COMP 445
Data Communications & Computer networks
Winter 2022

Application Layer

- ✓ ~~Principles of network applications~~
- ✓ ~~Web and HTTP~~
- ✓ Electronic mail
- ✓ DNS
- ✓ P2P applications
- ✓ Video streaming and CDN
- ✓ Sockets

Application Layer – Part 2

- ✓ Electronic mail
  - ✓ Overview
  - ✓ SMTP
  - ✓ Mail access protocols
- ✓ DNS
  - ✓ Overview
  - ✓ DNS records and messages

Learning objectives

- To explain the operation of the email application, its components, and the application-layer protocols involved
- To describe the operation of the Simple Mail Transfer Protocol and the operation of mail access protocols such as POP3 and IMAP, and HTTP, identifying the differences among them.
- To understand the way DNS Works and the services it provides
- To identify the components of DNS as a distributed system and the role of servers at the different layers in the hierarchical architecture

Application Layer – Part 2

- ✓ Electronic mail
  - ✓ Overview
  - ✓ SMTP
  - ✓ Mail access protocols
- ✓ DNS
  - ✓ Overview
  - ✓ DNS records and messages

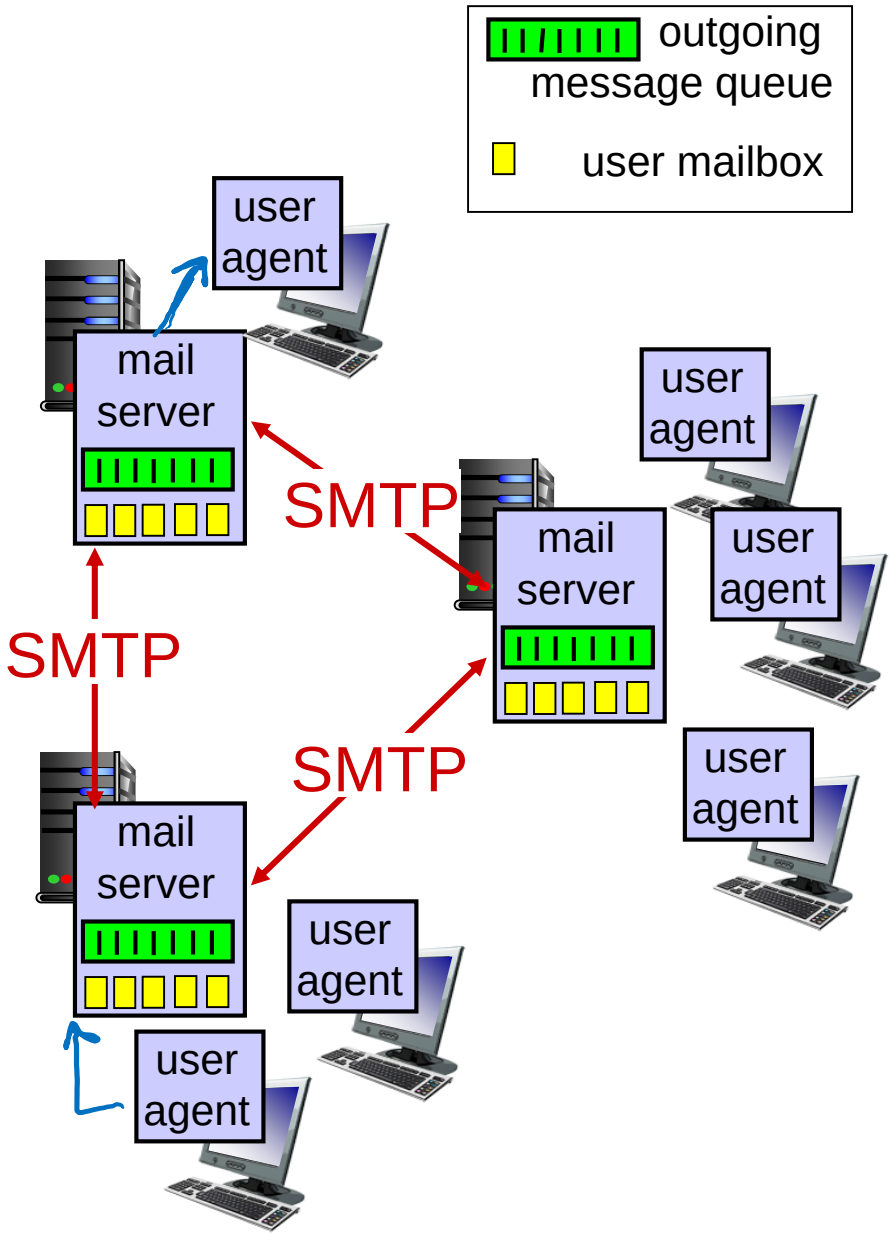
Electronic mail

Three major components:

