ELEC / COMP 177 – Fall 2016

Computer Networking
➔ Email (SMTP, POP, IMAP)
Domain Name System (DNS)

Some slides from Kurose and Ross, *Computer Networking*, 5th Edition

Upcoming Schedule

- **Presentation 1** – Application-Layer Protocol
  - Topic Approval – **PAST DUE**
  - Presentations – **Sept 22nd, Sept 29th, Oct 6th**
    - Upload slides to Canvas by midnight before presentation
  - Thursday, September 22nd
    - VNC (Virtual Network Computing) - **Eric Beutler**
    - File Transport Protocol (FTP) – **Jose Meza**
    - SSH - **Lonny Rasberry**
    - Skype - **Cody Berchem**
    - IRC – **Curtis Klein**
    - Bitcoin – **Kyle Phan**

Domain Name System (DNS)

Motivation

- IP addresses are hard to remember
  - `138.9.110.12`? Or was it `.21`?
- Human-friendly names are much better
  - `engineering.pacific.edu`
- How can we translate between the two?

Early Days (prior to 1984)

- Each computer on the ARPAnet (early Internet) had a single file
  - `hosts.txt` maps all known host names to IP address
- Master list maintained by SRI Network Information Center
  - Email them if your mapping changes
  - New list produced 1-2 times a week
  - All hosts download the new list
- **Problems with this approach?**

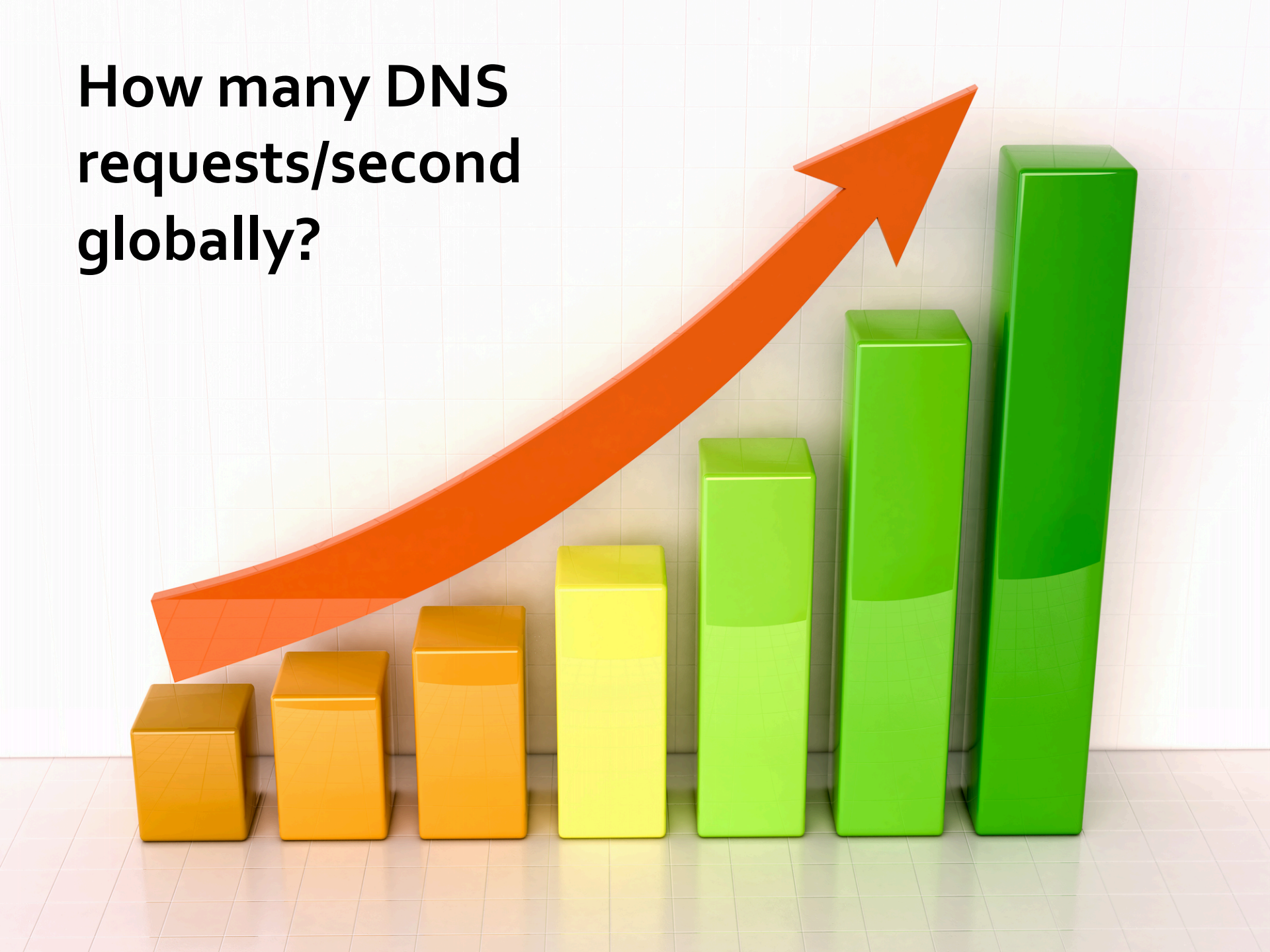
Domain Name System (DNS)

- **Distributed database** implemented in hierarchy of many **name servers**
- **Application-layer protocol**
  - Hosts, routers, and name servers communicate to resolve names (address/name translation)
  - Core Internet function, implemented as application-layer protocol
  - Complexity at network's "edge"

DNS is Decentralized

- No single point of failure
- No distant centralized database
- Easier maintenance
  - Take one or a dozen servers offline without issue
- Support high traffic volume
- *** Scalability ***

How many DNS requests/second globally?



DNS: Scalability

- **Challenging to find data on global DNS requests/sec**
  - No global internet "dashboard"
  - Internet is a "network of networks"
- Would have to inquire with AT&T, Comcast, TimeWarner, Pacific, etc
  - They would have to check stats on all of their local servers
- **Google Public DNS**
  - 400 billion requests/day as of Dec 2014
  - 70% international
  - http://googlewebmastercentral.blogspot.com/2014/12/google-public-dns-and-location.html
- **OpenDNS**
  - 80 billion requests/day as of Sept 2015
  - http://system.opendns.com/

What's in a Name?

