DATA COMMUNICATOIN NETWORKING

Instructor: Ouldooz Baghban Karimi

Course Book: Computer Networking, A Top-Down Approach
By: Kurose, Ross

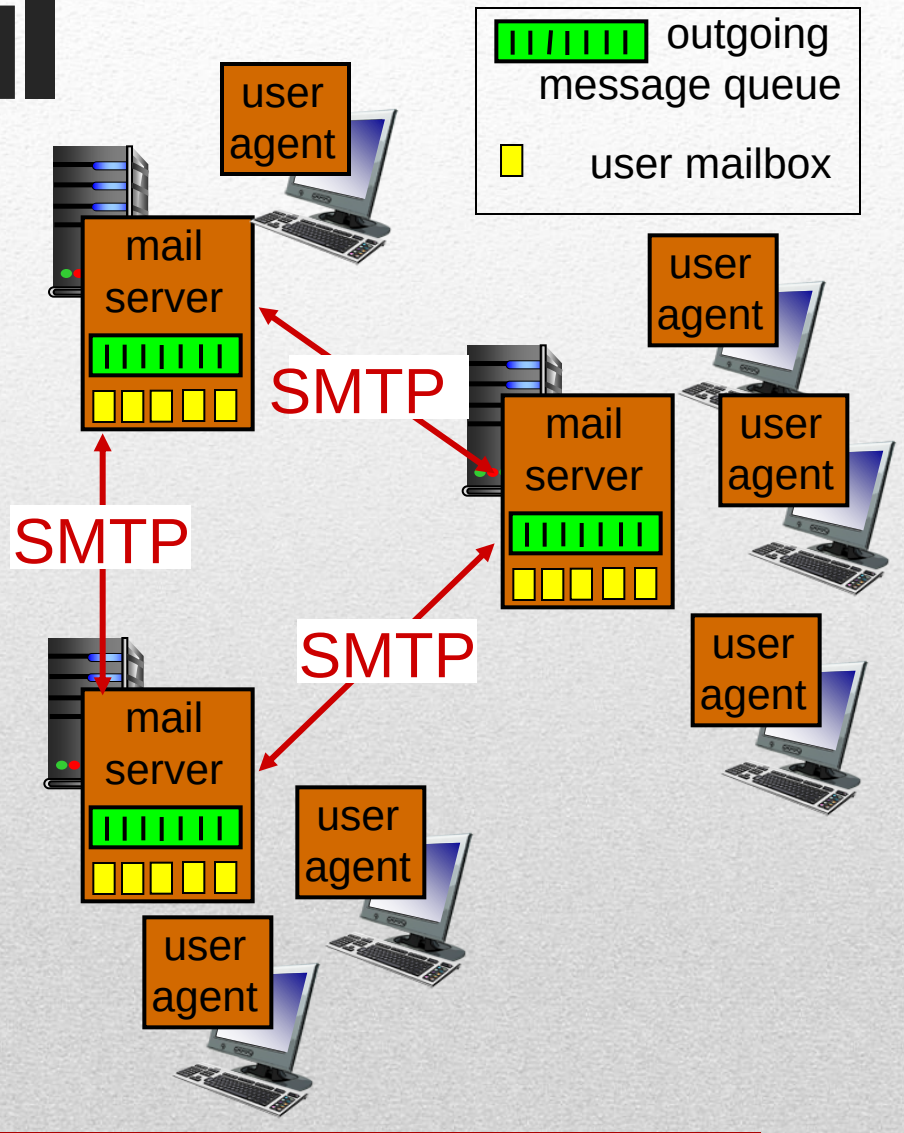
Course Overview

- **Basics of Computer Networks**
  - Internet & Protocol Stack
  - Application Layer
  - Transport Layer
  - Network Layer
  - Data Link Layer

- **Advanced Topics**
  - Case Studies of Computer Networks
  - Internet Applications
  - Network Management
  - Network Security

Electronic Mail

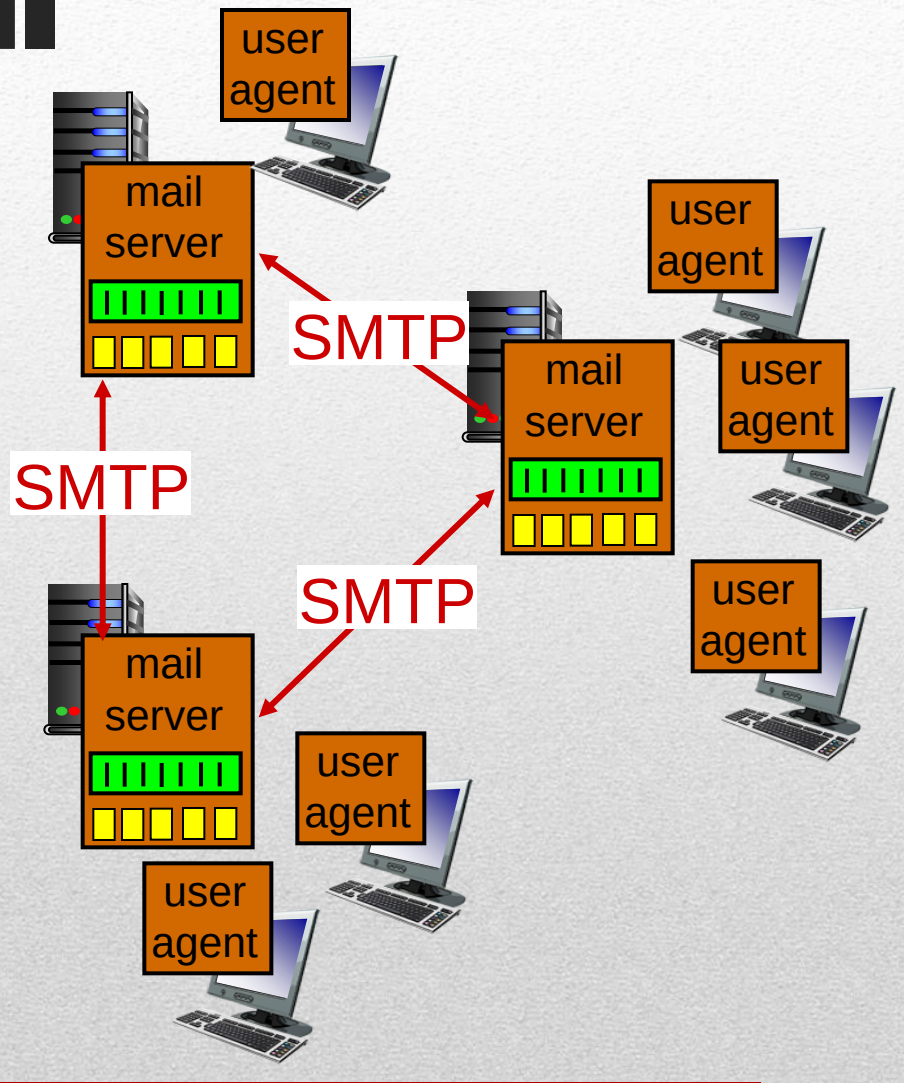