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

User Agent

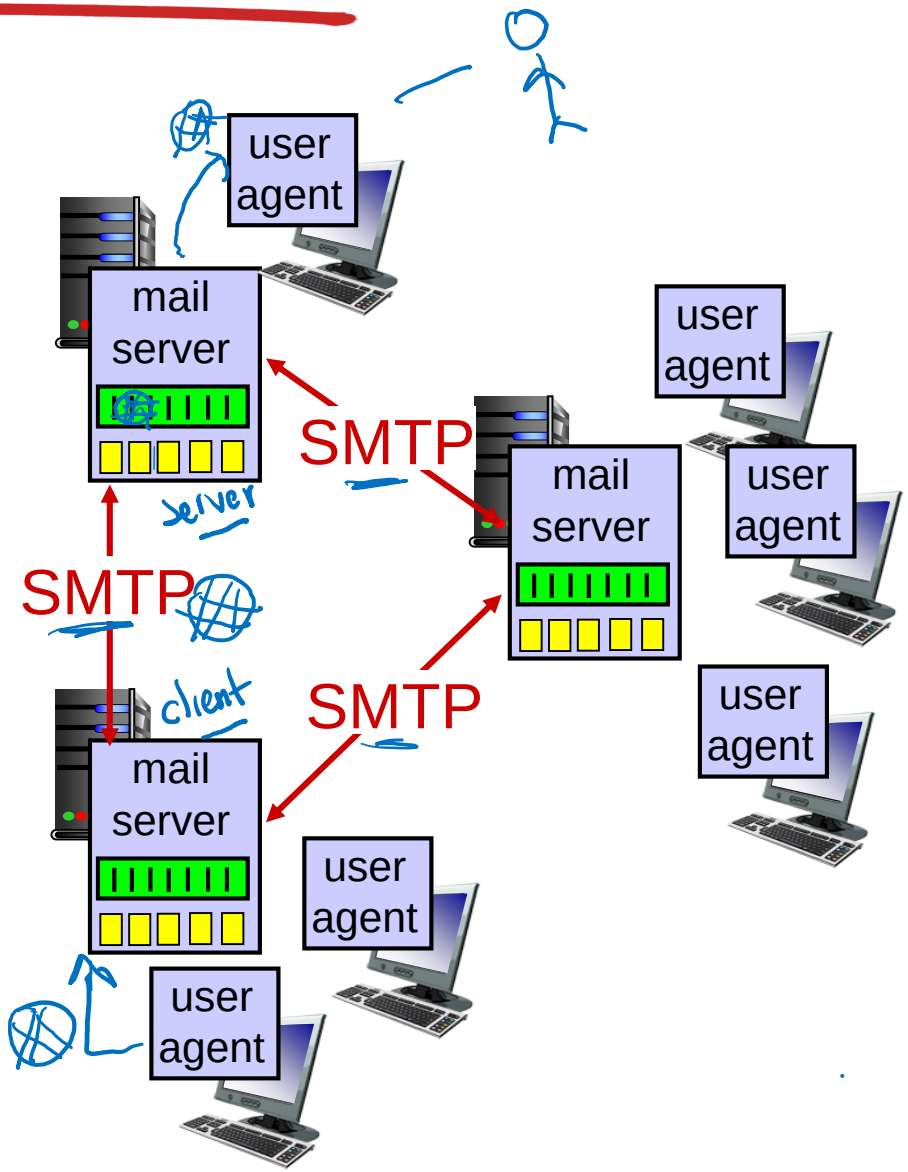
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Outlook, Thunderbird, iPhone mail client
- outgoing, incoming messages stored on server



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
  - SMTP client: sending mail server
  - SMTP server: receiving mail server

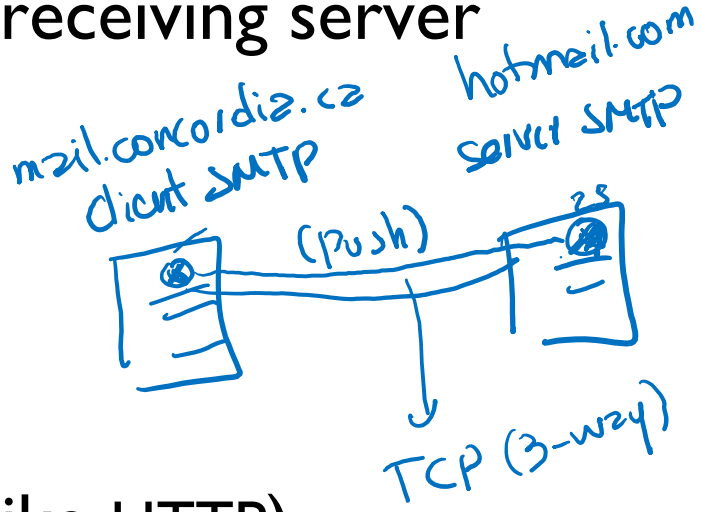


Application Layer – Part 2

- ✓ Electronic mail
  - ✓ Overview
  - ✓ SMTP
  - ✓ Mail access protocols
- ✓ DNS
  - ✓ Overview
  - ✓ DNS records and messages

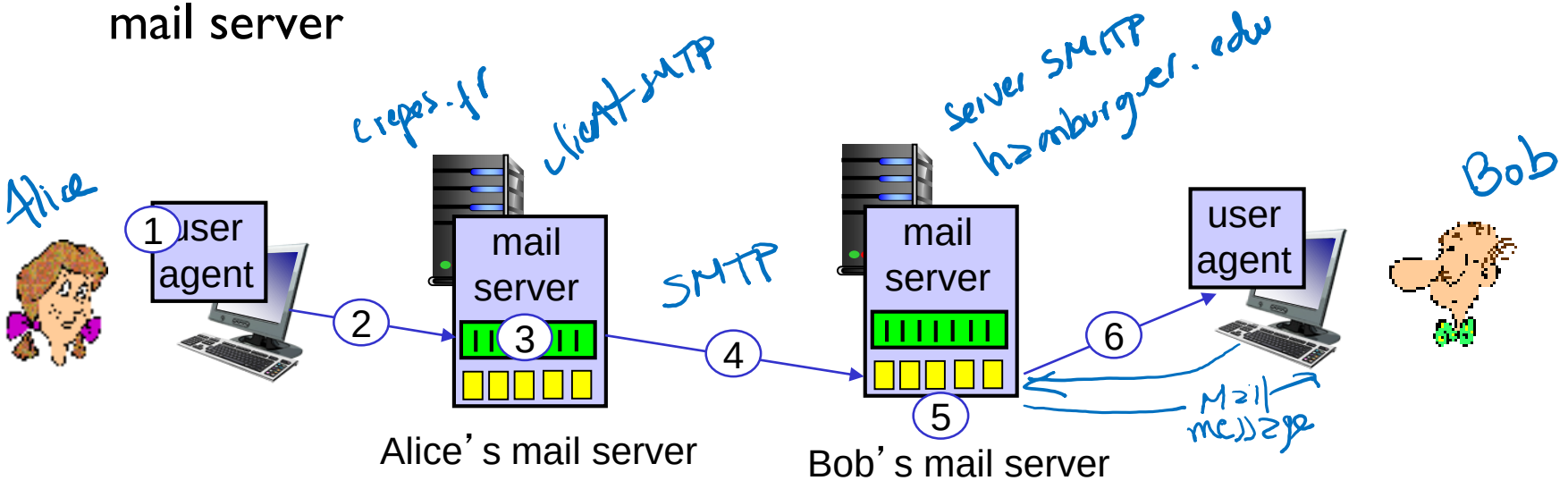
Electronic Mail: SMTP [RFC 5321]