- `engineering.pacific.edu`
  - .edu is top-level domain
  - "pacific" belongs to .edu
  - "engineering" belongs to "pacific"
  - Hierarchical! Read from right to left
- Limits?
  - Up to 127 levels of hierarchy
  - Each label can have up to 63 characters
  - Full domain name cannot exceed 253 characters

DNS: Services

- Hostname to IP address translation
  - *"www.pacific.edu" is 138.9.110.12*
- Hostname aliasing
  - Canonical, alias names
- Hostname load distribution
  - Replicated servers – Multiple IP addresses available for one name
  - *"google.com" is 74.125.239.128 or 74.125.239.135 or … or …. or … or ….*

DNS: Services

- Mail server aliasing
  - What are the **multiple** host names that receive mail for this domain?
  - 1st priority, then 2nd backup, then 3rd backup, etc…
  - Allows you to use 3rd party email services (e.g. Google Apps)
  - *Mail to "pacific.edu" is directed to "d73442a.ess.barracudanetworks.com" (SPAM filtering)*
- Other / Misc
  - SPF entries for email (Anti-spam)
  - DNSSec (security/encryption)
  - Many other attributes…

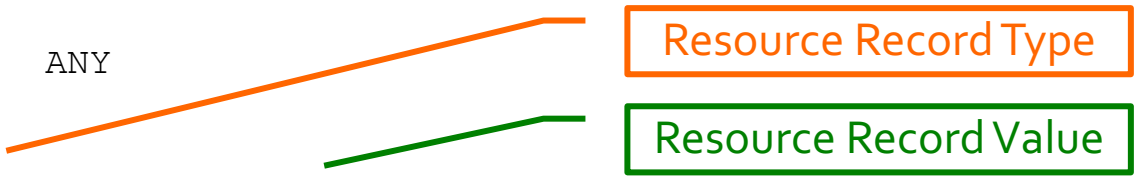
DNS: Record Types (Distributed Database)

Resource Record (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" (real) name
  - value is canonical name
- Type=**MX**
  - *value* is name of **mailserver** associated with name
- Type=**TXT**
  - *value* is machine readable text (arbitrary)

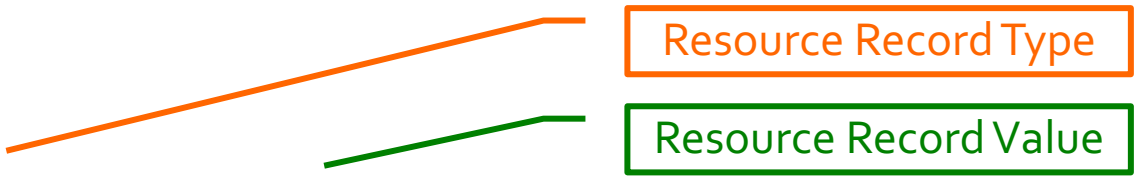
DNS: Example

```
$ dig pacific.edu any

; <<>> DiG 9.8.3-P1 <<>> pacific.edu any
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5270
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;pacific.edu.                   IN   ANY

;; ANSWER SECTION:
pacific.edu.        59   IN   A    138.9.110.12
pacific.edu.        21599    IN   NS   ns-110.awsdns-13.com.
pacific.edu.        21599    IN   NS   ns-1289.awsdns-33.org.
pacific.edu.        21599    IN   NS   ns-2044.awsdns-63.co.uk.
pacific.edu.        21599    IN   NS   ns-705.awsdns-24.net.
pacific.edu.        899  IN   SOA  ns-110.awsdns-13.com. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400
pacific.edu.        299  IN   MX   10 d73442a.ess.barracudanetworks.com.
pacific.edu.        299  IN   MX   10 d73442b.ess.barracudanetworks.com.
pacific.edu.        299  IN   TXT  "v=spf1 ip4:138.9.240.95 ip4:138.9.110.64
ip4:138.9.110.74 include:_spf.google.com
include:spf.protection.outlook.com include:_spf.qualtrics.com ~all"
```

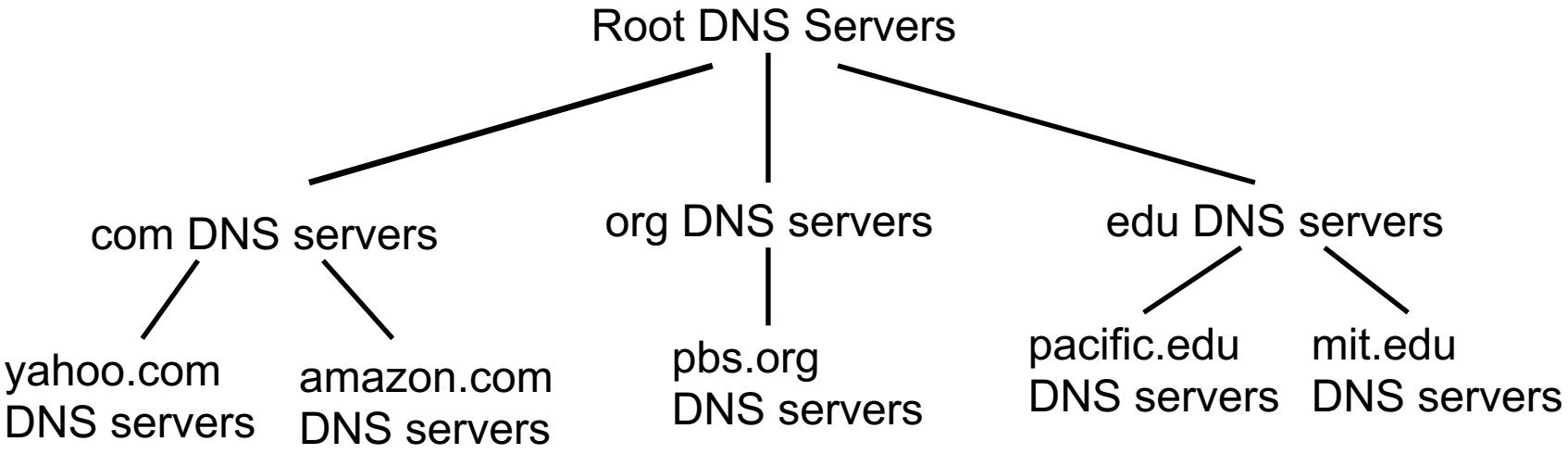


DNS: Example

```
$ dig google.com all

; <<>> DiG 9.8.3-P1 <<>> google.com all
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33808
;; flags: qr rd ra; QUERY: 1, ANSWER: 11, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;google.com.                    IN   A

;; ANSWER SECTION:
google.com.             299    IN   A    74.125.239.128
google.com.             299    IN   A    74.125.239.135
google.com.             299    IN   A    74.125.239.132
google.com.             299    IN   A    74.125.239.133
google.com.             299    IN   A    74.125.239.136
google.com.             299    IN   A    74.125.239.134
google.com.             299    IN   A    74.125.239.142
google.com.             299    IN   A    74.125.239.129
google.com.             299    IN   A    74.125.239.130
google.com.             299    IN   A    74.125.239.137
google.com.             299    IN   A    74.125.239.131
```