- **Three major components**
  - User agents
  - Mail servers
  - Simple mail transfer protocol: SMTP
- **User Agent**
  - "mail reader"
    - Outlook, Thunderbird, iPhone mail client
  - Composing, editing, reading mail messages
  - Outgoing, incoming messages stored on server



Electronic Mail

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, FTP)
  - Commands: ASCII text
  - Response: status code and phrase
- Messages must be in 7-bit ASCI

Alice Sends Messages to Bob

1) Alice uses UA to compose message "to" `bob@someschool.edu`

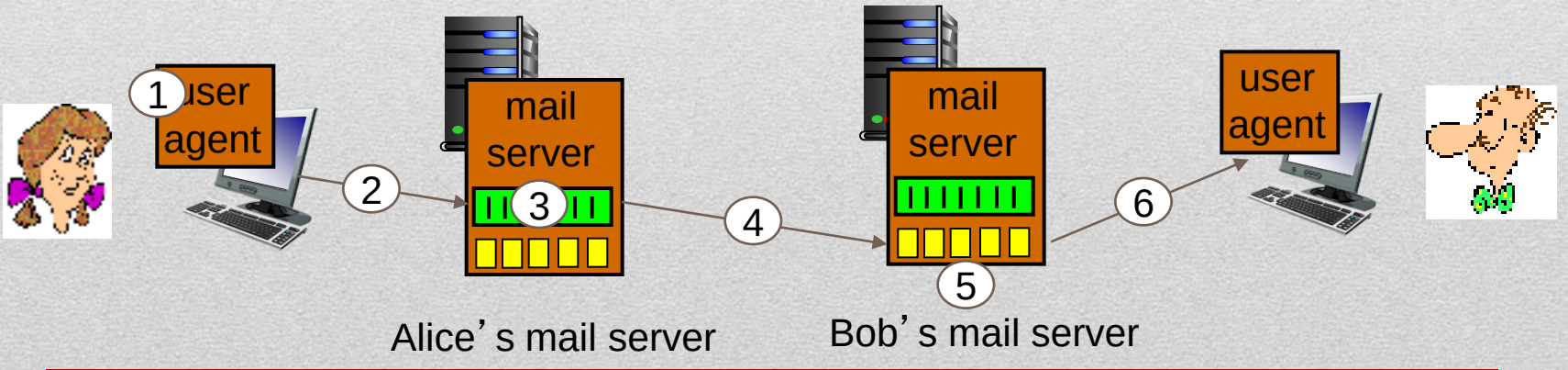
2) Alice's UA sends message to her mail server; message placed in message queue

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
```

SMTP

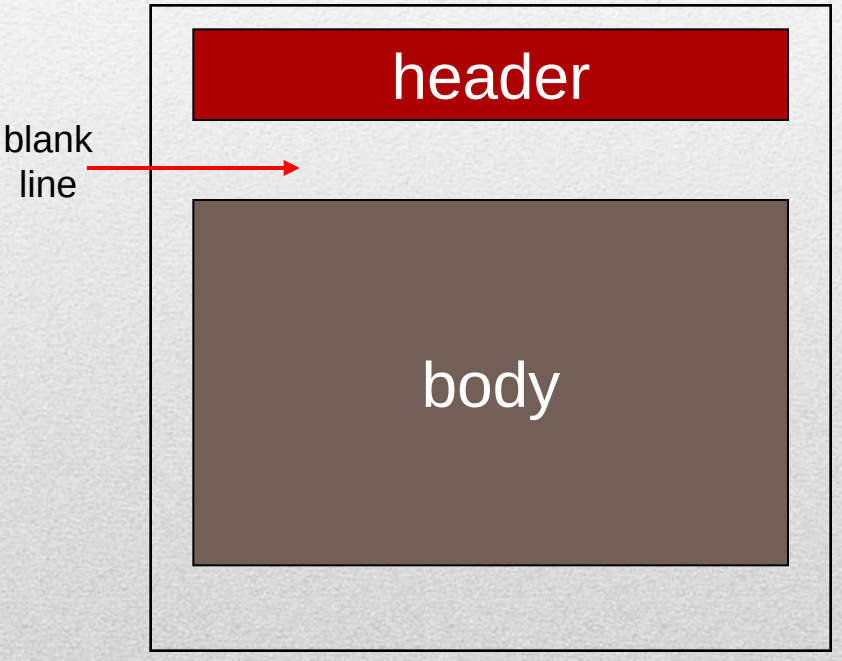
- SMTP uses persistent connections
- SMTP requires message (header & body) to be in 7-bit ASCII
- SMTP server uses CRLF.CRLF to determine end of message

- **Comparison with HTTP**
  - HTTP: pull
  - SMTP: push