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



Scenario: Alice sends message to Bob

1) Alice uses UA to compose message "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)

SMTP: final words

- SMTP uses persistent connections
- SMTP requires message (header & body) to be in 7-bit ASCII
- SMTP server uses `CRLF.CRLF` to determine end of message

comparison with HTTP:

- HTTP: pull
- SMTP: push
- both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response message
- SMTP: multiple objects sent in multipart message

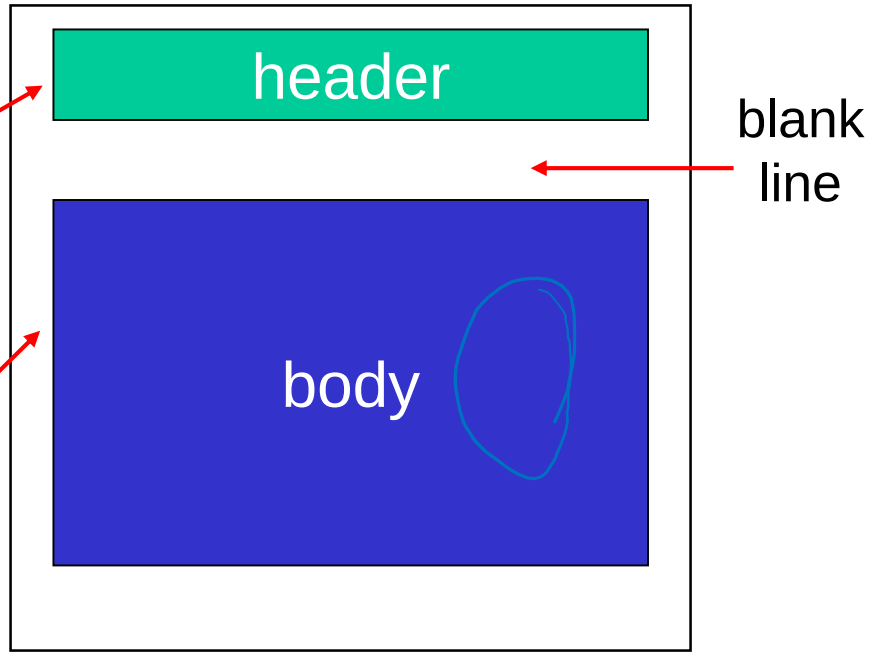
Mail message format

SMTP: protocol for exchanging email messages

RFC 5322: standard for Internet message format:

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

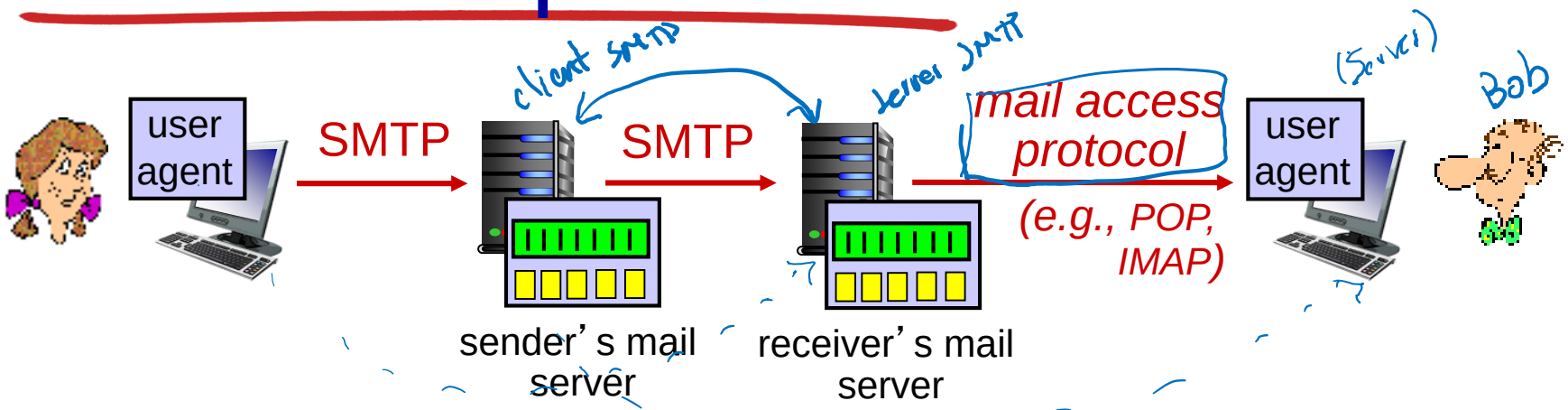
  different from SMTP MAIL FROM, RCPT TO: commands!
- Body: the "message"
  - ASCII characters only