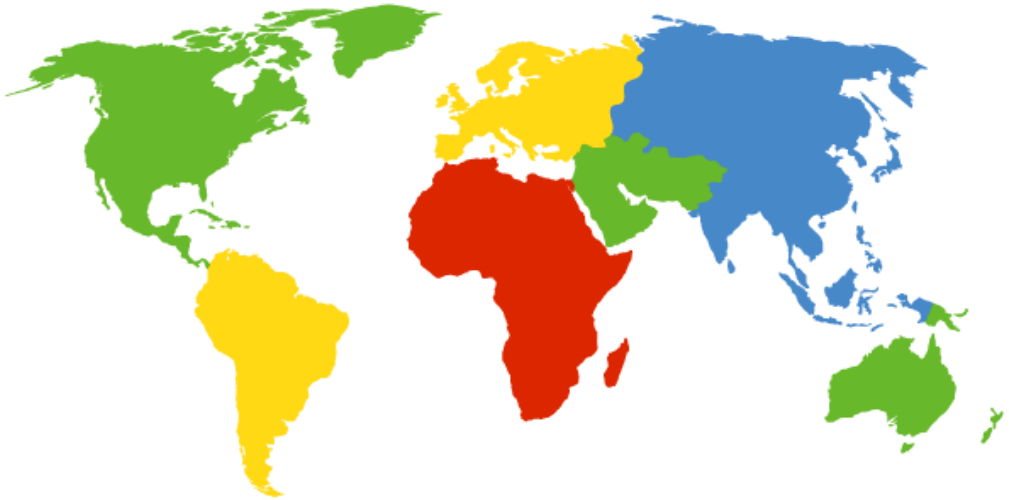
Distributed, Hierarchical Database



- Client wants IP for www.amazon.com
  1. Client queries a root server to find com DNS server
  2. Client queries com DNS server to get amazon.com DNS server
  3. Client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: Root Name Servers

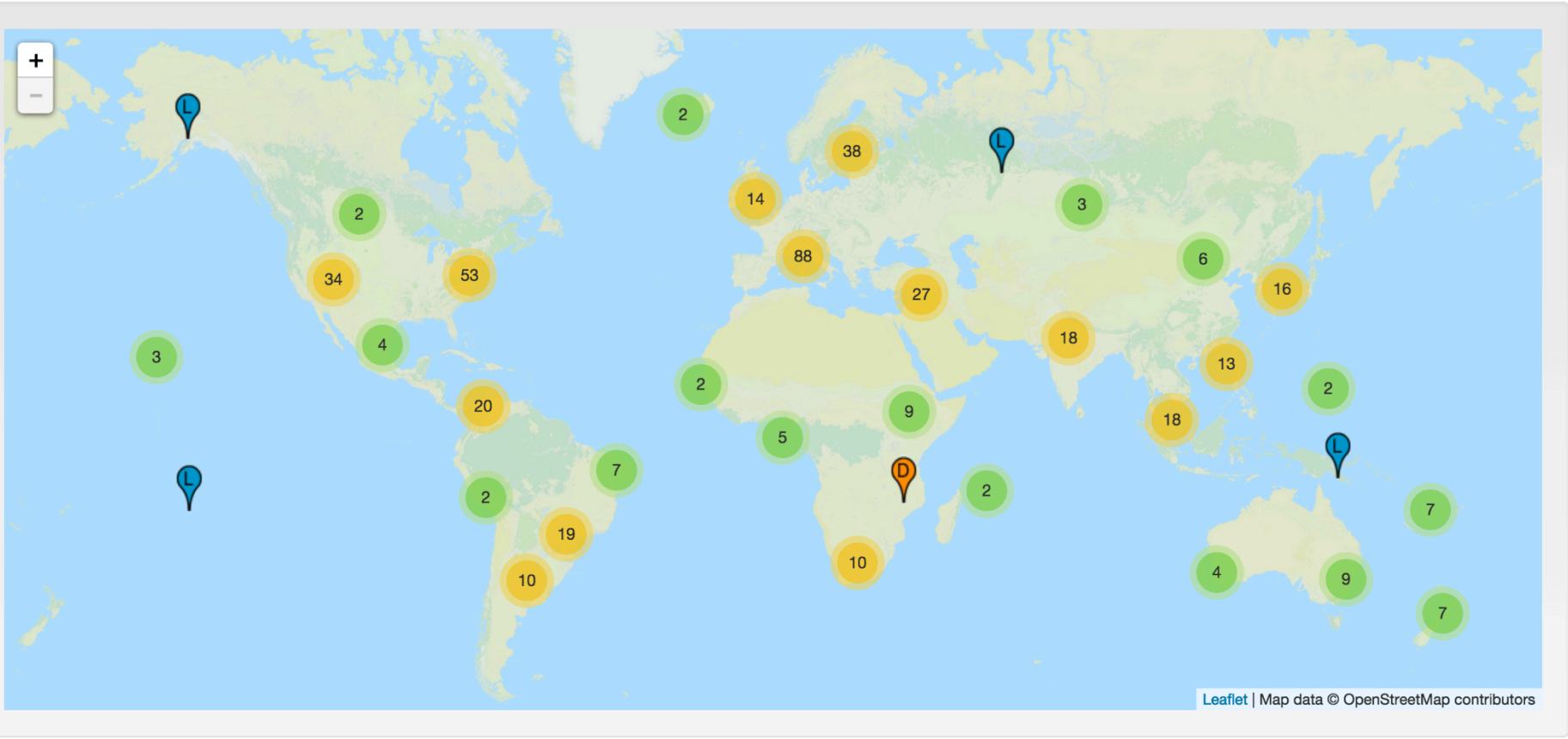
- Contacted by local name server that can not resolve top-level domain
- Root name server:
  - Contacts authoritative name server for TLD if name mapping not known
  - Gets mapping
  - Returns mapping to local name server



13 root name "servers" worldwide labeled `a` - `m`

- Each "server" is really a cluster
- Some clusters are geographically distributed
- 504 total in Fall 2014