  - Both have ASCII command/response interaction, status codes
  - HTTP: each object encapsulated in its own response msg
  - SMTP: multiple objects sent in multipart msg

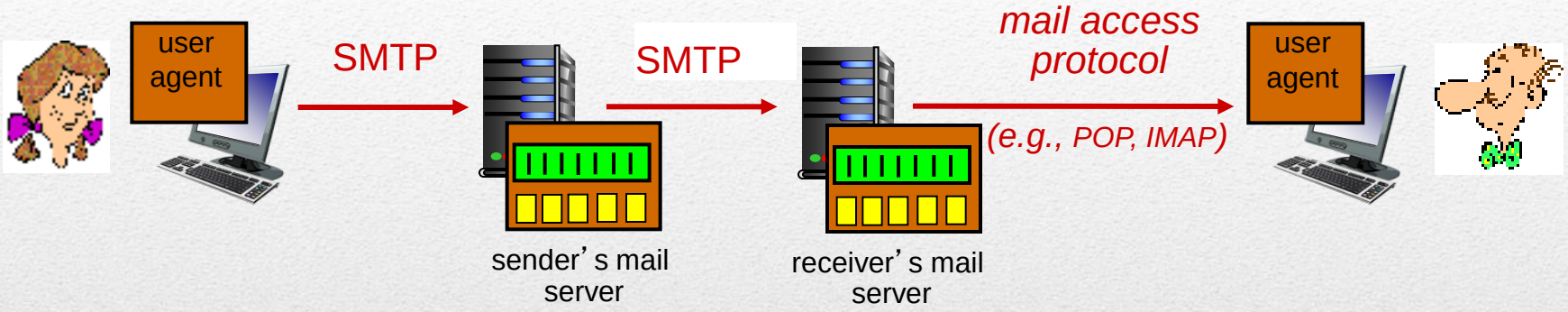
Mail Message Format

- **SMTP:** protocol for exchanging email messages
- **RFC 822:** standard for text message format:
  - header lines
    - To
    - From:
    - Subject:

  Different from SMTP MAIL FROM, RCPT TO: 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, download
