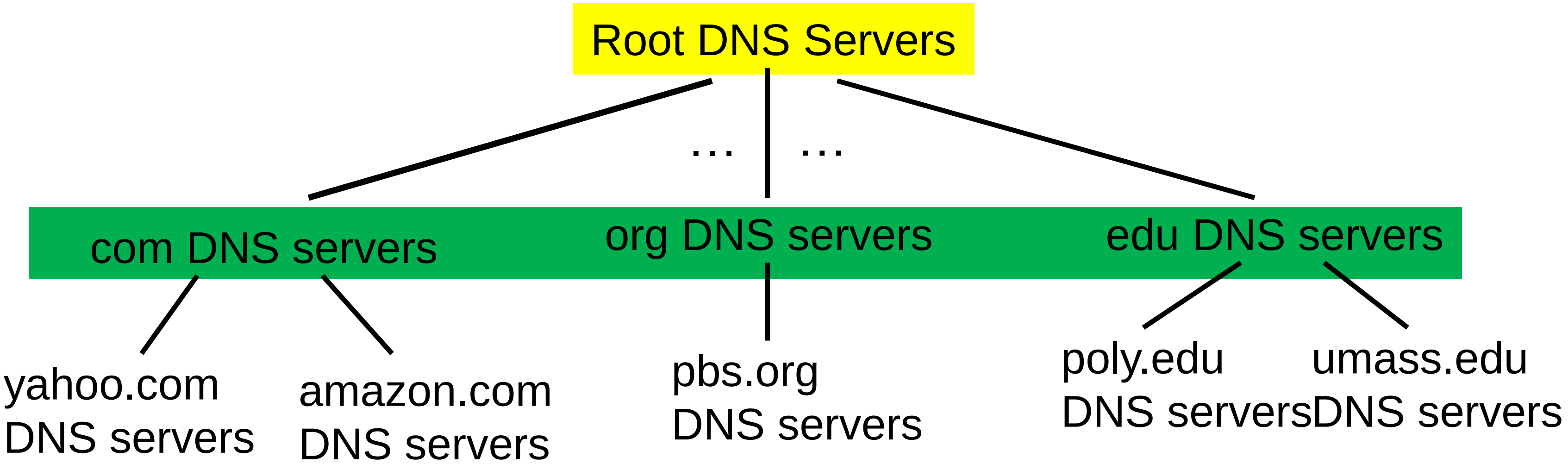
- hostname to IP address translation
- host aliasing
  - Canonical (real), alias names
  - A host with canonical (real) host name (e.g., relay1.west-coast.enterprise.com) may have several alias names (www.enterprise.com, enterprise.com)
- mail server aliasing, e.g.,
- Permit both mail/Web server to have identical (aliased) host name
- **someone@yahoo.com** instead of **someone@mail.yahoo.com**
- load distribution
  - replicated Web servers: many IP addresses correspond to one name
- DNS is built over UDP

why not centralize DNS?

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

A: doesn't scale!