DNS: Root Name Servers



http://www.root-servers.org/

TLD and Authoritative Servers

- **Top-level domain** (TLD) servers
  - Responsible for com, org, net, edu,… and all top-level country domains (uk, fr, ca, jp, …)
  - Server maintainers
    - VeriSign for `.com`, `.net` TLDs
    - Educause for `.edu` TLD
- **Authoritative** DNS servers:
  - Organization's DNS servers, providing authoritative hostname to IP mappings for organization's servers
  - Can be maintained by organization or service provider

Local Name Server (Cache)

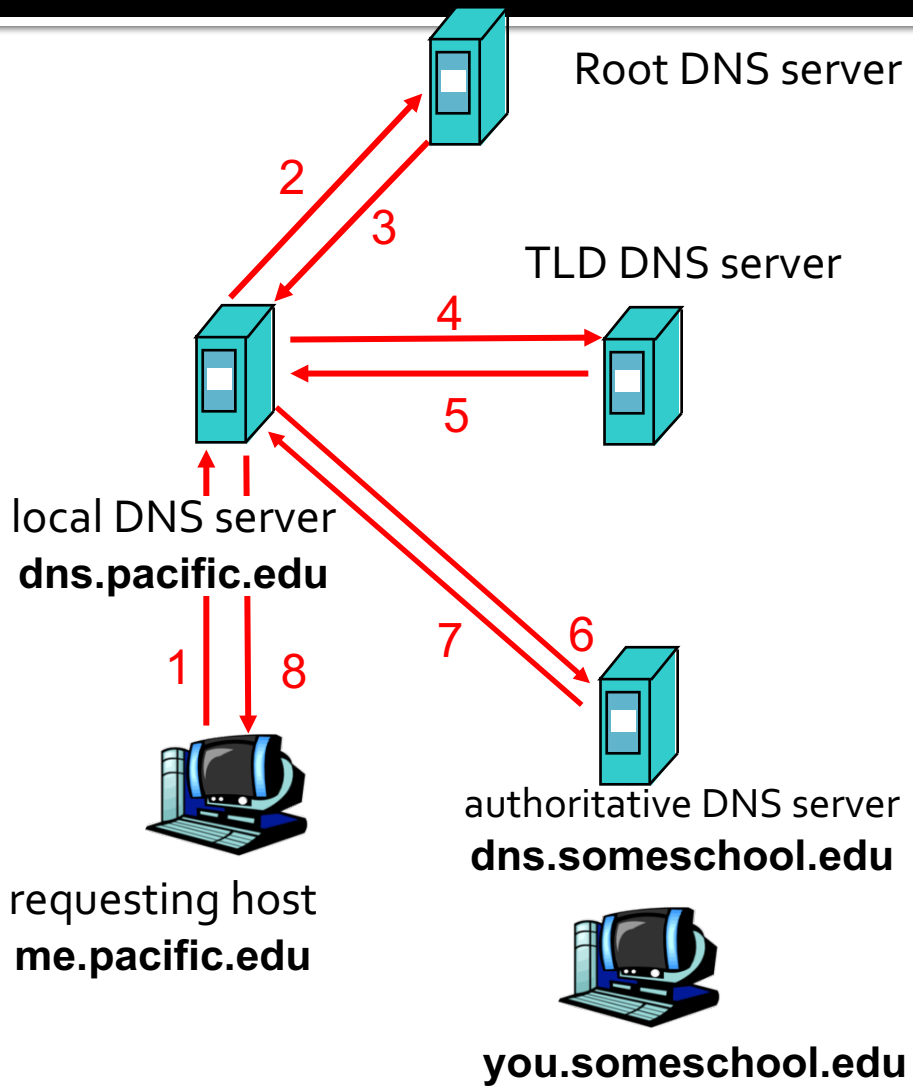
- Not part of previous hierarchy
- Each ISP (residential ISP, company, university) has one or more
- When host makes DNS query, query is sent to its local DNS server
  - Maintains local cache of common DNS records
    - *www.facebook.com?*
  - Acts as proxy, forwards query into hierarchy and provides eventual reply
- **You typically know this server's IP address from DHCP (upon connecting to the network)**

DNS: Name Resolution

- Two types
- **Recursive**
  - The server you contact provides the final answer
  - *Behind the scenes, it may make several consecutive requests*
- **Iterative**
  - The server you contact directs you to a different server to get (closer to) the final answer

DNS: Iterative and Recursive Query

- Host at `me.pacific.edu` wants IP address for `you.someschool.edu`
- Local DNS server
  - Provides *recursive* service for the requesting host
  - Makes *iterative* queries
    - Contacted server replies with name of server to contact
    - "I don't know this name, but ask this server"



DNS: Caching and Updating records

- Once (any) name server learns mapping, it **caches** mapping
  - This includes your computer
  - This includes your ISP's name server
- Cache entries eventually timeout
  - Can be specified by the authoritative server, and/or overruled by the local server
- TLD (.com, .net, .org, etc…) servers are typically cached in local name servers
  - Reduces traffic on the root servers!