Application Layer – Part 2

- ✓ Electronic mail
  - ✓ Overview
  - ✓ SMTP
  - ✓ Mail access protocols
- ✓ DNS
  - ✓ Overview
  - ✓ DNS records and messages

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
  - 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 2 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 (more) and IMAP

more about POP3

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

IMAP

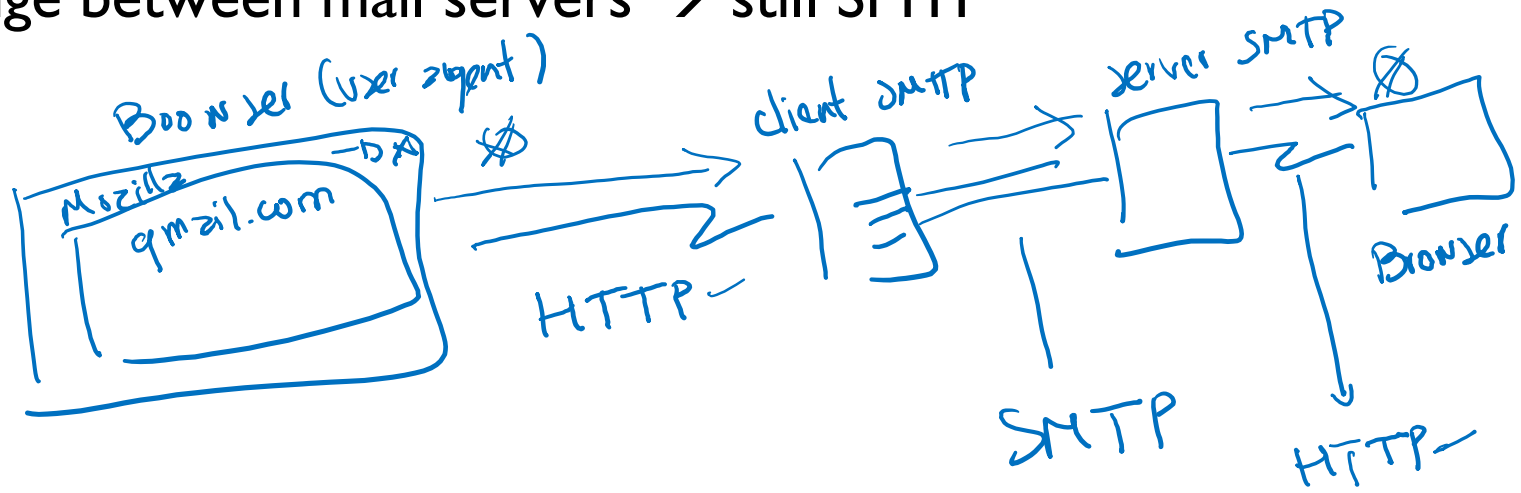
more complex

- 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

Web-based email

Using HTTP

- User agent is an ordinary Web browser
- E-mail messages are sent from recipient mail server to recipient's browser using the HTTP protocol
- E-mail messages are sent from sender user agent to recipient's mail server using the HTTP protocol
- Exchange between mail servers → still SMTP



Application Layer – Part 2

- ✓ Electronic mail
  - ✓ Overview
  - ✓ SMTP
  - ✓ Mail access protocols
- ✓ DNS
  - ✓ Overview
  - ✓ DNS records and messages

DNS: domain name system

people: many identifiers:

- SSN, name, passport #

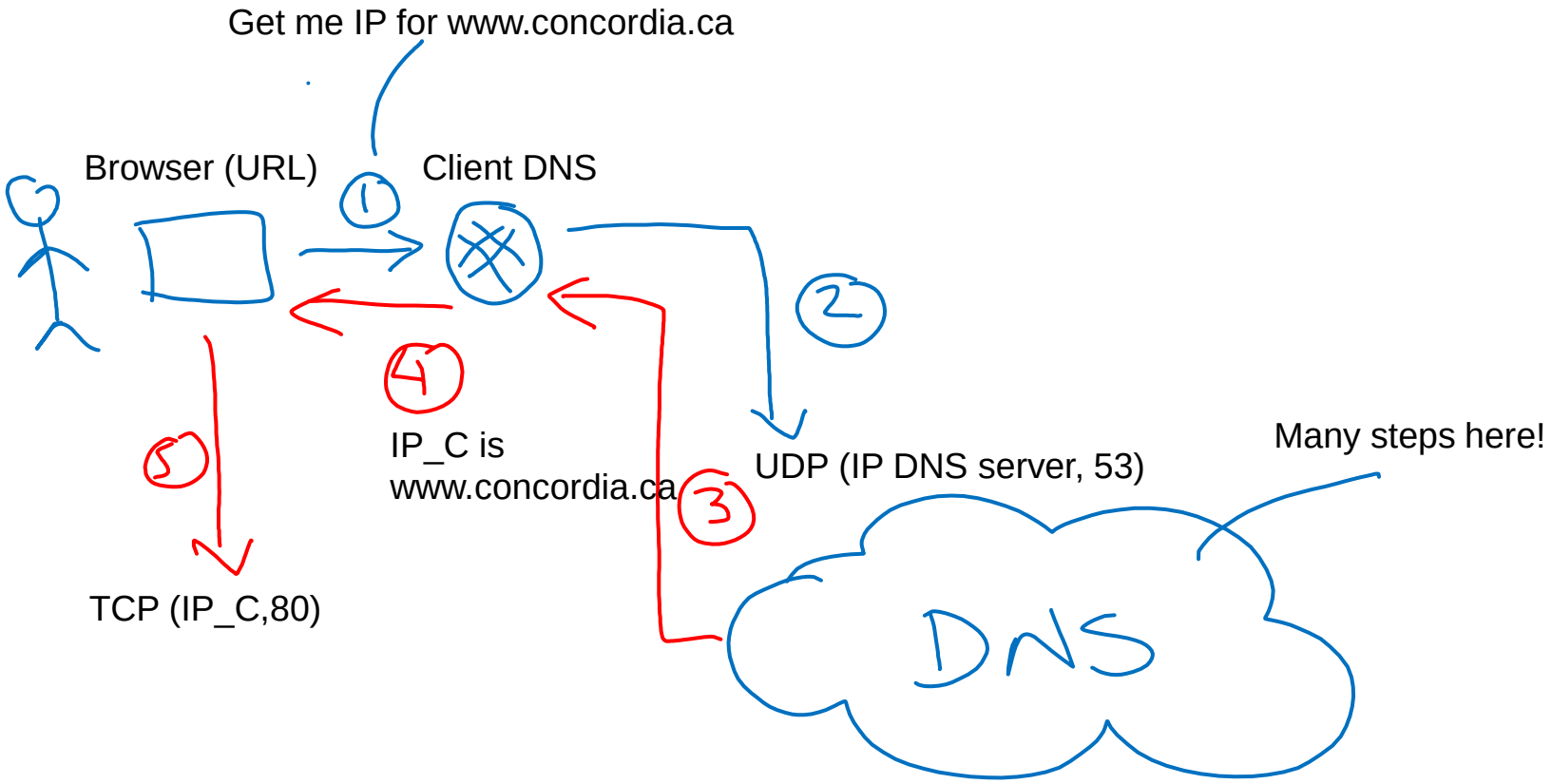
Internet hosts, routers:

- 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"
  - Runs on top of UDP


Get me IP for www.concordia.ca
Browser (URL)
Client DNS
1
2
3
4
5
IP_C is
www.concordia.ca
UDP (IP DNS server, 53)
Many steps here!
DNS
TCP (IP_C,80)

DNS: domain name system

User's host wants to send an HTTP request to web server www.concordia.ca

The same user machine runs the client side of the DNS application.

1. *The browser extracts the hostname, www.concordia.ca, from the URL and passes the hostname to the client side of the DNS application.*
2. *The DNS client sends a query containing the hostname to a DNS server.*
3. *The DNS client eventually receives a reply, which includes the IP address for the hostname.*
4. *The browser receives the IP address from DNS*
5. *Browser can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.*

DNS: services, structure

DNS services

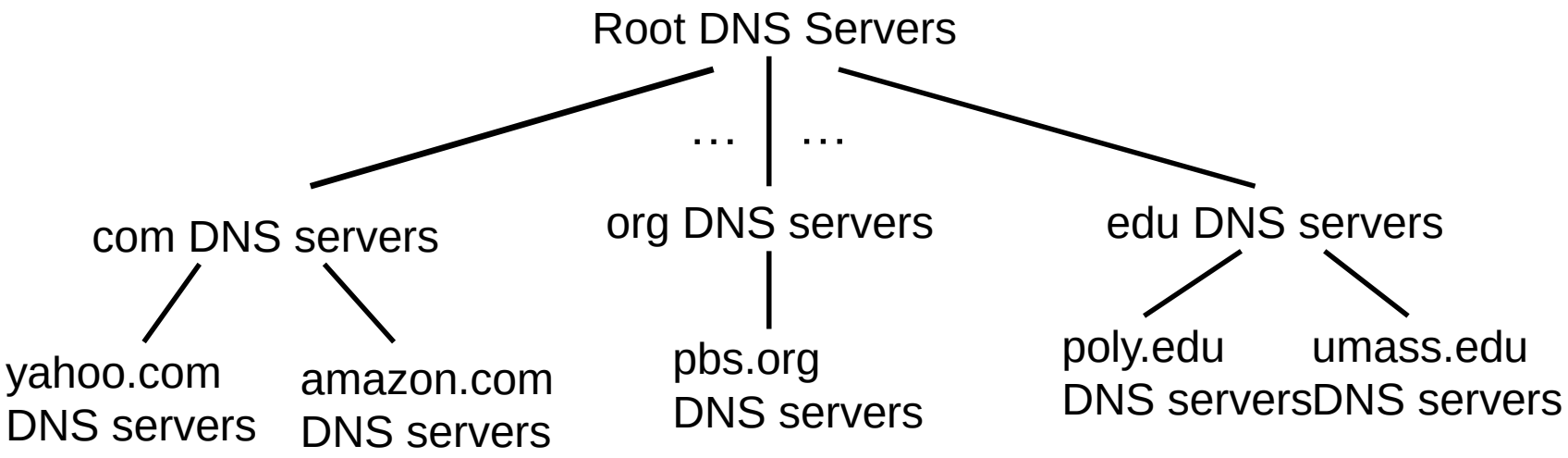
- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

why not centralize DNS?

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

A: doesn't scale!