  - **IMAP:** Internet Mail Access Protocol [RFC 1730]: more features, including manipulation of stored mesasges 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 & IMAP

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

- 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

DNS: Domain Name System

- **People identifiers**
  - SIN
  - Name
  - Passport Number

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

How to map between IP address and name?

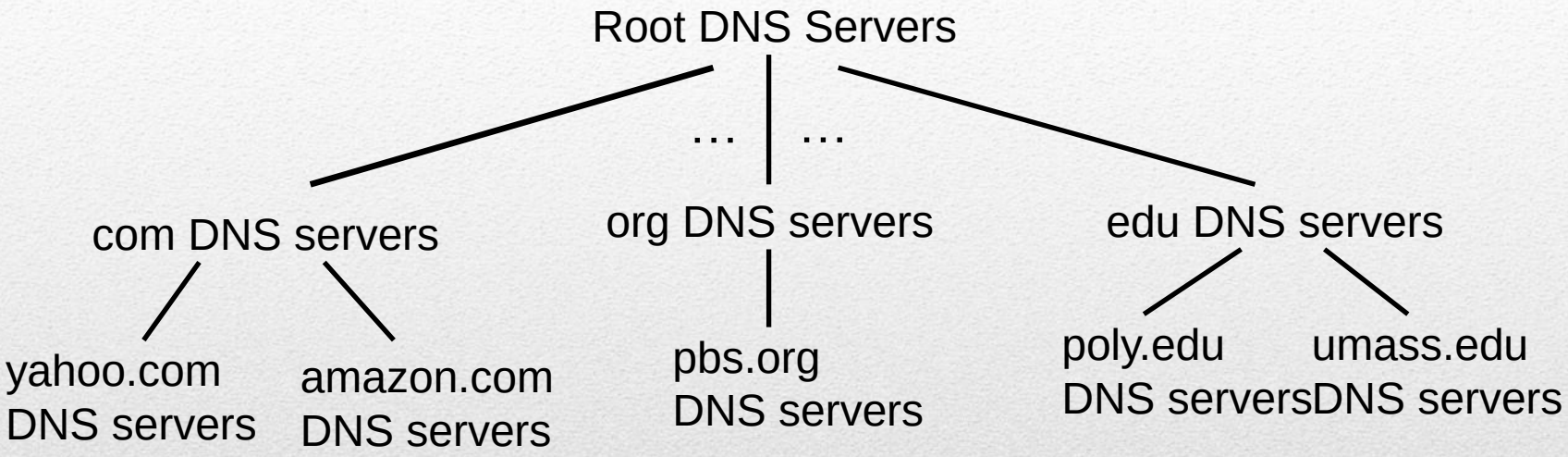
DNS: Domain Name System

- **Distributed Database**