DNS: a distributed, hierarchical database

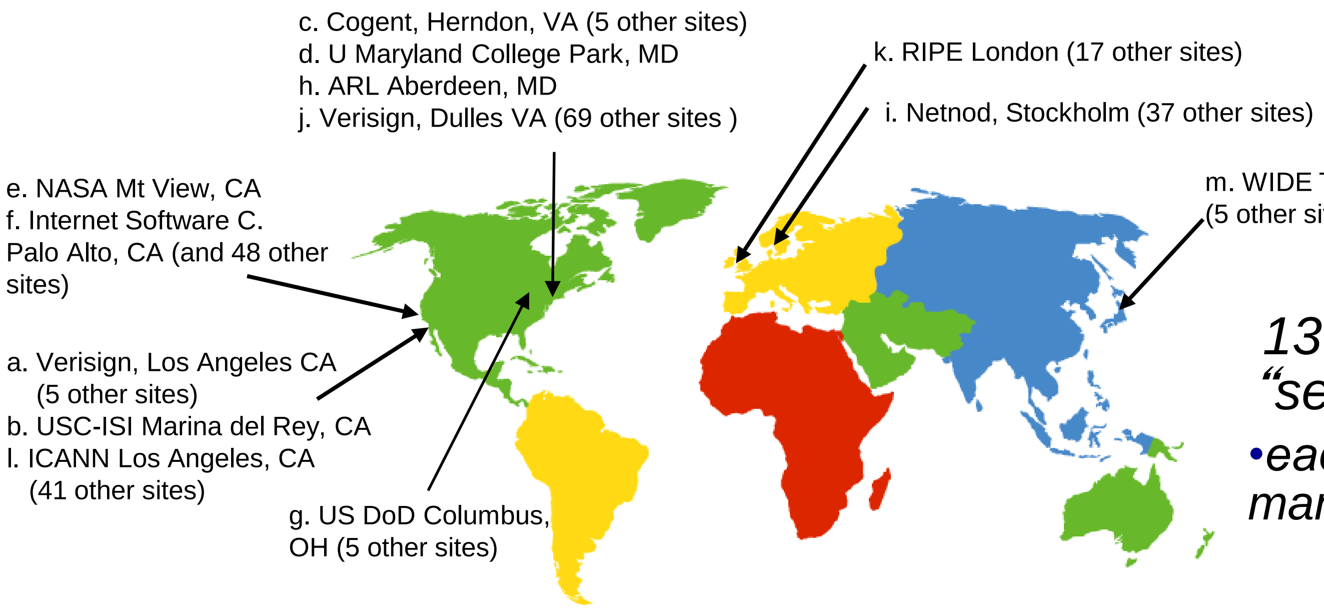


client wants IP for www.amazon.com; 1st approximation:

- 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

DNS: root name servers

- Root name servers
  - contacted by ***local name server*** that can not resolve name
- root name server:
  - *13 logical root name "servers" worldwide each "server" replicated many times*
  - return IP addresses of TLD servers to local name server


c. Cogent, Herndon, VA (5 other sites)
d. U Maryland College Park, MD
h. ARL Aberdeen, MD
j. Verisign, Dulles VA (69 other sites)
k. RIPE London (17 other sites)
i. Netnod, Stockholm (37 other sites)
e. NASA Mt View, CA
f. Internet Software C.
Palo Alto, CA (and 48 other
sites)
m. WIDE Tokyo
(5 other sites)
a. Verisign, Los Angeles CA
(5 other sites)
b. USC-ISI Marina del Rey, CA
l. ICANN Los Angeles, CA
(41 other sites)
g. US DoD Columbus,
OH (5 other sites)

13 logical root name "servers" worldwide

- *each "server" replicated many times*

TLD, authoritative servers

- *top-level domain (TLD) servers (or Intermediate Name Server):*
  - responsible for com, org, net, edu, aero and all top-level country domains, e.g.: uk, fr, ca, jp
  - *For example, Network Solutions* maintains servers for .com TLD
  - *and, 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