DNS: a distributed, hierarchical database



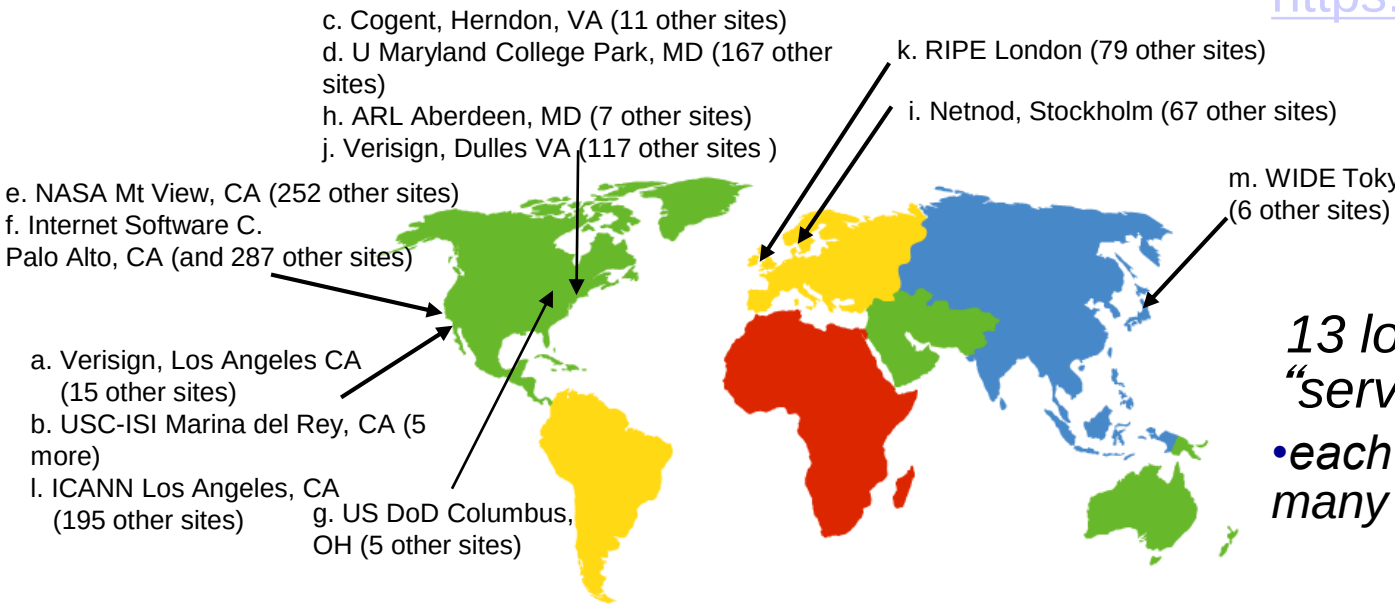
client wants IP for www.amazon.com; 1st approximation:

- 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

- 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

https://root-servers.org



13 logical root name "servers" worldwide

- *each "server" replicated many times*

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
- List of valid TLD is maintain by IANA

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

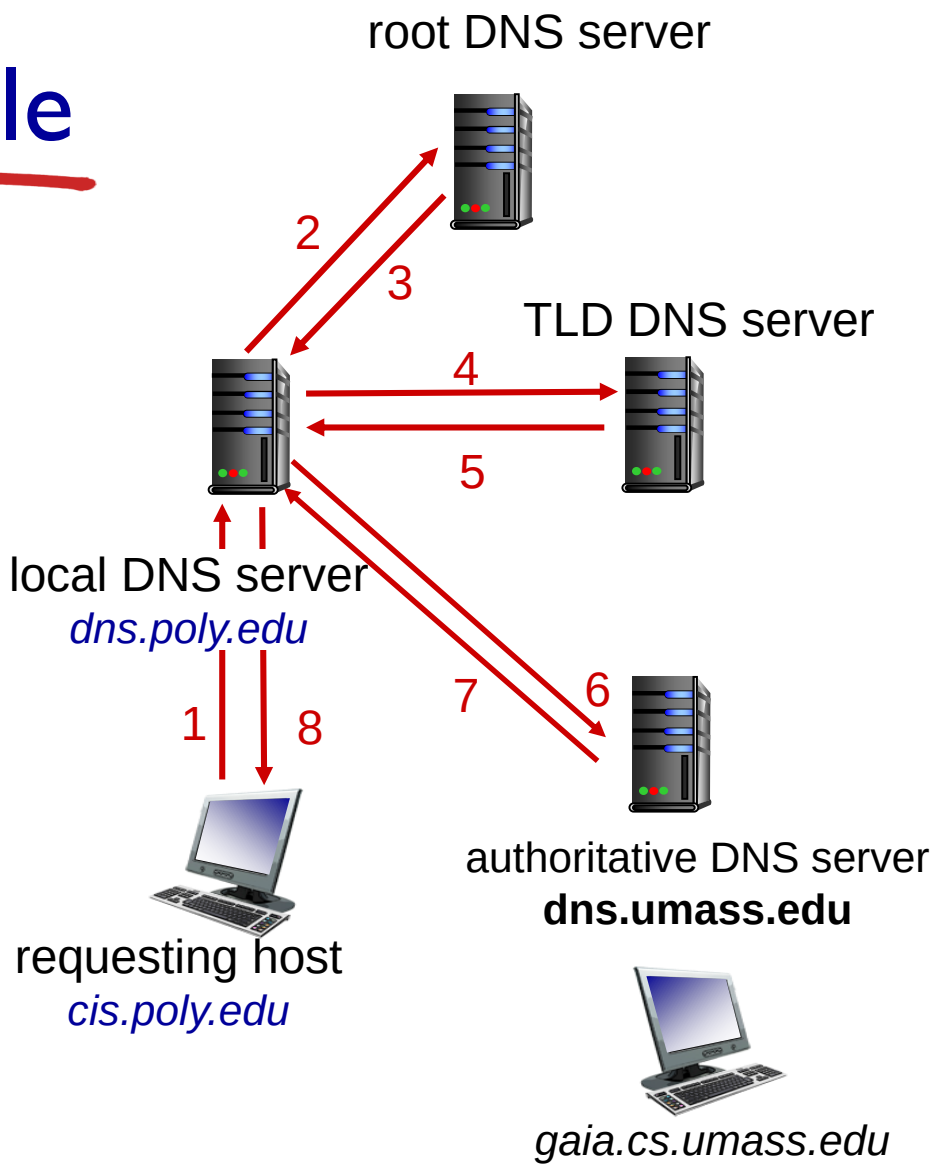
- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
  - also called "default name server"
- when host makes DNS query, query is sent to its local DNS server
  - has local cache of recent name-to-address translation pairs (but may be out of date!)
  - acts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