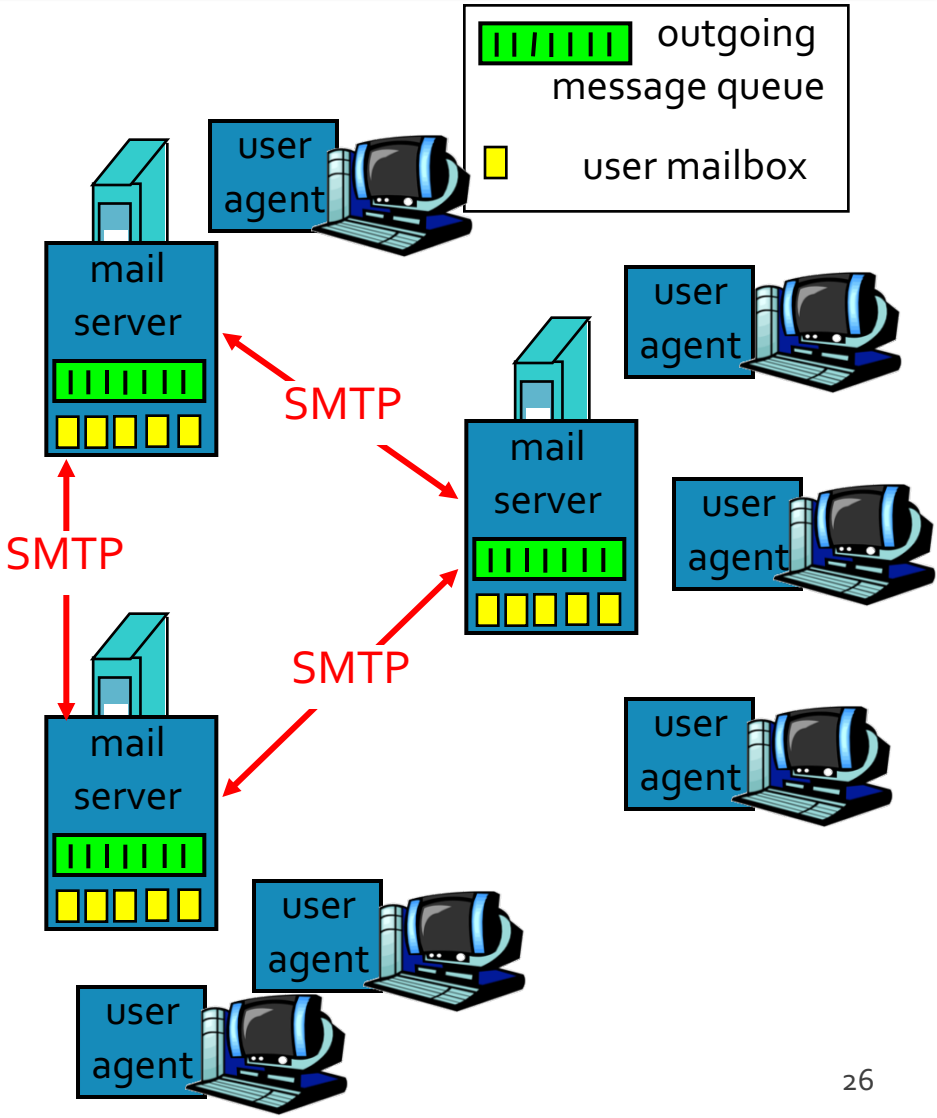
DNS and UDP

- DNS uses UDP by default
  - It *can* use TCP, but it's rare
  - **Isn't this unreliable?**
- Why use UDP
  - Faster (in three ways!)
    - No need to establish a connection (RTT/latency overhead)
    - Lower per-packet byte overhead in UDP header
    - Less packet processing by hosts
  - Reliability not needed
    - DNS will just re-request if no response received (2-5 seconds)

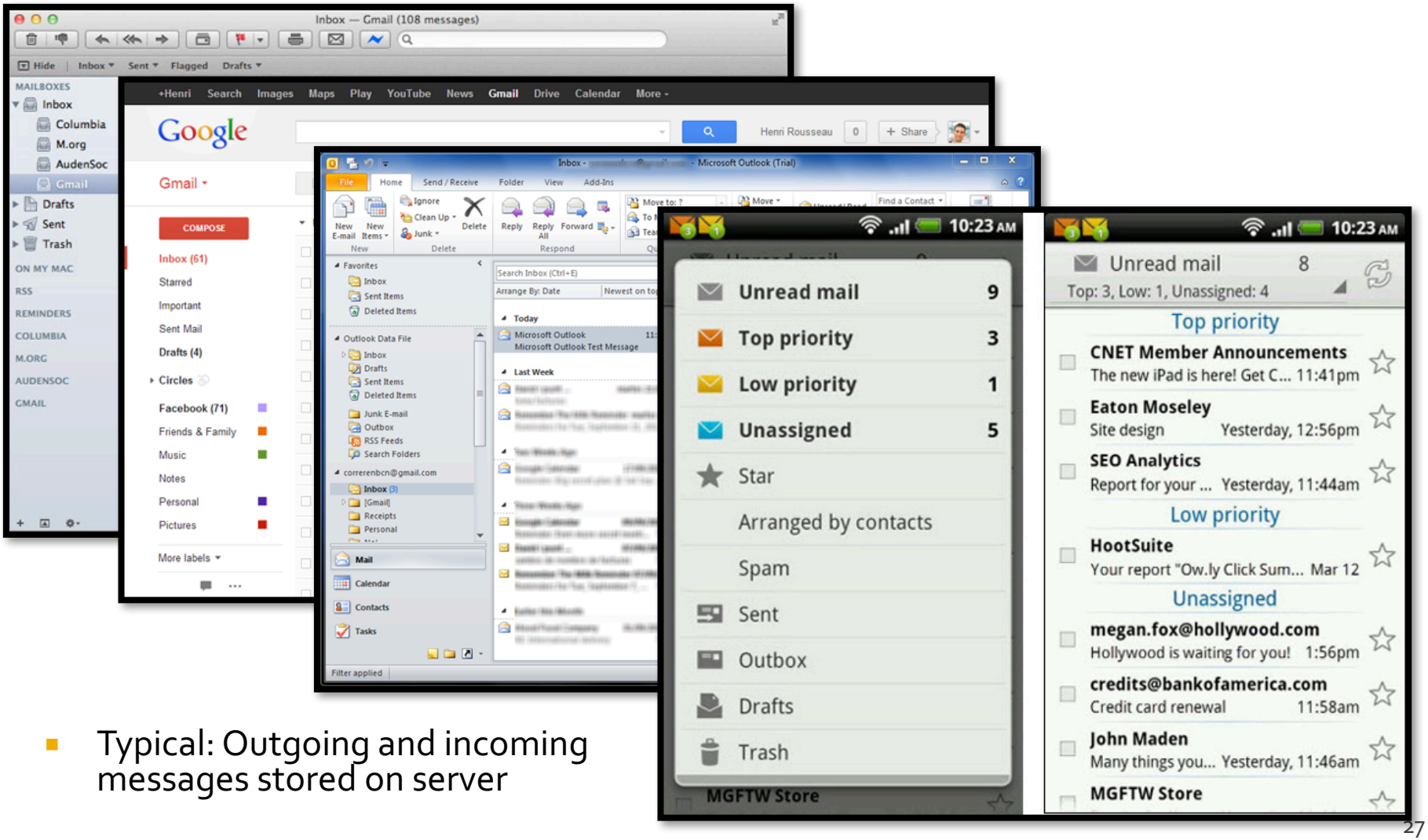
Email (SMTP, POP, IMAP)

Electronic Mail

- **Major components**
  - User agents
  - Mail servers
  - Protocol for message transfer (SMTP)
  - Protocol for message access (IMAP, MAPI, POP)