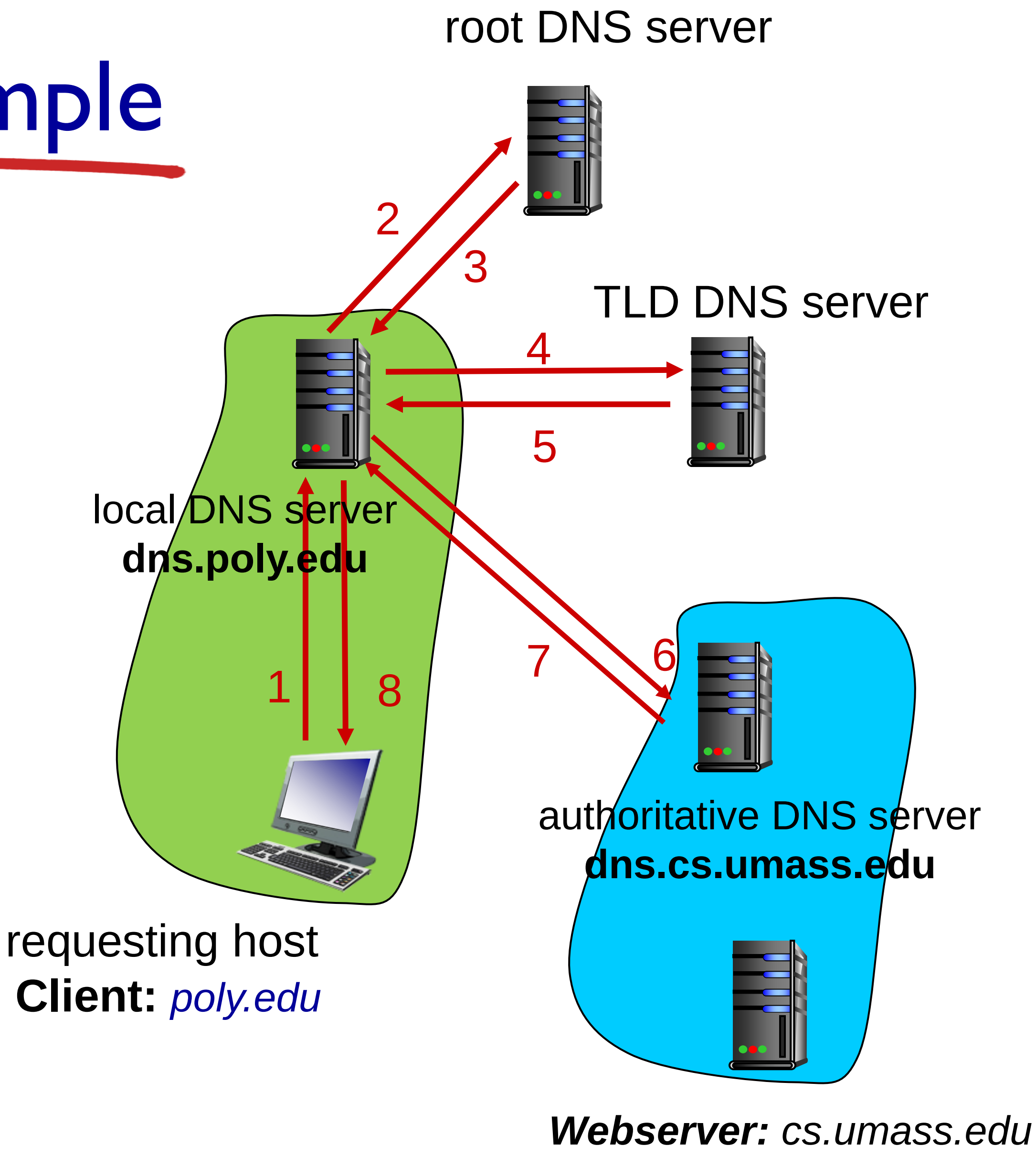
- Local Name Server
- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
  - also called "*default name server*"
- when host makes DNS query, query is sent to its local DNS server
  - has *local cache* of recent name-to-address translation pairs (but may be 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:

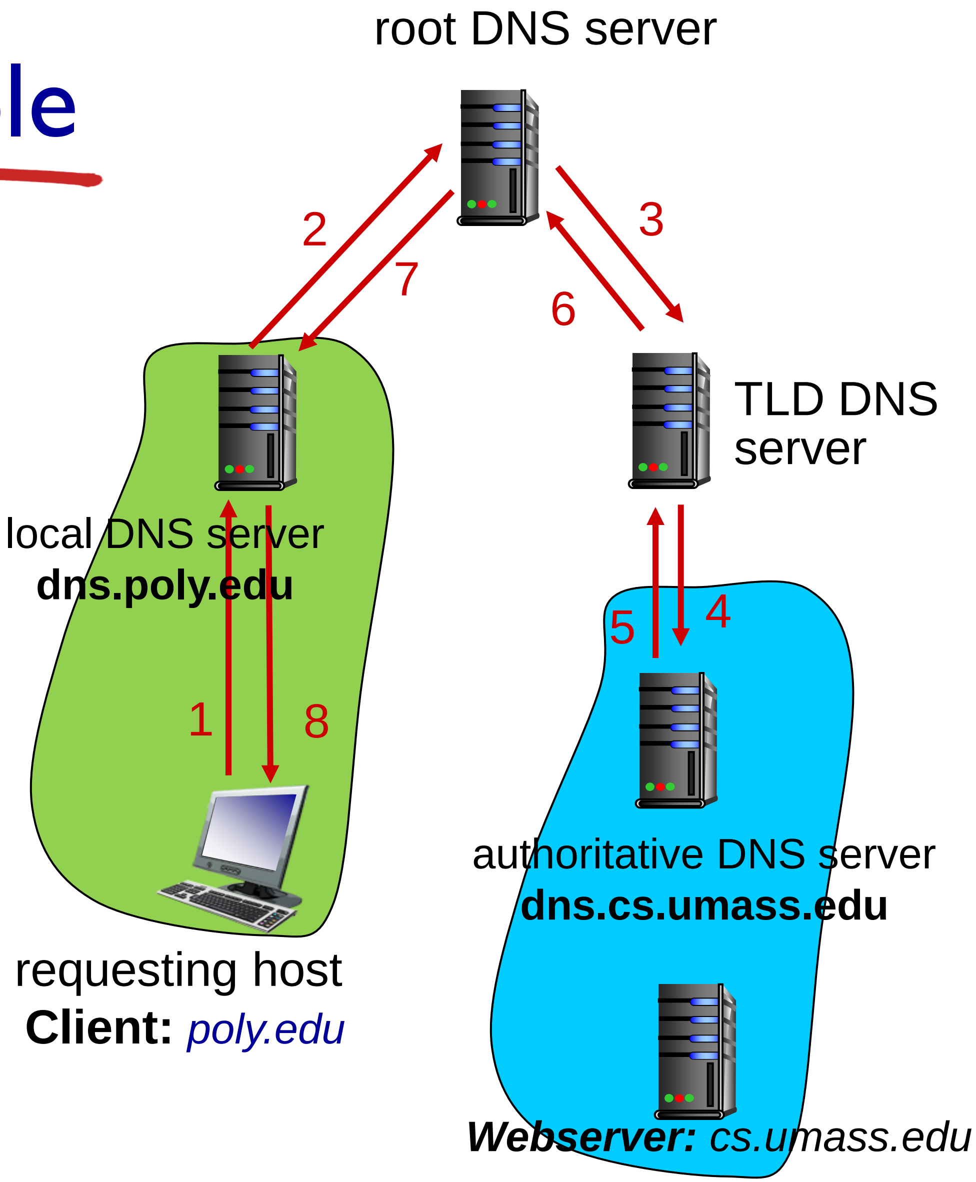
- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



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 *caches* mapping
  - cache entries timeout (disappear) after some time (TTL)
  - TLD servers (intermediate name servers) typically cached in local name servers
    - thus root name servers not often visited
- cached entries may be *out-of-date*
  - if name host changes IP address, may not be known Internet-wide until all TTLs expire

DNS records

DNS: distributed database storing resource records (RR)

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

- Query(domain name, RR type)
  - Resource Record (RR) type is like an attribute type
- Answer(values, additional RRs)

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 (mail exchanger)