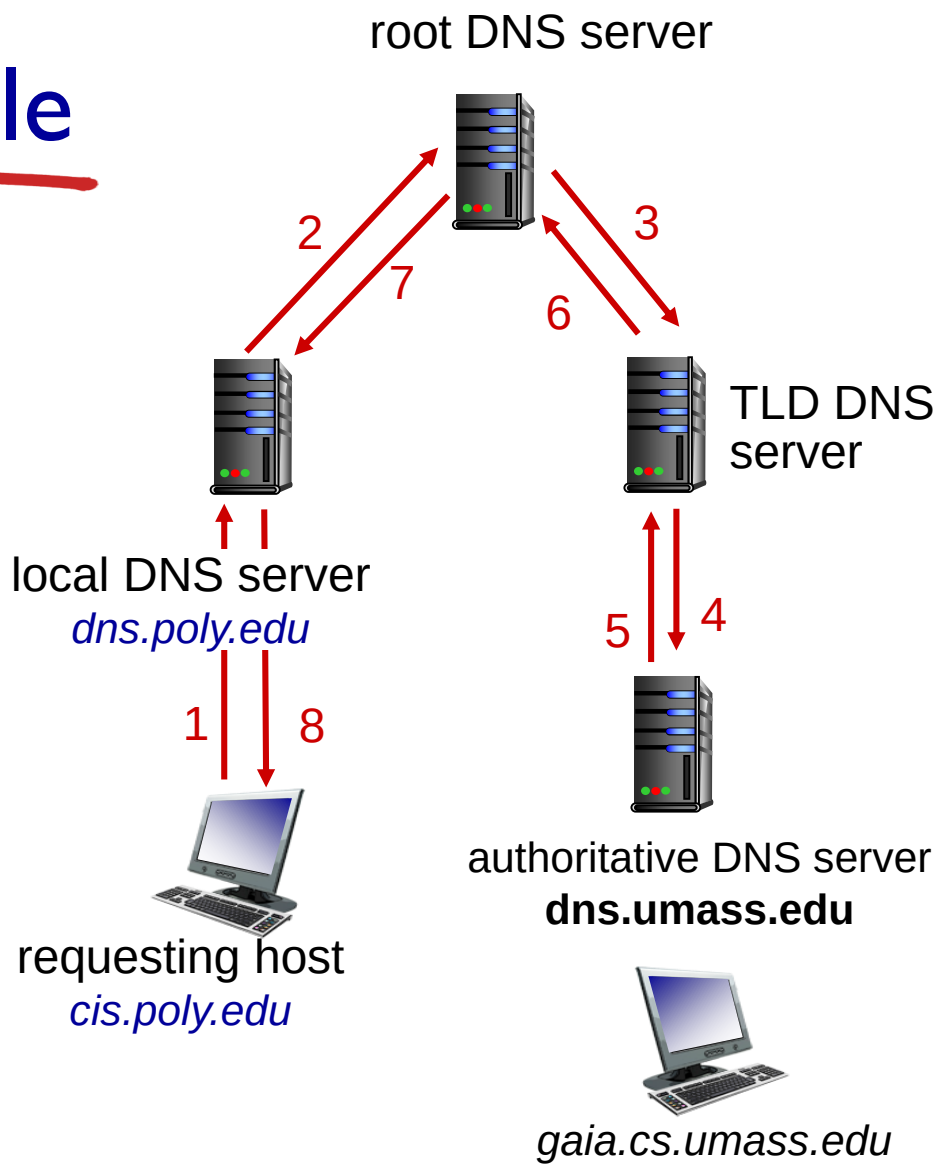
- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



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 *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 IETF standard
  - RFC 2136

DNS records

DNS: distributed database storing resource records (RR)

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

type=A

- **name** is hostname
- **value** is IP address

*(**relay1.bar.foo.com**, 145.37.93.126, A)*

type=NS

- **name** is domain (e.g., foo.com)
- **value** is hostname of authoritative name server for this domain

*(**foo.com**,**dns.foo.com**, NS)*

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 canonical name of mailserver with alias **name**

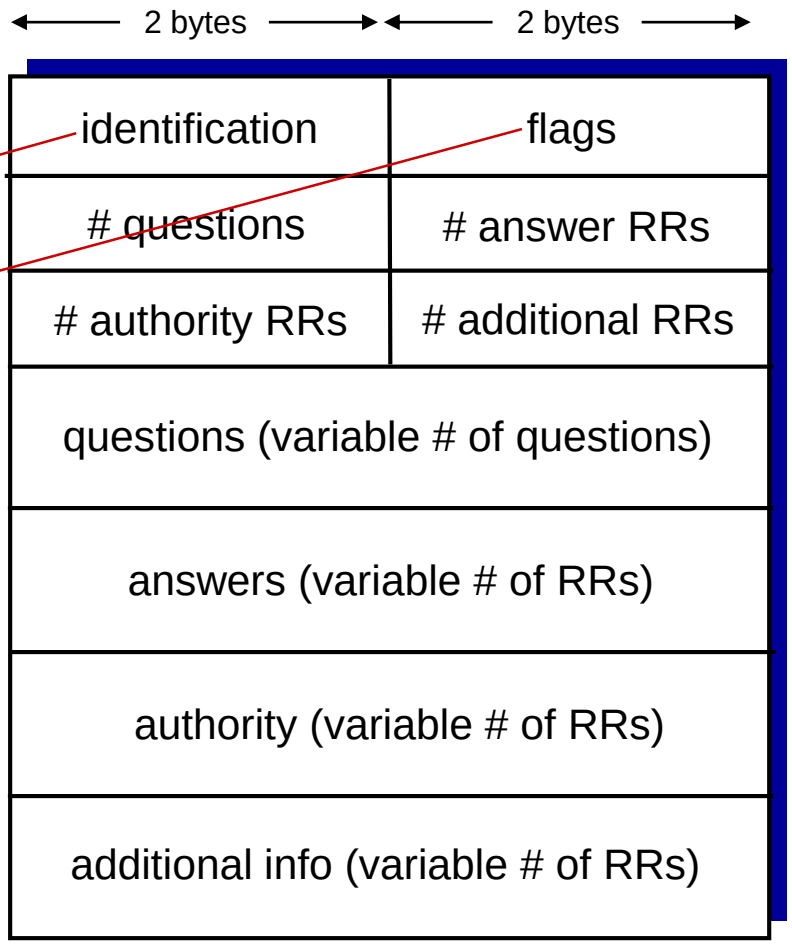
*(**foo.com**, **mail.bar.foo.com**, MX)*