  - Implemented in hierarchy of many name servers
- **Application Layer Protocol**
  - Host and name servers communicate to resolve names
    - Address/name translation
    - Core internet function implemented as application layer protocol
    - Complexity at network edge

DNS: Domain Name System

- **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
  - Distance Centralized Database
  - Maintenance
  - Scalability

DNS: A Distributed Hierarchical Database



Client wants IP for www.avazon.com

- 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
  - 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

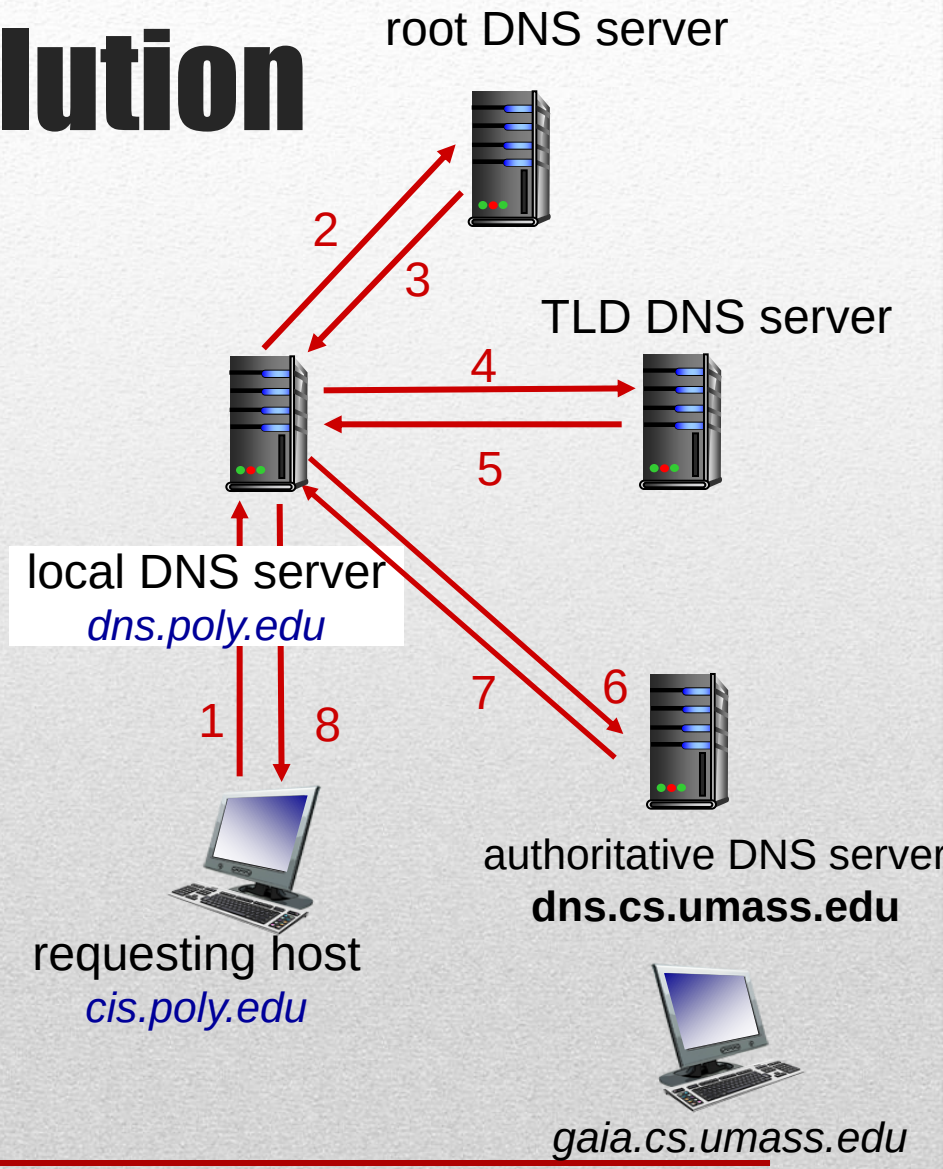
DNS Name Resolution

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"

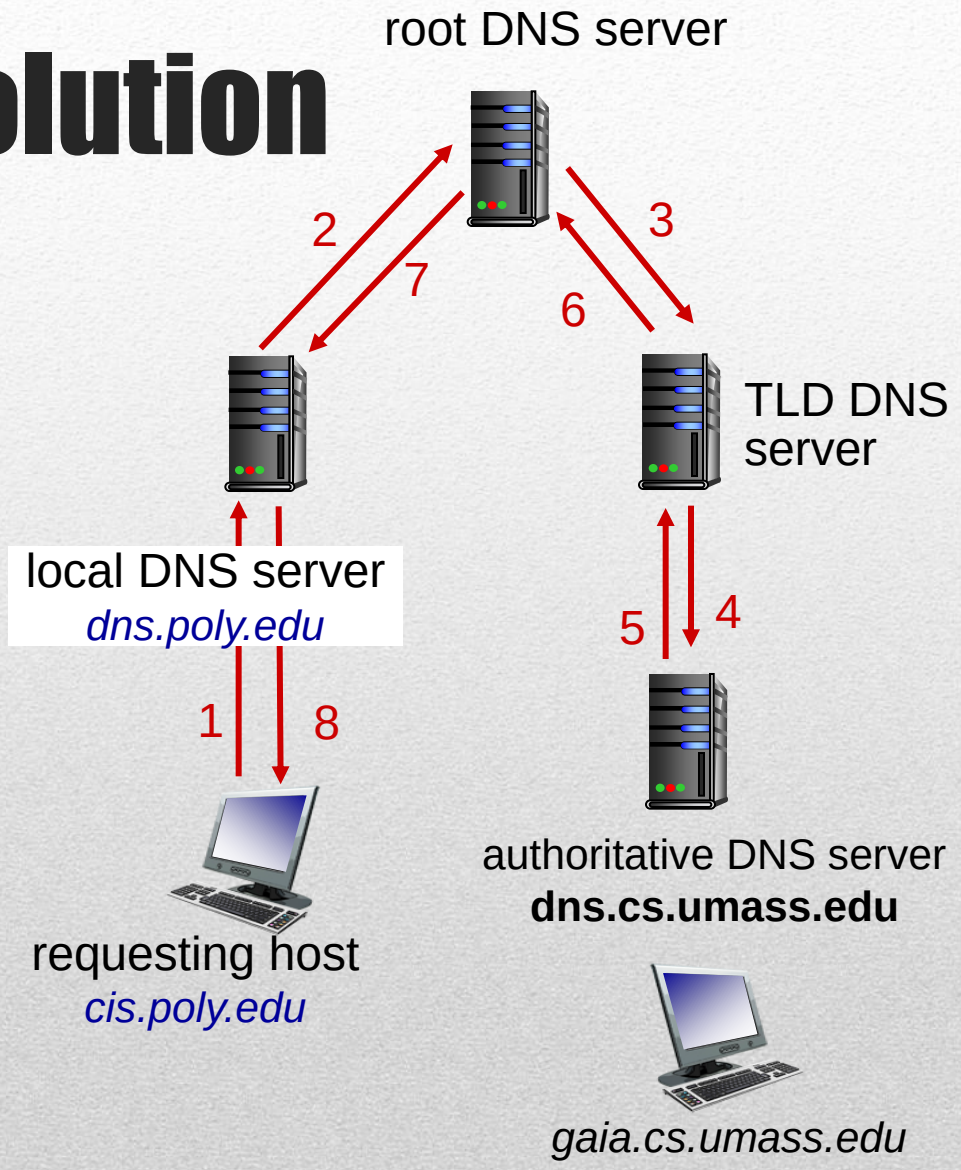


DNS Name Resolution

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 DB storing resource records (RR)

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

Type=A

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

Type=MX