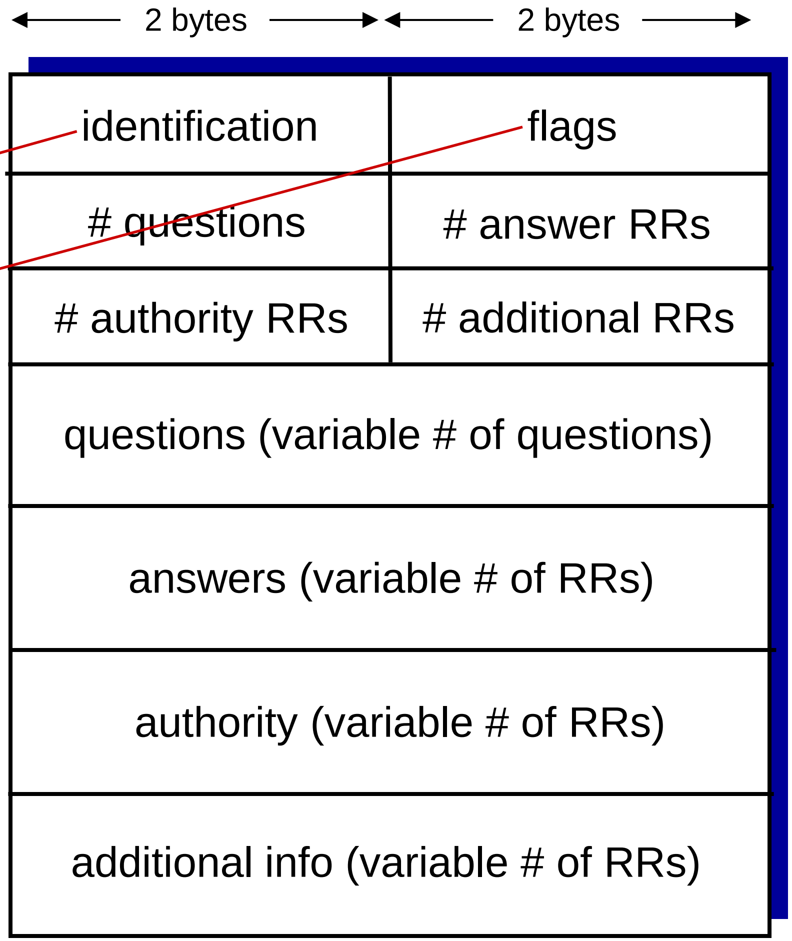
- **`value`** is name of mailserver associated with **`name`**