DNS protocol, messages

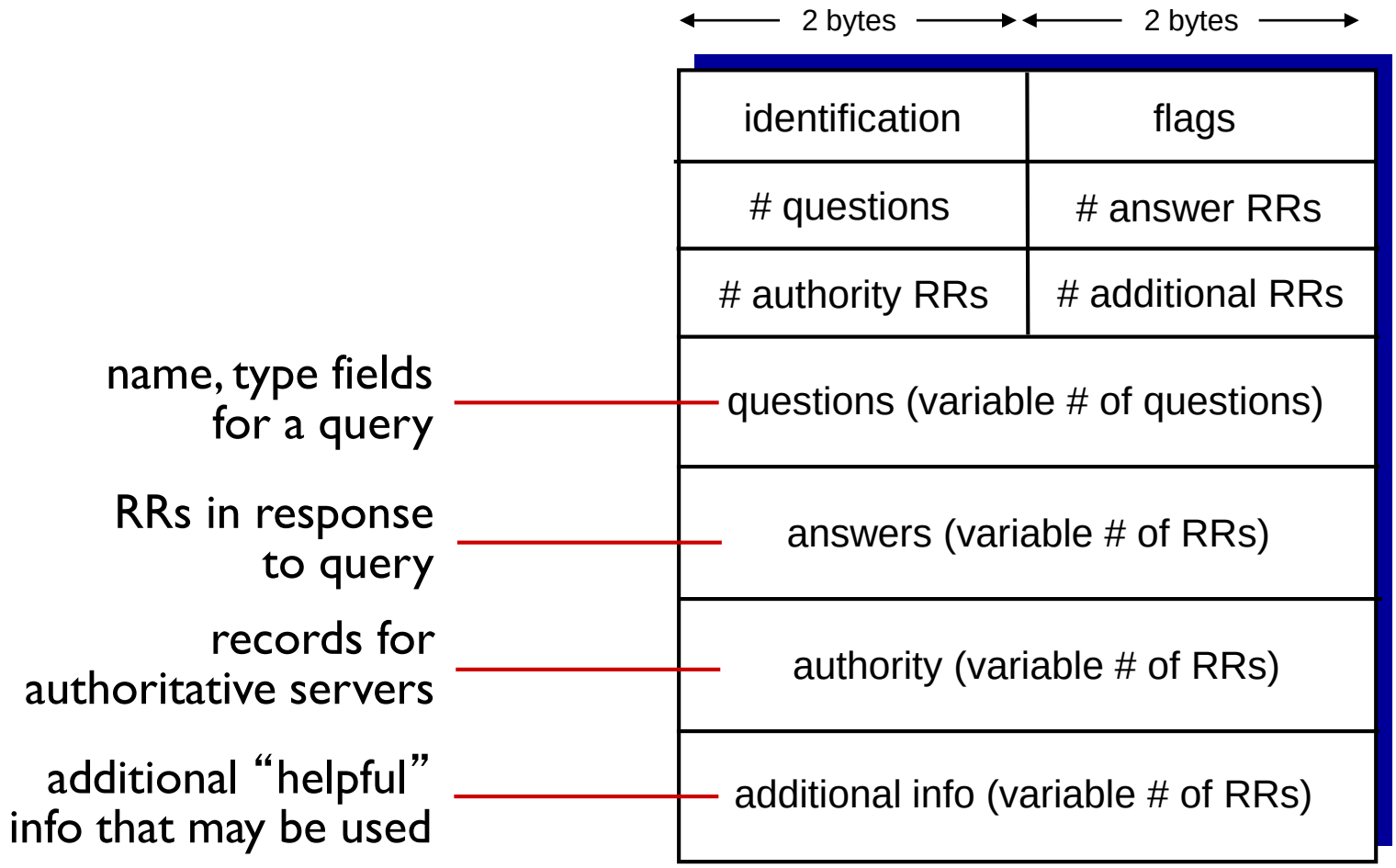
- *query* and *reply* messages, both with same *message format*

message header

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



DNS protocol, messages



Inserting records into DNS

- example: new startup "Network Utopia"
- register name networkuptopia.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.networkuptopia.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 relies to DNS server, which caches

exploit DNS for DDoS

- send queries with spoofed source address: target IP
- requires amplification

References

Figures and slides are taken/adapted from:

- Jim Kurose, Keith Ross, "Computer Networking: A Top-Down Approach", 7th ed. Addison-Wesley, 2012. All material copyright 1996-2016 J.F Kurose and K.W. Ross, All Rights Reserved