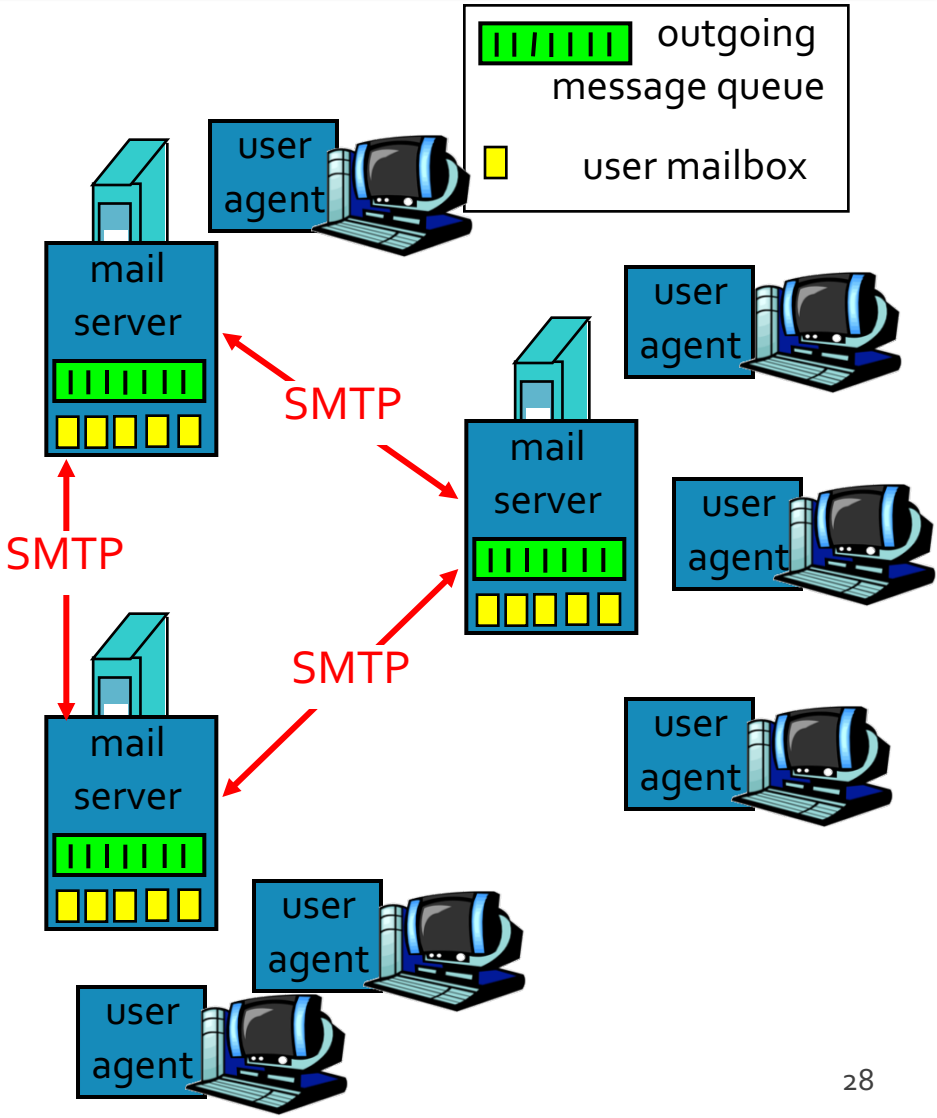
Electronic Mail – User Agent



- Typical: Outgoing and incoming messages stored on server

Electronic Mail – Mail Servers

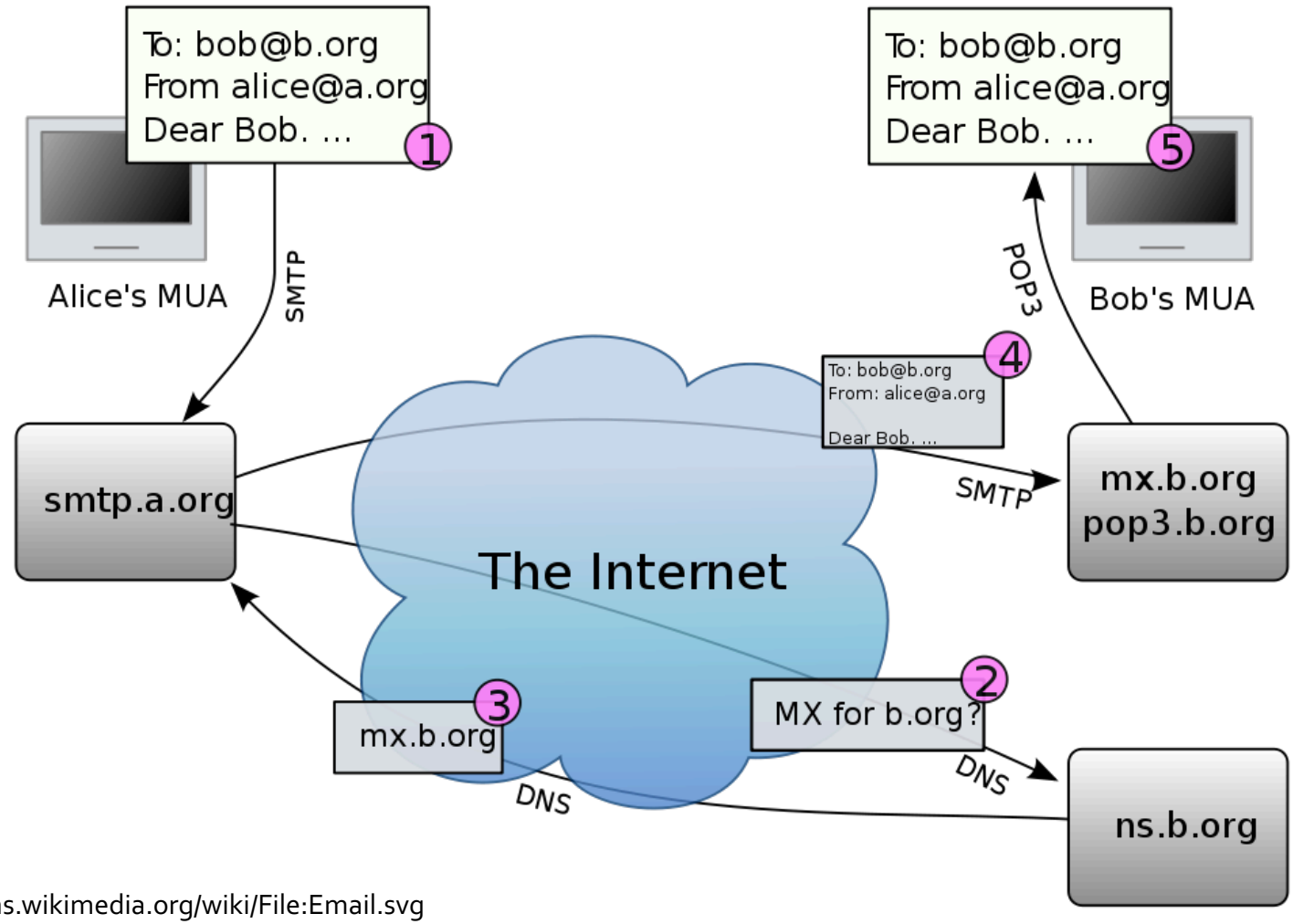
- **Mail Servers**
  - **Mailbox** contains incoming messages for user
  - **Message queue** of outgoing mail messages (to be sent)
- **SMTP protocol**
  - Used to move email messages between mail servers
  - Client: sending mail server
  - Server: receives messages