DNS protocol, messages

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

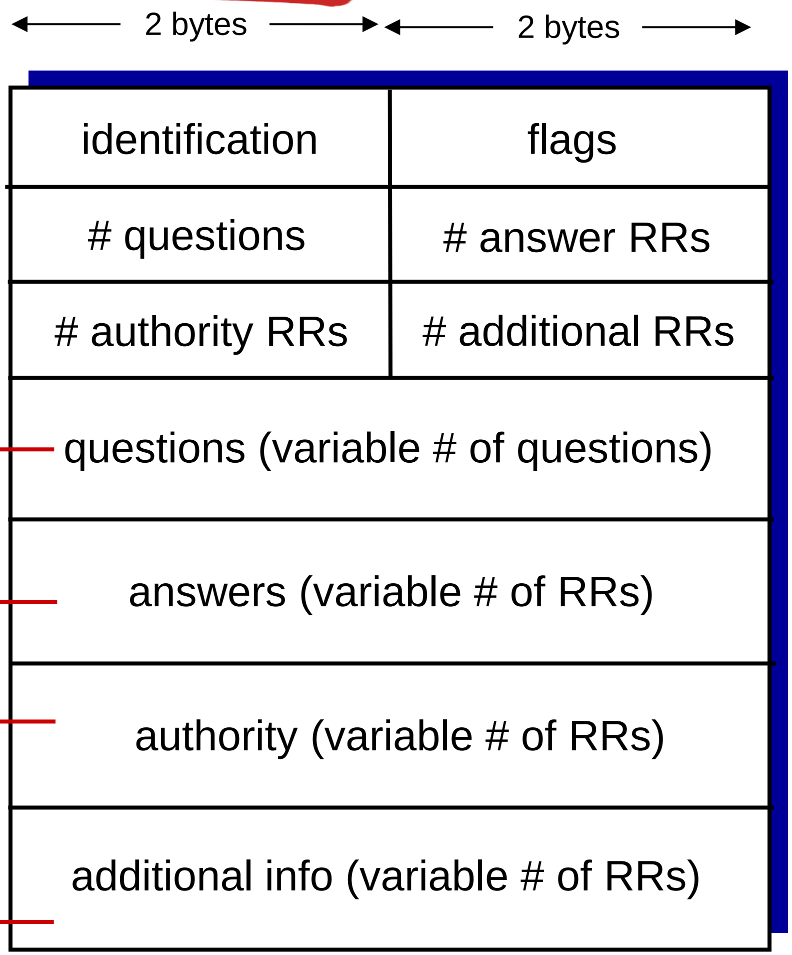
message header

- identification: 16 bit # for query, reply to query uses same #
- flags:
  - query(0) or reply(1)
  - recursion desired
  - recursion available
  - reply is authoritative (whether DNS Server is authoritative server. Sometimes there is intermediate DNS server which isn’t 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



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)	

name, type fields for a query

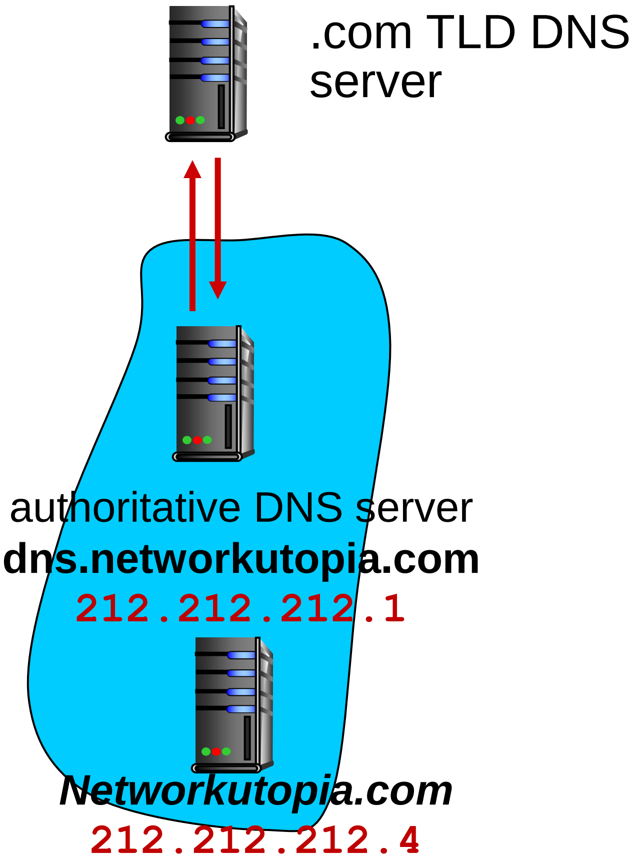
RRs in response to query

records of other authoritative servers

additional "helpful" info that may be used, (For example, RR with IP address of someone's mail server that I queried for)

How to insert Records into DNS?

- example: new startup "*Network Utopia*"
- **First step:** Assign names and IP addresses to your web server and authoritative dns server.



Inserting records into DNS

- example: new startup "*Network Utopia*"
- Register name networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
  - **2nd step:** provide names, IP addresses of authoritative name server (primary and secondary)
  - **3rd step:** registrar inserts two RRs into .com TLD server:
    RR1: `(networkutopia.com, dns1.networkutopia.com, NS)`

    RR 2: `(dns1.networkutopia.com, 212.212.212.1, A)`
- 4th step: In authoritative server add (Located on the Server premises)
  - type A record for www.networkuptopia.com, e.g.,
    - `networkutopia.com,212.212.212.4, A`
  -

Attacking DNS

DDoS attacks

- bombard root servers with traffic
  - not successful to date
  - local DNS servers cache IPs of TLD servers, allowing root server bypass

redirect attacks

- man-in-middle
  - Intercept queries
- DNS poisoning
  - Send bogus replies to DNS server, which caches