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

Type=NS

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

Type=CNAME

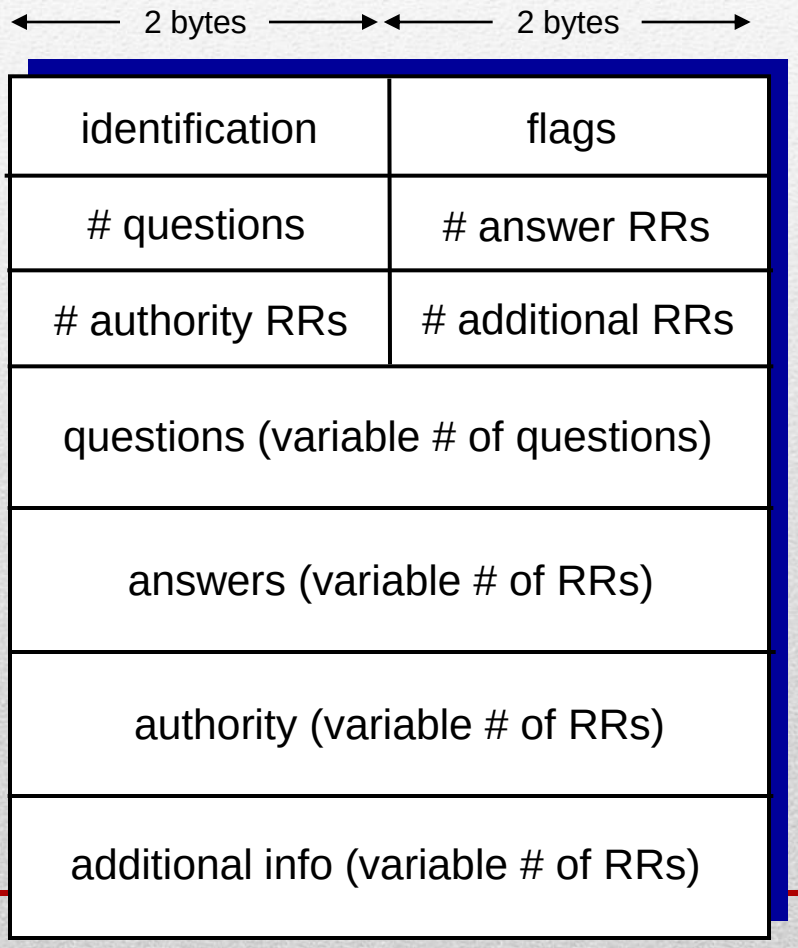
- **`name`** is alias name for some "canonical" (the real) name
- **`www.ibm.com`** is really **`servereast.backup2.ibm.com`**
- **`value`** is canonical name

DNS Protocol & Messages

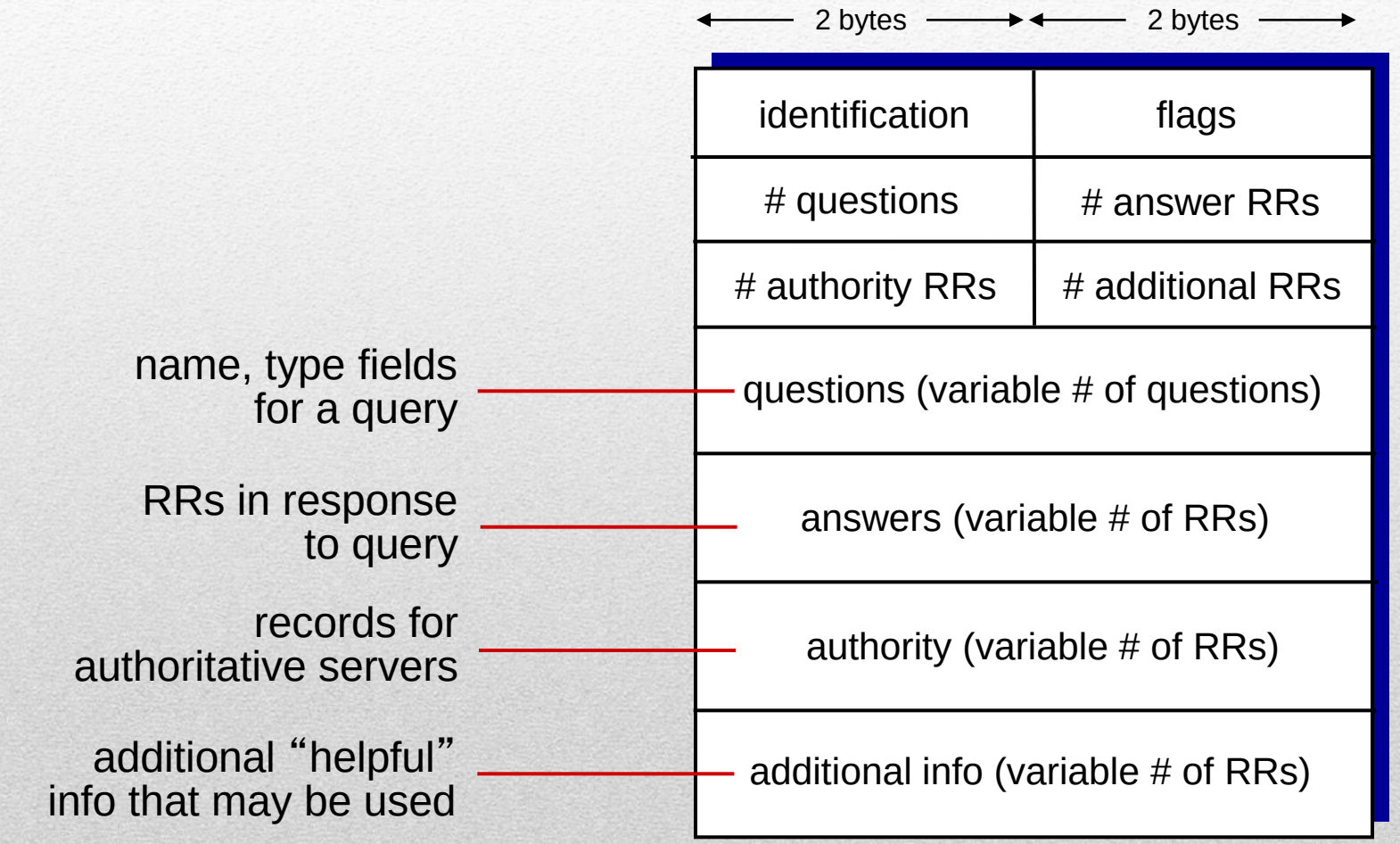
Query and *reply* messages, both with same *message format*

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