Simple Mail Transport Protocol (SMTP)

- Uses TCP (port 25) to reliably transfer email message from user agent to server, or direct from server to 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
  - Binary attachments are Base64 *encoded*

Typical SMTP Usage



https://commons.wikimedia.org/wiki/File:Email.svg

Sample SMTP interaction

S=Server, C=Client

```
S: 220 bigschool.edu
C: HELO smallschool.edu
S: 250  Hello smallschool.edu, pleased to meet you
C: MAIL FROM: <alice@smallschool.edu>
S: 250 alice@smallschool.edu... Sender ok
C: RCPT TO: <bob@bigschool.edu>
S: 250 bob@bigschool.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: This is a test message
C: This is still a test message
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 bigschool.edu closing connection
```

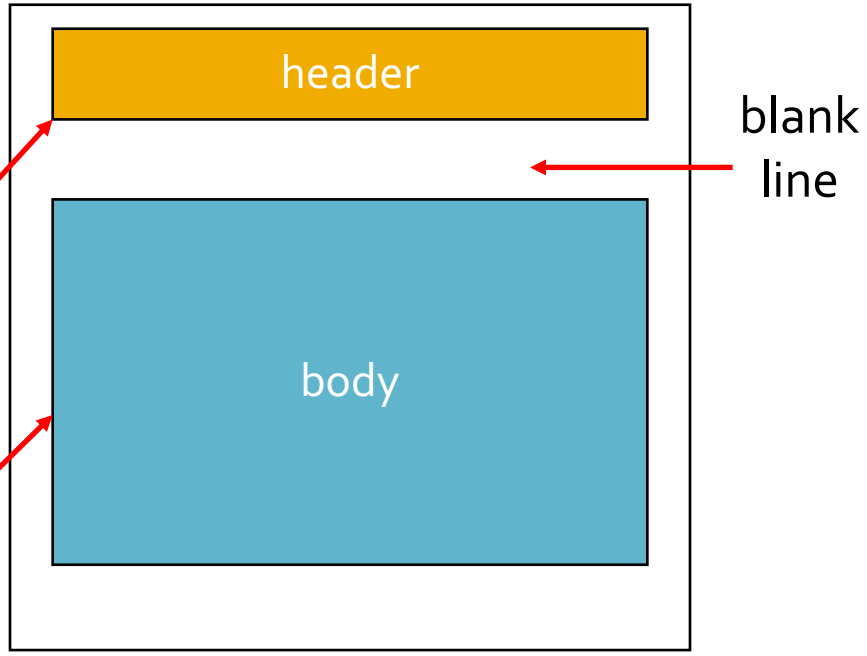
SMTP server uses CRLF.CRLF to determine end of message

SMTP versus HTTP

- "Direction" of transfer
  - HTTP: pull *from* server *(at least, HTTP GET)*
  - SMTP: push *to* server
- Protocol "style"
  - Both have ASCII command/response interaction and status codes
- Granularity
  - HTTP: each object encapsulated in its own response message *(version 1.0 only)*
  - SMTP: multiple objects sent in multipart message

Mail Message format

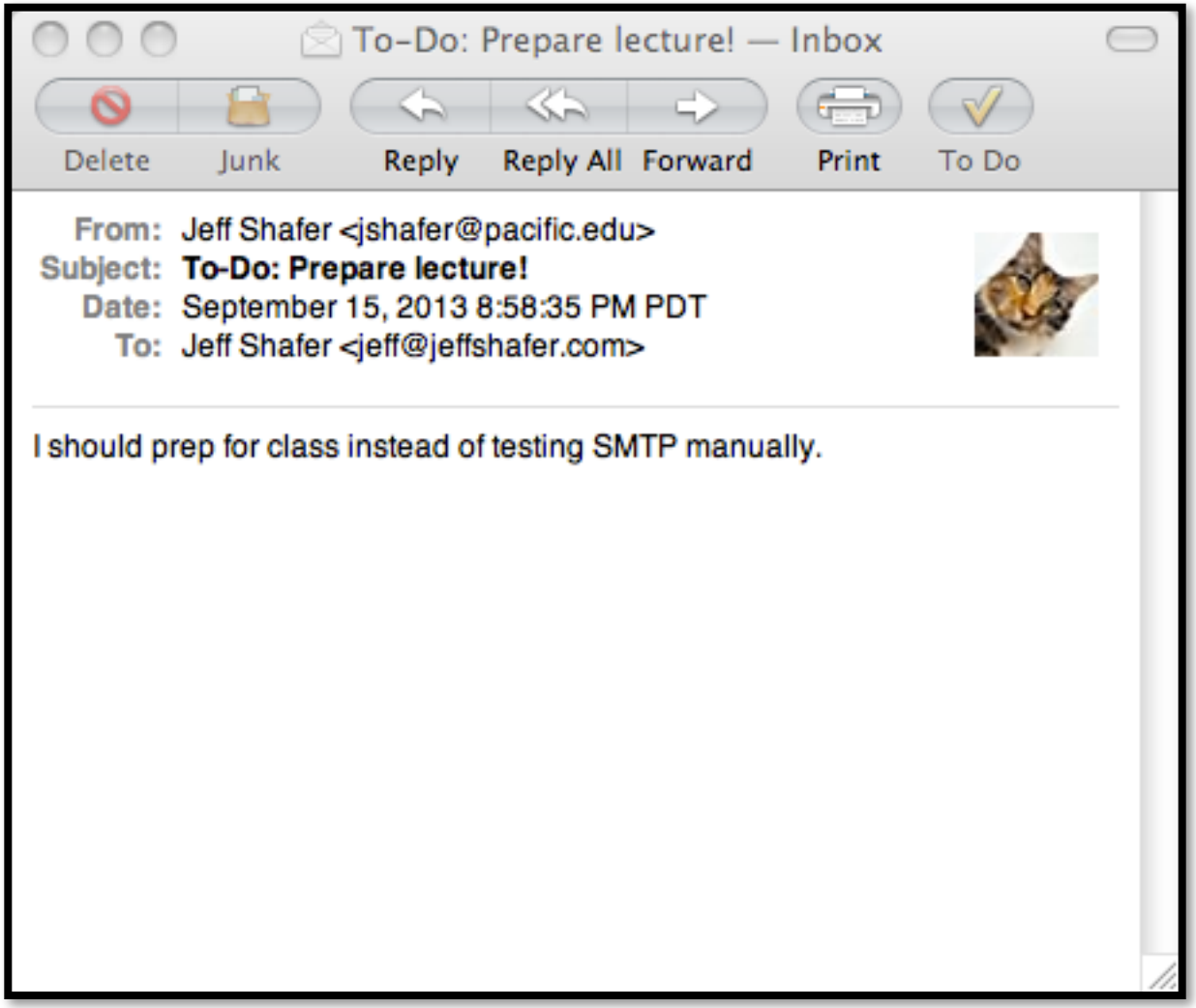
- SMTP defines exchanging messages between systems *(transport)*
  - It does **not** specify the format for data inside the message! *(content)*
- RFC 822 defines a standard for text message format
- Header lines
  - To / From / Subject / …
  - **Different from SMTP commands!**
- Body
  - The "message"



SMTP + RFC 822 Manually

```
tiger [~] <!> telnet smtp.pacific.edu 25
Trying 192.168.100.100...
Connected to smtp.pacific.edu.
Escape character is '^]'.
220 mx20.pacific.edu ESMTP
HELO pacific.edu
250 mx20.pacific.edu
MAIL FROM: <jshafer@pacific.edu>
250 2.1.0 Ok
RCPT TO: <jeff@jeffshafer.com>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
To: "Jeff Shafer" <jeff@jeffshafer.com>
From: "Jeff Shafer" <jshafer@pacific.edu>
Subject: To-Do: Prepare lecture!

I should prep for class instead of testing SMTP manually.
.
250 2.0.0 Ok: queued as 9BD3478EC
QUIT
221 2.0.0 Bye
```

SMTP Manually – The result!



SMTP and SPAM

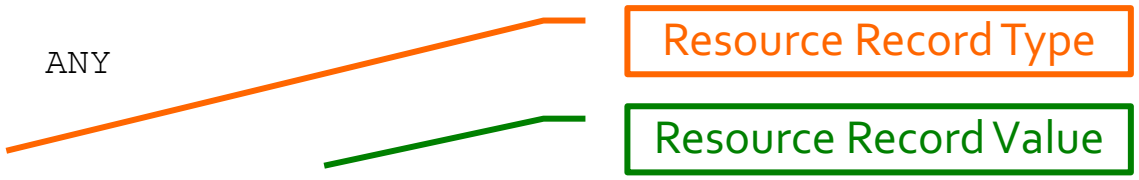
- Telnet example – did not have to log in!
  - Security an afterthought in original design
- Open relay
  - SMTP server that sends mail to *all* destinations for *all* clients
  - Typically blacklisted today in spam filters
- Optional security measures
  - Only accept clients inside your network?
    - `smtp.pacific.edu` will not respond on port 25 when I'm at home
  - Only accept destinations inside your network?
  - Require users to login? (ESMTP)

SMTP and SPAM

- You can lie to an SMTP server
  - Instead of claiming to be jshafer@pacific.edu, I could have said I was president@pacific.edu
- Countermeasures?
  - `smtp.pacific.edu` could prevent this by forcing me to log on
- What if I send mail via my own SMTP server?
  - Spam filter **challenge**
  - SPF – Sender Protection Framework
    - Puts notes into DNS specifying which IPs are allowed to send mail claiming to be from `pacific.edu`

DNS: SPF Data

```
$ dig pacific.edu any

; <<>> DiG 9.8.3-P1 <<>> pacific.edu any
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5270
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;pacific.edu.                   IN   ANY

;; ANSWER SECTION:
pacific.edu.      59   IN   A    138.9.110.12
pacific.edu.      21599    IN   NS   ns-110.awsdns-13.com.
pacific.edu.      21599    IN   NS   ns-1289.awsdns-33.org.
pacific.edu.      21599    IN   NS   ns-2044.awsdns-63.co.uk.
pacific.edu.      21599    IN   NS   ns-705.awsdns-24.net.
pacific.edu.      899  IN   SOA  ns-110.awsdns-13.com. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400
pacific.edu.      299  IN   MX   10 d73442a.ess.barracudanetworks.com.
pacific.edu.      299  IN   MX   10 d73442b.ess.barracudanetworks.com.
pacific.edu.      299  IN   TXT  "v=spf1 ip4:138.9.240.95 ip4:138.9.110.64
ip4:138.9.110.74 include:_spf.google.com
include:spf.protection.outlook.com include:_spf.qualtrics.com ~all"
```



Mail Access Protocols

- **SMTP:** delivery/storage to receiver's server
- Mail access protocol: **retrieval from server**
  - **POP:** Post Office Protocol
    - Authorization (agent <-->server) and download
  - **IMAP:** Internet Mail Access Protocol
    - More features (more complex)
    - Manipulation of stored messages on server
  - **MAPI:** Messaging Application Programming Interface
    - Microsoft outlook
  - **HTTP:** Gmail, Outlook.com, Yahoo! Mail, etc…

Post Office Protocol (POP3)

- "Classic" email – infrequently used today
- Modes:
  - "Download and delete from server" mode.
    - Only suitable for 1 email client
  - "Download and keep on server" mode
    - Allows copies of messages on different clients

Internet Message Access Protocol (IMAP)

- Keep all messages in one place: the server
  - Clients might have a temporary *cache* for offline access
- Allows user to organize messages in folders
- IMAP keeps user state across sessions:
  - Names of folders and mappings between message IDs and folder name
- Other features
  - Server-side searches (don't have to download mailbox!)
  - Multiple concurrent clients